Trimmed Conformal Prediction for High-Dimensional Models

Wenyu Chen¹, Zhaokai Wang¹, Wooseok Ha¹, and Rina Foygel Barber¹

¹Department of Statistics, University of Chicago

December 1, 2016

Abstract

In regression, conformal prediction is a general methodology to construct prediction intervals in a distribution-free manner. Although conformal prediction guarantees strong statistical property for predictive inference, its inherent computational challenge has attracted the attention of researchers in the community. In this paper, we propose a new framework, called Trimmed Conformal Prediction (TCP), based on two stage procedure, a trimming step and a prediction step. The idea is to use a preliminary trimming step to substantially reduce the range of possible values for the prediction interval, and then applying conformal prediction becomes far more efficient. As is the case of conformal prediction, TCP can be applied to any regression method, and further offers both statistical accuracy and computational gains. For a specific example, we also show how TCP can be implemented in the sparse regression setting. The experiments on both synthetic and real data validate the empirical performance of TCP.

1 INTRODUCTION

High-dimensional data are omnipresent in many scientific fields such as computational biology, and accordingly modern statistical inference has been evolving rapidly to develop tools that are valid in high-dimensional data. Though most research focuses explicitly on constructing and providing the confidence sets of the parameters of interest, they inevitably require strong assumptions on the data generation distribution or noise distribution. In contrast, recent work by Lei et al. [2] studies a general framework for predictive inference under no distribution assumptions and only requiring the exchangeability of the training data and a new test data point. In their work, the main tool allowing the distribution free predictive inference is a method called “conformal prediction”, first proposed by Vovk et al. [4, 5, 6] in the context of sequential classification and regression problems. To put it simply, conformal prediction is a way of exploiting the exchangeability of the residuals by treating them equally and sorting in the magnitudes—we provide a more detailed background in the next section. While conformal prediction is a powerful method to construct the prediction intervals in a distribution free manner, as pointed out by Lei et al. [2], it it usually computationally prohibitive. To avoid the computational issue, the authors also propose an alternative procedure based on sample splitting, called “split conformal prediction”. Although the sample splitting method can reduce the computational cost drastically, the obvious drawback is losing the full power of the available sample, which is problematic as it can substantially increase the width of the resulting prediction intervals.

In this paper, we address such issue and propose an efficient algorithm, Trimmed Conformal Prediction (TCP). TCP consists of a trimming step and a prediction step: in the trimming step, potential y values for the test point are rapidly preprocessed to determine which y values are most unlikely; these are then “trimmed” away. Then in the prediction step, we apply the conformal prediction algorithm only on this restricted set of y values, to construct the prediction intervals more efficiently. The key idea of TCP is that by performing conformal prediction on a small set of candidate y values, the computation can be improved significantly. Of course this comes at the price of losing some confidence
in the prediction intervals, but in practice the loss is almost negligible. Our main theoretical result provides a simple and rigorous quantification of the loss of the coverage rate due to the trimming step.

To manifest the computational benefits of our algorithm, we also apply our method to the sparse linear model; here we use the lasso \[3\] as our prediction algorithm, and show that an easy trimming step can form a relatively small set of candidate \( y \) values for the conformal prediction algorithm. In total, our algorithm requires far fewer calls of the lasso solver while losing almost no coverage rate in practice. Hebiri \[1\] also considered conformal prediction based on the lasso estimator, but they do not take account for the change of the support when adding a new test point to the training data and thus their method does not offer a guarantee of finite sample coverage. On the other hand, our method guarantees the exact finite sample coverage while enjoying fast computation. We also demonstrate the superior performance of our method through a series of numerical experiments and real data applications.

2 BACKGROUND

2.1 Sparse Models and the Lasso

In the task of predicting the response \( y \) at new feature point \( x \), linear regression is often preferred due to its simplicity and interpretability. In the high-dimensional regime where \( p > n \), it is natural to assume that the regression coefficient is sparse. Under such assumption, the most popular approach is by solving the \( \ell_1 \)-penalized least squares problem, also known as the lasso \[3\], given by

\[
\hat{\beta} = \arg \min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{2} \| Y - X \beta \|_2^2 + \lambda \| \beta \|_1 \right\},
\]

where \( Y \in \mathbb{R}^n \) is the response while \( X \in \mathbb{R}^{n \times p} \) contains the \( p \) potential features. One attractive property of the lasso \[1\] is that it selects only a few variables in the solution \( \hat{\beta} \), resulting a sparse model. While many results are known guaranteeing strong accuracy of the lasso solution when the assumed sparse linear model is true or approximately true (e.g. \[3, \[2\]), in practice we cannot know whether we are in such a setting, and assessing the predictive accuracy of the lasso without such assumptions is critical.

2.2 Conformal Prediction

Conformal prediction \[5\] seeks to provide reliable prediction intervals for new data points without assuming that the model used for prediction is necessarily true. The only assumption is that the training and test data points are exchangeable. Let \( \hat{\mu} : X \rightarrow \mathbb{R} \) denote a fitted regression function, which is determined by an unordered sample of \( n + 1 \) many \((x, y)\) points, and maps \( x \) values to predicted \( y \) values.

Let \((X_1, Y_1), \ldots, (X_n, Y_n)\) be our training data, and let \((X_{n+1}, Y_{n+1})\) be the new test point where \( X_{n+1} \) is known while \( Y_{n+1} \) is the true but unknown response value. The key idea is that if we were able to fit \( \hat{\mu} \) using this entire sample of size \( n + 1 \), \( \{(X_i, Y_i) : i = 1, \ldots, n + 1\} \), then the exchangeability of the \( n + 1 \) data points implies that the residuals, \( \hat{r}_i = Y_i - \hat{\mu}(X_i) \), are exchangeable as well. In particular, the residual \( \hat{r}_{n+1} \) of the test point is equally likely to rank anywhere in the list, and so the event that \( |\hat{r}_{n+1}| \) is in the bottom \( 1 - \alpha \) quantile of \( |\hat{r}_1|, \ldots, |\hat{r}_{n+1}| \) has probability at least \( 1 - \alpha \).

Of course, we cannot compute \( \hat{\mu} \) and \( \hat{r} \) since they depend on the unknown test point value \( Y_{n+1} \). Instead, define \( \mu_y \) to be the fitted regression function using data points \((X_1, Y_1), \ldots, (X_n, Y_n), (X_{n+1}, y)\) and let

\[
\hat{r}_y = (Y_1 - \hat{\mu}_y(X_1), \ldots, Y_n - \hat{\mu}_y(X_n), y - \hat{\mu}_y(X_{n+1}))
\]

be the residuals using this function on this set of \( n + 1 \) data points. Note that if we happen to choose
\( y = Y_{n+1} \) we recover the original values, i.e. \( \hat{\mu}_{Y_{n+1}} = \hat{\mu} \) and \( \hat{\tau}_{Y_{n+1}} = \hat{\tau} \). Then define
\[
Y = \left\{ y \in \mathbb{R} : |(\hat{\tau}_y)_{n+1}| \text{ is in the bottom } 1 - \alpha \text{ quantile of } |(\hat{\tau}_y)_1, \ldots, (\hat{\tau}_y)_{n+1}| \right\}.
\]
For \( y = Y_{n+1} \), since \( \hat{\tau}_y = \hat{\tau} \), we see that this statement must hold with probability at least \( 1 - \alpha \). That is,
\[
\mathbb{P} \{ Y_{n+1} \in Y \} \geq 1 - \alpha,
\]
giving the desired probability of coverage for the prediction interval \( Y \).

### 2.3 Split Conformal Prediction

While the conformal prediction method described above gives the desired statistical properties (namely, the correct coverage level without assuming any model or distribution), computationally it can be quite challenging. The reason is that the regression model must be fitted once for each value \( y \) which is being considered—technically, we would have to fit a model \( \hat{\mu}_y \) for all \( y \in \mathbb{R} \), although in practice a large and fine grid of \( y \) values may be used. If the model is computationally demanding, this process can be inefficient. To address this, Lei et al. [2] propose a split conformal prediction procedure, where \( \hat{\mu}_{\text{split}} \) is fitted on half the training sample, \( \{ (X_i, Y_i) : i \in I_1 \} \) where \( I_1 \subset [n] \) is a set of size \( |I_1| = n/2 \), while the remaining half is used to obtain an empirical distribution of the residuals. Specifically, defining residuals \( (\hat{\tau}_{\text{split}})_i = Y_i - \hat{\mu}_{\text{split}}(X_i) \) for \( i \in I_2 = [n] \setminus I_1 \), then the prediction interval is given by
\[
Y_{\text{split}} = \{ y \in \mathbb{R} : |y - \hat{\mu}_{\text{split}}(X_{n+1})| \text{ is in the bottom } (1 - \alpha) \text{ quantile of } |(\hat{\tau}_{\text{split}})_i| : i \in I_2 \cup \{ n+1 \} \}.
\]
As pointed out in [2] this set can simply be computed as the interval
\[
Y_{\text{split}} = \hat{\mu}_{\text{split}}(X_{n+1}) \pm \hat{\tau}_{\text{split}}^*,
\]
where \( \hat{\tau}_{\text{split}}^* \) is the \( \left( (1 - \alpha) \cdot (n/2 + 1) \right) \)-st smallest value among \( |(\hat{\tau}_{\text{split}})_i| : i \in I_2 \).

This gives the same target coverage level, while only requiring one regression \( \hat{\mu}_{\text{split}} \) to be computed; the drawback is that the intervals are likely to be wider, as the model is fitted with only \( n/2 \) many samples and is therefore less accurate.

### 3 METHOD

Before presenting our method, we motivate it with a simple example. For the original form of conformal prediction, except in some special cases like linear regression where the resulting interval can be computed in closed form, in general we would need to choose some finite range \([-y_{\max}, y_{\max}]\) within which we select a grid of \( y \) values that we test for inclusion in \( Y \). An intuitive choice for \( y_{\max} \) is to choose the largest value of the training data, \( y_{\max} = \max_{1 \leq i \leq n} |Y_i| \). We can justify this theoretically by observing that, due to exchangeability,
\[
\mathbb{P} \left\{ |Y_{n+1}| > \max_{1 \leq i \leq n} |Y_i| \right\} \leq \frac{1}{n+1},
\]
and so
\[
\mathbb{P} \{ Y_{n+1} \in Y \cap [-y_{\max}, y_{\max}] \} \geq 1 - \alpha - \frac{1}{n+1}, \tag{2}
\]
giving nearly the nominal coverage level. (Here \( Y \) is the output of conformal prediction run on the full range \( y \in \mathbb{R} \), as described in Section 2.2)

Our method, TCP, can be viewed as a generalization of this idea, and is summarized with these two steps:
1. Trimming step: apply conformal prediction with a *fast but less accurate* method to construct a prediction interval $Y_{\text{trim}}$ (which is wide, but generally will be much smaller than $[-y_{\text{max}}, y_{\text{max}}]$).

2. Prediction step: apply conformal prediction with a *slow but accurate* regression model, working only over the restricted set $Y_{\text{trim}}$.

The goal is to obtain (nearly) the same accuracy as the second (slow) model, while saving significant computation time by only fitting this slow regression model over a restricted range $Y_{\text{trim}}$.

To achieve this goal we consider two possibilities: trimming via a preliminary conformal prediction step, or a split conformal prediction step.

**Trimming with Conformal Prediction** Suppose we are equipped with two regression algorithms, one fast (and inaccurate) and one slow (and highly accurate), which each map a (unordered) data set of size $n+1$ to a predictive function, denoted as $\hat{\mu}_{\text{fast}}$ and $\hat{\mu}_{\text{slow}}$, respectively. For any $y \in \mathbb{R}$, let $(\hat{\mu}_{\text{fast}})_y$ denote the model $\hat{\mu}_{\text{fast}}$ fitted to data set $(X_1, Y_1), \ldots, (X_n, Y_n), (X_{n+1}, y)$, and let $(\hat{r}_{\text{fast}})_y$ denote the residuals of this model. Define $(\hat{\mu}_{\text{slow}})_y, (\hat{r}_{\text{slow}})_y$ analogously.

Our algorithm is given by the steps:

1. Trimming step:
   
   $$Y_{\text{trim}} = \{ y \in \mathbb{R} : |((\hat{r}_{\text{fast}})_y)_{n+1}| \text{ is in the bottom } (1 - \alpha_{\text{trim}}) \text{ quantile}$$
   
   of $|((\hat{r}_{\text{fast}})_y)_1, \ldots, ((\hat{r}_{\text{fast}})_y)_{n+1}|$.

2. Prediction step:
   
   $$Y_{\text{predict}} = \{ y \in Y_{\text{trim}} : |((\hat{r}_{\text{slow}})_y)_{n+1}| \text{ is in the bottom } (1 - \alpha_{\text{predict}}) \text{ quantile}$$
   
   of $|((\hat{r}_{\text{slow}})_y)_1, \ldots, ((\hat{r}_{\text{slow}})_y)_{n+1}|$.

Note the computational advantage: the slow model $(\hat{\mu}_{\text{slow}})_y$ only needs to be fitted over $y \in Y_{\text{trim}}$ (or rather, over a grid of $y$ values covering $Y_{\text{trim}}$), rather than over all $y \in \mathbb{R}$. (In practice, the trimming step may need to be carried out over a finite grid of $y$ values, as well.)

In fact, we can consider a special case: if we choose the fast method $\hat{\mu}_{\text{fast}}$ to simply predict a value of zero always, $\hat{\mu}_{\text{fast}} \equiv 0$, and set $\alpha_{\text{trim}} = \frac{1}{n+1}$, then this reduces to the method described earlier where conformal prediction is computed only on the empirical range $[-y_{\text{max}}, y_{\text{max}}]$, since the first step of our algorithm would compute $Y_{\text{trim}} = [-y_{\text{max}}, y_{\text{max}}]$, and would then test only those $y$ values in this range for inclusion into $Y_{\text{predict}}$.

**Trimming with Split Conformal Prediction** In some cases, we may prefer to use a split conformal prediction approach for the trimming step. In this setting, consider again two regression algorithms, $\hat{\mu}_{\text{split}}$ and $\hat{\mu}_{\text{slow}}$. We then carry out the steps:

1. Trimming step: apply split conformal prediction using $\hat{\mu}_{\text{split}}$, namely, fit $\hat{\mu}_{\text{split}}$ to the first half of the training data, then define
   
   $$Y_{\text{trim}} = \hat{\mu}_{\text{split}}(X_{n+1}) \pm \hat{r}_{\text{split}}^*$$

   where $\hat{r}_{\text{split}}^*$ is the $[(1 - \alpha_{\text{trim}})(n/2 + 1)]$-th smallest absolute residual of $\hat{\mu}_{\text{split}}$ on the second half of the training data.

2. Prediction step: same as before.
This procedure again offers a computational advantage over simply applying the costly regression method $\hat{\mu}_{\text{slow}}$ to a large grid of $y$ values (without a trimming step). Of course, we also have the option of simply using split conformal prediction; relative to this option, we instead have a statistical advantage—since we are using the full sample size $n$ to build our predictive interval $\mathcal{Y}_{\text{predict}}$, we expect to have a narrower interval than if we had used the less accurate model $\hat{\mu}_{\text{split}}$ which is fitted only on a sample of size $n/2$. We will see these tradeoffs in practice in our experiments below.

**Theoretical Guarantee**  As expected, our algorithm (in either form) offers a coverage guarantee that combines the coverage levels of the two steps:

**Theorem 1.** If the data points $(X_1, Y_1), \ldots, (X_{n+1}, Y_{n+1})$ are exchangeable, then either version of the TCP method gives coverage level

$$\mathbb{P} \{ Y_{n+1} \in \mathcal{Y}_{\text{predict}} \} \geq 1 - \alpha_{\text{trim}} - \alpha_{\text{predict}}.$$  

Before proving this result, we note that this result reduces to the coverage level given in (2) in the special case where we choose $\hat{\mu}_{\text{fast}} \equiv 0$.

**Proof.** We see that $\mathcal{Y}_{\text{trim}}$ is simply the conformal prediction interval determined by the fast regression model $\hat{\mu}_{\text{fast}}$ (in the first version of the method) or the interval determined by split conformal inference with $\hat{\mu}_{\text{split}}$ (in the second version). Furthermore, defining

$$\tilde{\mathcal{Y}}_{\text{predict}} = \left\{ y \in \mathbb{R} : \left| (\hat{r}_{\text{slow},y})_{n+1} \right| \text{ is in the bottom } (1 - \alpha_{\text{predict}}) \text{ quantile} \right\},$$

this is the conformal prediction interval calculated with the slow model $\hat{\mu}_{\text{slow}}$. Then, using existing results on conformal prediction (or split conformal prediction), we know that $\mathbb{P} \{ Y_{n+1} \in \mathcal{Y}_{\text{trim}} \} \geq 1 - \alpha_{\text{trim}}$ and $\mathbb{P} \{ Y_{n+1} \in \tilde{\mathcal{Y}}_{\text{predict}} \} \geq 1 - \alpha_{\text{predict}}$. By definition of our algorithm, we can also see that $\mathcal{Y}_{\text{predict}} = \mathcal{Y}_{\text{trim}} \cap \tilde{\mathcal{Y}}_{\text{predict}}$, which proves the desired coverage level. 

**3.1 Application to Sparse Regression with the Lasso**

For the sparse linear model setting, as mentioned before, we would often like to perform conformal prediction with the lasso but cannot afford to refit the lasso over a long list of $y$ values (e.g. over a fine grid spanning the range $[-y_{\text{max}}, y_{\text{max}}]$).

We note here that the lasso is known to be easy to solve if the support and signs of the solution $\hat{\beta}$ are known in advance—the solution then takes a closed form. After fitting the lasso to some $y_* \in [-y_{\text{max}}, y_{\text{max}}]$, we might then hope that the support and signs of $\hat{\beta}$ would remain unchanged across this entire interval; this could easily be checked using the KKT optimality conditions for the lasso solution, and we would then avoid a second call to the lasso algorithm. Unfortunately, in practice, we have found that the support changes many times (even dozens of times) across this range even in simple simulations.

As an alternative, we run our two-stage algorithm with a more aggressive trimming step to reduce the trial set farther. We now give the details.

**3.1.1 Trimming Step**

For the trimming step, we consider two possibilities:
Trimming via Ridge Regression For the ridge regression option, the fast model \( \hat{\mu}_{\text{fast}} \) is given by the penalized least squares regression

\[
\hat{\beta}_{\text{ridge}} = \arg \min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{2} \| Y - X \beta \|_2^2 + \frac{\lambda}{2} \| \beta \|_1 \right\} = (X^T X + \rho I_p)^{-1} X^T Y,
\]

where \( X \in \mathbb{R}^{(n+1) \times p} \) and \( Y \in \mathbb{R}^{n+1} \) contain the full (training and test) data. In this case we can solve for \( Y_{\text{trim}} \) in closed form, for the \( n + 1 \) residuals can be written as \( (\hat{r}_{\text{fast}})_y = u + v y \), a linear function in \( y \), where

\[
u = \begin{bmatrix} Y_{1:n} \\ 0 \end{bmatrix} - X (X^T X + \rho I_p)^{-1} X^T \begin{bmatrix} Y_{1:n} \\ 0 \end{bmatrix}, \quad v = \begin{bmatrix} 0_n \\ 1 \end{bmatrix} - X (X^T X + \rho I_p)^{-1} X_{n+1}.
\]

For each \( i \), then, the inequality \(|(\hat{r}_{\text{fast}})_y^i| \leq |(\hat{r}_{\text{fast}})_y^i|\) holds in the interval enclosed by \( \frac{u_{n+1} - u_i}{v_i v_{n+1}} \). Denote the smaller of these two values as \( c_i \) and the larger as \( d_i \); we can show that \( \max_i c_i \leq y \leq \min_i d_i \) always, since for all \( i \), the interval \([c_i, d_i]\) contains the value \( y_i = -u_{n+1}/v_{n+1} \), which is the unique value for \( y \) ensuring a zero residual value, i.e. \((\hat{r}_{\text{fast}})_y^i = 0 \). Finally, choose endpoints \( c_{\text{ridge}} \leq y \leq d_{\text{ridge}} \) such that \([c_i, d_i] \subseteq [c_{\text{ridge}}, d_{\text{ridge}}]\) is violated for exactly \( \alpha_{\text{trim}}(n + 1) \) many values \( i \in [n] \); this gives the trimmed range, \( Y_{\text{trim}} = [c_{\text{ridge}}, d_{\text{ridge}}] \). In particular, if \( \alpha_{\text{trim}} = \frac{1}{n + 1} \), then we can simply set \( Y_{\text{trim}} = [\min_{i \in [n]} c_i, \max_{i \in [n]} d_i] \).

Trimming via Split Conformal Inference with Lasso Alternately, we can apply the lasso to half of the training data, indexed by \( I_1 \subset [n] \) with \(|I_1| = n/2 \), then use this fitted model for trimming via split conformal inference. Specifically, define

\[
\hat{\beta}_{\text{trim}} = \arg \min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{2} \| Y_{1:n} - X_{I_1} \beta \|_2^2 + \lambda \| \beta \|_1 \right\}.
\]

We then set

\[
Y_{\text{trim}} = X_{n+1}^T \hat{\beta}_{\text{trim}} + \hat{r}_{\text{split}}^*.
\]

where, as before, \( \hat{r}_{\text{split}}^* \) is the \((1 - \alpha_{\text{trim}})(n/2 + 1)\)-th smallest value among

\[
\{|Y_i - X_i^T \hat{\beta}_{\text{trim}}| : i \in I_2 = [n] \setminus I_1\}.
\]

In particular, if we choose \( \alpha_{\text{trim}} = \frac{1}{n/2 + 1} \), then we obtain

\[
Y_{\text{trim}} = X_{n+1}^T \hat{\beta}_{\text{trim}} \pm \max_{i \in [n]} |Y_i - X_i^T \hat{\beta}_{\text{trim}}|.
\]

3.1.2 Prediction Step For the prediction step, we simply apply the conformal prediction algorithm using lasso regression as the base method, over only the restricted trial set \( y \in Y_{\text{trim}} \) (or, in practice, a fine grid of points over this set). Specifically, writing

\[
\hat{\beta}_y = \arg \min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{2} \| Y_{1:n} \_y - X \beta \|_2^2 + \lambda \| \beta \|_1 \right\},
\]

we define

\[
Y_{\text{predict}} = \left\{ y \in Y_{\text{trim}} : |y - X_{n+1}^T \hat{\beta}_y| \text{ is in the bottom (1 - \alpha_{\text{predict}}) quantile of} \right\}
\]

\[
|Y_1 - X_1^T \hat{\beta}_y|, \ldots, |Y_n - X_n^T \hat{\beta}_y|, |y - X_{n+1}^T \hat{\beta}_y|\).\]

\[6\]
4 EMPIRICAL RESULTS

We now test our method empirically, both on simulated data and on real data from the Capital Bikeshare program in Washington DC. Code to reproduce our experiments is available online.\footnote{Available at \url{https://www.stat.uchicago.edu/~rina/conformal.html}}

4.1 Simulation Studies

We generate simulated data to compare the performance of four algorithms: conformal inference with the lasso on the trial set \([-y_{\text{max}}, y_{\text{max}}]\) (denoted as “MaxTrim” in our results); trimmed conformal inference using ridge regression to produce \(Y_{\text{trim}}\) (“RidgeTrim”); trimmed conformal inference where \(Y_{\text{trim}}\) is obtained via split conformal inference with the lasso (“SplitTrim”); and split conformal inference with the lasso \([2]\) (“Split”). Note that MaxTrim, which performs conformal inference on the interval \([-y_{\text{max}}, y_{\text{max}}]\), is essentially how conformal prediction is implemented in practice since, for methods such as lasso that must be rerun for each new \(y\) value, we cannot perform conformal prediction over the entire real line \(y \in \mathbb{R}\).

In each case we aim for 90% coverage. For the trimmed methods, we set \(\alpha_{\text{trim}}\) as small as possible, namely \(\alpha_{\text{trim}} = \frac{1}{n+1}\) (or \(\alpha_{\text{trim}} = \frac{1}{n/2+1}\) for trimming with split conformal inference); see Section 3 for details on this. Since this is very small, we then set \(\alpha_{\text{predict}} = 0.1\) to obtain (nearly) 90% coverage.

4.1.1 Settings

We generate data from a linear model, \(Y = X\beta + \epsilon\), where:

- The \(p\) features (the columns of \(X\)) are generated either with i.i.d. \(N(0, 1)\) entries, or are generated with high correlations, with the rows of \(X\) drawn i.i.d. from a \(N(0, \Sigma)\) distribution, where \(\Sigma_{i,j} = 0.9^{|i-j|}\);
- The noise is generated either as \(\epsilon_i \sim \mathcal{N}(0, 1)\) or \(\epsilon_i \sim \mathcal{t}(5)\), the \(t\) distribution with 5 degrees of freedom;
- The true signal \(\beta\) is given \(\beta_i = 2\) for \(i \in S\), where the true support \(S\) is a set of size \(k\) chosen at random, and \(\beta_i = 0\) for \(i \notin S\).

Our methods will be run with desired level \(\alpha = 0.1\). We use \(\lambda = \sqrt{n \log(p)}\) for normal noise case, and \(\lambda = \sqrt{\frac{n}{2} \log(p)}\) for \(t\) noise case (when we use split conformal inference, and the effective sample size is \(n/2\), we use \(\lambda = \sqrt{(n/2) \log(p)}\) or \(\sqrt{\frac{5}{3}(n/2) \log(p)}\)).

4.1.2 Results

Table I displays our results for the setting \(n = 200, p = 2000, k = 10\), across both feature and noise models; we record the coverage level for each method, as well as the width of the trial set \(Y_{\text{trim}}\) (which should be interpreted as the computational cost—note that this is not relevant for split conformal inference, which only ever fits the model once), and the width of the resulting prediction interval \(Y_{\text{predict}}\). Results are averaged over 500 trials.

In Figures I and II we show the widths of \(Y_{\text{predict}}\) and \(Y_{\text{trim}}\) when the number of features varies, \(p = 100, 200, \ldots, 3000\), with \(n = 200\) and \(k = 10\) fixed. We do the same in Figures III and IV when instead \(n = 200, p = 2000\) are fixed and we vary the true model size, \(k = 2, 4, \ldots, 30\). For these figures, we show results only for the normal noise model and the uncorrelated features model. Results are averaged over 500 trials.

From the simulation, we see that the three TCP methods (MaxTrim, RidgeTrim, and SplitTrim) give substantially narrower prediction intervals than split conformal prediction. With more features
Table 1: Results of the simulated data experiment, for uncorrelated or highly correlated features, and for \( N(0,1) \) or \( t(5) \) distributed noise, for settings \( n = 200, p = 2000, k = 10 \). Desired coverage level is \( 1 - \alpha = 90\% \). “Trial set width” is the width of \( \hat{Y}_{\text{trim}} \) for the trimmed conformal prediction methods, and should be viewed as a proxy for computational cost.

Figure 1: Width of prediction intervals (PIs) for all methods, as the number of features \( p \) varies. The results for MaxTrim, RidgeTrim, and SplitTrim are nearly identical and cannot be distinguished in the plot.

Figure 2: Width of trial sets for all methods, as the number of features \( p \) varies; this width is a proxy for computational cost. (The split conformal inference method is not included as it only fits the lasso model once in any setting.)

\( p \) or a larger true model size \( k \), the gap between split conformal inference and the TCP methods increases—this is intuitive, as using the full sample size becomes more important for accuracy as the
Figure 3: Width of prediction intervals (PIs) for all methods, as the size $k$ of the true model varies. The results for MaxTrim, RidgeTrim, and SplitTrim are nearly identical and cannot be distinguished in the plot.

Figure 4: Width of trial sets for all methods, as the size $k$ of the true model varies; this width is a proxy for computational cost. (The split conformal inference method is not included as it only fits the lasso model once in any setting.)

Problem becomes more difficult.

In terms of computational cost, among the three trimmed methods, SplitTrim is always the most aggressive, giving the sharpest (smallest) $\lambda_{trim}$. RidgeTrim is more conservative for this data, but nonetheless improves over MaxTrim.

The coverage is near the desired level of 90% for all methods in all settings.

4.2 Real Data Analysis

In this data analysis, we study the rental records of the Capital Bikeshare program in Washington DC. The data consists of bike rental records noting the date, time, and locations of the rental and return of each bicycle used in the program. We aggregate this data to record only the total number of rentals originating at each station on each day. We will choose one station to use as the response and the rest will be features; that is, we will predict the number of rentals originating at one station on a given day, as a function of the number of rentals at each of the other stations.

As the business expands, more bike stations are added. Therefore, we only pick a period of time when no new stations appears: the 93 days from Nov. 7, 2010 through Feb. 7, 2011. There are 107 stations with activity during these days. Choosing one station $j_\ast \in \{1, \ldots, 107\}$ as the response and one day $i_\ast \in \{1, \ldots, 93\}$, as the test point, we therefore have $p = 106$ many stations given by the remaining features $\{1, \ldots, 107\} \backslash \{j_\ast\}$, and $n = 92$ training data points given by the remaining days $\{1, \ldots, 93\} \backslash \{i_\ast\}$.

\footnote{Capital Bikeshare data is publicly available at \url{https://www.capitalbikeshare.com/trip-history-data}}
We set $\alpha = 0.1$, and repeat our experiment with each station as the response, i.e. $j_\ast = 1, 2, \ldots, 107$, and average our results over these 107 runs. For choosing $i_\ast$, we consider two settings:

- **“Random day” setting.** We pick a random day $i_\ast$ from the 93 data points. Results are averaged over 10 randomly selected days.

- **“Last day” setting.** We pick the most recent day as the test point, i.e. $i_\ast = 93$. This is more practical for real time analysis and prediction, since in practice we would typically use past data (from the first 92 days) to predict upcoming events (the present day, i.e. the 93rd day). However, as behavior patterns may change over time, this setting violates the exchangeability assumption, and the coverage properties of the conformal inference methods may not hold.

Table 2 shows the average coverage rates for the four algorithms under the two different test methods.

|               | PI width | Trial set width | Coverage (%) |
|---------------|----------|-----------------|--------------|
| **Random day**|          |                 |              |
| MaxTrim       | 11.09    | 37.21           | 92.1         |
| RidgeTrim     | 11.08    | 21.40           | 92.1         |
| SplitTrim     | 10.96    | 18.03           | 92.1         |
| Split         | 11.47    | —               | 88.3         |
| **Last day**  |          |                 |              |
| MaxTrim       | 11.36    | 37.19           | 86.0         |
| RidgeTrim     | 11.38    | 22.34           | 86.0         |
| SplitTrim     | 11.14    | 18.15           | 86.0         |
| Split         | 11.43    | —               | 85.0         |

Table 2: Results for the Capital Bikeshare data experiment, for the “random day” and the “last day” settings. Desired coverage level is $1 - \alpha = 90\%$.

Despite slight numerical differences, in the “random day” setting, the average coverage rates are all close to the desired level $1 - \alpha = 90\%$. For the “last day” setting, however, the coverage level is noticeably lower, presumably due to the violation of exchangeability in this setting.

In this experiment, the prediction intervals are all quite similar in length (however, it is worth noting that the split conformal inference method achieves slightly lower coverage at roughly the same interval length compared to the other methods), while the trimmed conformal prediction methods indeed show substantial reduction in computation time (as measured by the length of the trial set, $Y_{\text{trim}}$) relative to conformal prediction applied to the broad interval $[-y_{\text{max}}, y_{\text{max}}]$.

5 DISCUSSION

In this paper, we present a fast two-stage algorithm for conformal prediction, called Trimmed Conformal Prediction (TCP). In the trimming step, the most unlikely $y$ values are trimmed away quickly, and in the prediction step, the conformal prediction algorithm is applied over this reduced range of potential $y$ values. Our empirical results on simulated and real data show that TCP achieves computational gains over conformal prediction without the trimming step, while offering sharper prediction intervals (a statistical advantage) compared to split conformal prediction. This is highly desirable in the high-dimensional data analysis, where we are faced with both statistical and computational challenges.

References

[1] Mohamed Hebiri. Sparse conformal predictors. *Statistics and Computing*, 20(2):253–266, 2010.
[2] Jing Lei, Max G’Sell, Alessandro Rinaldo, Ryan J Tibshirani, and Larry Wasserman. Distribution-free predictive inference for regression. *arXiv preprint arXiv:1604.04173*, 2016.

[3] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[4] Vladimir Vovk, Ilia Nouretdinov, and Alex Gammerman. On-line predictive linear regression. *The Annals of Statistics*, 37(3):1566–1590, 2009.

[5] Volodya Vovk, Alex Gammerman, and Craig Saunders. Machine-learning applications of algorithmic randomness. *Proceedings of the 16th International Conference on Machine Learning*, pages 444–453, 1999.

[6] Volodya Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*. Springer, New York, 2005.

[7] Martin J Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using constrained quadratic programming (lasso). *IEEE transactions on information theory*, 55(5): 2183–2202, 2009.

[8] Peng Zhao and Bin Yu. On model selection consistency of lasso. *Journal of Machine Learning Research*, 7:2541–2563, 2006.