Journal of Machine Learning Research 14 (2013) 3261-3294 Submitted 4/12; Revised 1/13; Published 11/13

Multivariate Convex Regression with Adaptive Partitioning

Lauren A. Hannah LAH2178@COLUMBIA.EDU
Department of Statistics
Columbia University
New York, NY 10027, USA

David B. Dunson DUNSON@STAT.DUKE.EDU
Department of Statistical Science
Duke University
Durham, NC 27708, USA

Editor: Hui Zou

Abstract

We propose a new, nonparametric method for multivariate regression subject to convexity or concavity constraints on the response function. Convexity constraints are common in economics, statistics, operations research, financial engineering and optimization, but there is currently no multivariate method that is stable and computationally feasible for more than a few thousand observations. We introduce convex adaptive partitioning (CAP), which creates a globally convex regression model from locally linear estimates fit on adaptively selected covariate partitions. CAP is a computationally efficient, consistent method for convex regression. We demonstrate empirical performance by comparing the performance of CAP to other shape-constrained and unconstrained regression methods for predicting weekly wages and value function approximation for pricing American basket options.

Keywords: adaptive partitioning, convex regression, nonparametric regression, shape constraint, treed linear model

1. Introduction

Consider the regression model for $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^p$ and $y \in \mathbb{R}$,

$$y = f_0(\mathbf{x}) + \varepsilon,$$

where $f_0 : \mathbb{R}^p \to \mathbb{R}$ and ε is a mean 0 random variable. In this paper, we study the situation where f_0 is convex. That is,

$$\lambda f_0(\mathbf{x}_1) + (1-\lambda) f_0(\mathbf{x}_2) \geq f_0(\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2),$$

for every $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$ and $\lambda \in (0,1)$. Given the observations $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, we would like to estimate f_0 subject to the convexity constraint. Convex regression is easily extended to concave regression since a concave function is the negative of a convex function.

Convex regression problems occur in a variety of settings. Economic theory often dictates that demand (Varian, 1982), production (Varian, 1984; Allon et al., 2007) and consumer preference (Boyd and Vandenberghe, 2004) functions are concave. In financial engineering, stock option prices often have convexity restrictions (Aït-Sahalia and Duarte, 2003). Stochastic optimization

problems in operations research and reinforcement learning can be solved with response surfaces (Lim, 2010) or value-to-go functions. These exhibit concavity in many settings, like resource allocation (Topaloglu and Powell, 2003; Powell, 2007; Toriello et al., 2010) or stochastic control (Keshavarz et al., 2011). Similarly, efficient frontier methods like data envelopment analysis (Kuosmanen and Johnson, 2010) include convexity constraints. In density estimation, shape restrictions like log-concavity provide flexible estimators without tunable parameters (Cule et al., 2010; Cule and Samworth, 2010; Schuhmacher and Dümbgen, 2010). Finally, in optimization, convex approximations to polynomial constraints are valuable for geometric programming (Kim et al., 2004; Boyd et al., 2007; Magnani and Boyd, 2009).

Although convex regression has been well explored in the univariate setting, the literature remains underdeveloped in the multivariate setting. Methods where an objective function is constrained to the set of convex functions through supporting hyperplane constraints for each pair of observations (Hildreth, 1954; Holloway, 1979; Kuosmanen, 2008; Seijo and Sen, 2011; Lim and Glynn, 2012; Allon et al., 2007) or semidefinite constraints over all observations (Roy et al., 2007; Aguilera and Morin, 2008, 2009; Henderson and Parmeter, 2009; Wang and Ni, 2012) are too computationally demanding for more than a few thousand observations.

In more recent approaches, different methods have been developed. Fitting a convex hull to a smoothed version of the data (Aguilera et al., 2011) scales to larger data sets, but is inefficient for more than 4 or 5 dimensions. Refitting a series of hyperplanes can be done in a frequentist (Magnani and Boyd, 2009) or Bayesian (Hannah and Dunson, 2011) manner. While the Bayesian method does not scale to more than a few thousand observations, the frequentist method scales to much larger data sets but can exhibit unstable behavior. Recent literature is more fully reviewed in Section 2.

In this paper, we introduce the first computationally efficient and theoretically sound multivariate convex regression method: convex adaptive partitioning (CAP). It fits a series of hyperplanes to the data through adaptive partitioning. It relies on an alternate, first-order definition of convexity,

$$f_0(\mathbf{x}_1) \geq f_0(\mathbf{x}_2) + g_0(\mathbf{x}_2)^T(\mathbf{x}_1 - \mathbf{x}_2), \tag{1}$$

for every $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$, where $g_0(\mathbf{x}) \in \partial f_0(\mathbf{x})$ is a subgradient of f_0 at $\mathbf{x}$. Equation (1) states that a convex function lies above all of its supporting hyperplanes, or subgradients tangent to f_0. Moreover, with enough supporting hyperplanes, f_0 can be approximately reconstructed by taking the maximum over those hyperplanes.

The CAP estimator is formed by adaptively partitioning a set of observations in a method similar to trees with linear leaves (Chaudhuri et al., 1994). Within each subset of the partition, we fit a linear model to approximate the subgradient of f_0 within that subset. Given a partition with K subsets and linear models, $(\alpha_k, \beta_k)_{k=1}^K$, a continuous, convex (concave) function is then generated by taking the maximum (minimum) over the hyperplanes by

$$f_n(\mathbf{x}) = \max_{k \in \{1,\ldots,K\}} \alpha_k + \beta_k^T \mathbf{x}.$$

The partition is refined by a twofold strategy. First, one of the subsets is split along a cardinal direction (say, x_1 or x_3) to grow K. Then, the hyperplanes themselves are used to refit the subsets. A piecewise linear function like f_n induces a partition; a subset is defined as the region where a particular hyperplane is dominant. The refitting step places the hyperplanes in closer alignment with the observations that generated them. This procedure is repeated until all subsets have a minimal number of observations. The CAP estimator is then created by selecting the value of K that balances

fit with complexity using a generalized cross validation method (Golub et al., 1979; Friedman, 1991). We show that CAP is consistent with respect to the ℓ_∞ metric. Because of the dramatic reduction in runtime, CAP opens a new class of problems for study, namely moderate to large problems with convexity or concavity constraints.

2. Literature Review

The literature for convex regression is scattered throughout a variety of fields, including statistics, operations research, economics numerical analysis and electrical engineering. Most methods are designed for the univariate setting, which is closely related to isotonic regression. Univariate methods rely on the ordering implicit to the real line. Setting $x_{i-1} < x_i < x_{i+1}$ for $i = 2, \ldots, n-1$,

$$\frac{f_0(x_i) - f_0(x_{i-1})}{x_i - x_{i-1}} \leq \frac{f_0(x_{i+1}) - f_0(x_i)}{x_{i+1} - x_i}, \qquad i = 2, \ldots, n-1, \tag{2}$$

is equivalent to Equation (1). When f_0 is differentiable, Equation (2) is equivalent to an increasing derivative function.

The oldest and simplest solution method is the least squares estimator (LSE), which produces a piecewise linear estimator by solving a quadratic program with a least squares objective function subject to the constraints in Equation (2) (Hildreth, 1954; Dent, 1973). Although the LSE is completely free of tunable parameters, the estimator is not smooth and can overfit in boundary regions. Consistency, rate of convergence, and asymptotic distribution were shown by Hanson and Pledger (1976), Mammen (1991) and Groeneboom et al. (2001), respectively. Algorithmic methods for solving the quadratic program were given in Wu (1982); Dykstra (1983) and Fraser and Massam (1989).

Splines use linear combinations of basis functions to produce a smooth estimator; in univariate convex regression, an increasing function can be fit to the derivative of the original function. Meyer (2008) and Meyer et al. (2011) used convex-restricted splines with positive parameters in frequentist and Bayesian settings, respectively. Turlach (2005) and Shively et al. (2011) used unrestricted splines with restricted parameters in frequentist and Bayesian settings, respectively. In other methods, Birke and Dette (2007) used convexity constrained kernel regression. Chang et al. (2007) used a random Bernstein polynomial prior with constrained parameters. Due to the constraint on the derivative of f_0, univariate convex regression is quite similar to univariate isotonic regression; see Brunk (1955), Hall and Huang (2001), Neelon and Dunson (2004) and Shively et al. (2009) for examples.

In the multivariate setting Equation (1) cannot be reduced to a set of $n-1$ linear inequalities. Instead, it needs to hold for every pair of points. The multivariate least squares estimator Hildreth (1954); Holloway (1979) solves the quadratic program,

$$\begin{aligned} &\min \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \\ &\text{subject to } \hat{y}_j \geq \hat{y}_i + \mathbf{g}_i^T (\mathbf{x}_j - \mathbf{x}_i), \quad i, j = 1, \ldots, n. \end{aligned} \tag{3}$$

Here, $\hat{y}_i$ and $\mathbf{g}_i$ are the estimated values of $f_0(\mathbf{x}_i)$ and the subgradient of f_0 at $\mathbf{x}_i$, respectively. The estimator f_n^{LSE} is piecewise linear,

$$f_n^{LSE}(\mathbf{x}) = \max_{i \in \{1,\ldots,n\}} \hat{y}_i + \mathbf{g}_i^T (\mathbf{x} - \mathbf{x}_i).$$

The characterization (Kuosmanen, 2008) and consistency (Seijo and Sen, 2011; Lim and Glynn, 2012) of the least squares problem have only recently been studied. The LSE quickly becomes impractical due to its size: Equation (3) has $n(n-1)$ constraints. This results in a computational complexity of $O((p+1)^4 n^5)$ (Monteiro and Adler, 1989), which becomes impractical after one to two thousand observations. It can also severely overfit in boundary regions. In similar approach, Allon et al. (2007) proposed a method based on reformulating the maximum likelihood problem as one minimizing entropic distance, again subject to n^2 linear constraints generated by the dual problem.

An alternative to first order constraints in Equation (1) is second order, or Hessian, constraints. Roy et al. (2007) and Aguilera and Morin (2008, 2009) solved a math program with a least squares objective function and semidefinite constraints through semidefinite programming. Henderson and Parmeter (2009) used kernel smoothing with a restricted Hessian and found a solution with sequential quadratic programming. While these methods are consistent in some cases (Aguilera and Morin, 2008, 2009), they are computationally infeasible for more than about a thousand observations.

Recently, multivariate convex regression methods have been proposed with different approaches. Aguilera et al. (2011) proposed a two step smoothing and fitting process. First, the data were smoothed and functional estimates were generated over an ε-net over the domain. Then the convex hull of the smoothed estimate was used as a convex estimator. Again, although this method is consistent, it is sensitive to the choice of smoothing parameter and does not scale to more than a few dimensions. Hannah and Dunson (2011) proposed a Bayesian model that placed a prior over the set of all piecewise linear models. They were able to show adaptive rates of convergence, but the inference algorithm did not scale to more than a few thousand observations. Koushanfar et al. (2010) transformed the ordering problem associated with shape constrained inference into a combinatorial optimization problem which was solved with dynamic programming; this scales to a few hundred observations.

The work that is closest to CAP is an iterative fitting scheme of Magnani and Boyd (2009). In this method, the data were divided into K random subsets and a linear model was fit within each subset; a convex function was generated by taking the maximum over these hyperplanes. This new function induced a partition over the covariate space, which generated a new collection of K subsets. Again, linear models were fitted and another convex function was produced by taking the maximum over the new hyperplanes. This sequence was repeated until convergence. Although this method usually produces a high quality estimate, it does not always converge and can be unstable.

3. Convex Adaptive Partitioning

A natural way to model a convex function f_0 is through the maximum of a set of K hyperplanes. We do this by partitioning the covariate space and approximating the gradients within each region by hyperplanes generated by the least squares estimator. The covariate space partition and K are chosen through adaptive partitioning. Given a partition $\{A_1,\ldots,A_K\}$ of $\mathcal{X}$, an estimate of the gradient for each subset can be created by taking the least squares linear estimate based on all of the observations within that region,

$$(\alpha_k,\beta_k) = \arg\min_{\alpha,\beta} \sum_{i:\mathbf{x}_i\in A_k} \left(y_i - \alpha - \beta^T \mathbf{x}_i\right)^2.$$

A convex function $\hat{f}$ can be created by taking the maximum over $(\alpha_k, \beta_k)_{k=1}^K$,

$$\hat{f}_n(\mathbf{x}) = \max_{k \in \{1,\ldots,K\}} \alpha_k + \beta_k^T \mathbf{x}.$$

Adaptive partitioning models with linear leaves have been proposed before; see Chaudhuri et al. (1994), Chaudhuri et al. (1995), Alexander and Grimshaw (1996), Nobel (1996), Dobra and Gehrke (2002), Györfi et al. (2002) and Potts and Sammut (2005) for examples. In most of these cases, the partition is created by adaptively refining an existing partition by dyadic splitting of one subset along one dimension. That is, all data is initially placed within a single subset, which is then split into two new subsets along a single dimension, for example at $x_1 = 5$. The split dimension and value is chosen in a way that minimizes local error within the subset, through impurity (Chaudhuri et al., 1994) or mean squared error minimization (Alexander and Grimshaw, 1996). The SUPPORT algorithm of Chaudhuri et al. (1994) computes test statistics for the difference between the means and variances of the residuals and selects the split with the smallest associated $p-$value. Splitting is continued within a subset until a terminal level of purity or a minimal number of observations is reached in that subset; however, SUPPORT uses a cross-validation based method as a stopping rule. Once a full tree has been created, it is pruned using a variety of cross-validation based methods that aim to remove individual leaves or branches to produce the most simple tree that represents the data well; see Breiman et al. (1984) and Quinlan (1993) for pruning methods.

There are two problems that arise when a piecewise linear additive function,

$$f^*(\mathbf{x}) = \sum_{k=1}^{K} \left(\alpha_k + \beta_k^T \mathbf{x}\right) \mathbf{1}_{\{\mathbf{x} \in A_k\}},$$

is changed into a piecewise linear maximization function, like $\hat{f}$. First, a split that minimizes local error does not necessarily minimize global error for $\hat{f}$. This is easily remedied by selecting splits based on minimizing global error. The second problem is more difficult: the linear models often act in areas over which they were not estimated.

The piecewise linear max function, f_n, generates a new partition, $\{A'_1, \ldots, A'_K\}$, by

$$A'_k = \left\{\mathbf{x} \in \mathcal{X} : \alpha_k + \beta_k^T \mathbf{x} > \alpha_j + \beta_j^T \mathbf{x}, \forall\ j \neq k\right\}.$$

The partition $\{A_1, \ldots, A_K\}$ is not necessarily the same as $\{A'_1, \ldots, A'_K\}$. We can use this new partition to refit the hyperplanes and produce a significantly better estimate. A graphical representation is given in Figure 1.

Refitting hyperplanes in this manner can be viewed as a Gauss-Newton method for the non-linear least squares problem (Magnani and Boyd, 2009),

$$\text{minimize} \sum_{i=1}^{n} \left(y_i - \max_{k \in \{1,\ldots,K\}} \left(\alpha_k + \beta_k^T \mathbf{x}_i\right) \right)^2.$$

Similar methods for refitting hyperplanes have been proposed in Breiman (1993) and Magnani and Boyd (2009). However, repeated refitting may not converge to a stationary partition and is sensitive to the initial partition.

Convex adaptive partitioning uses adaptive partitioning with linear leaves to fit a convex function that is defined as the maximum over the set of leaves. The adaptive partitioning itself differs from

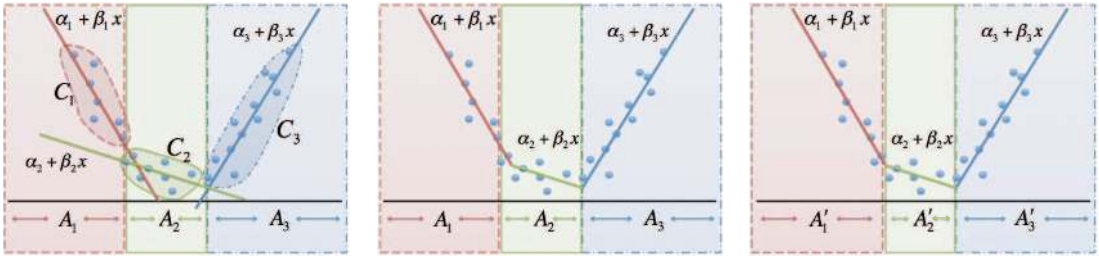

Figure 1: The original space partition A and accompanying data partition C with hyperplanes fit according to that partition (left), the convex estimator based on those hyperplanes; some points are not represented by the hyperplane they were used to fit (center), and subsets refit based on the hyperplanes (right).

previous methods in order to fit piecewise linear maximization functions. Partitions are refined in two steps. First, candidate splits are generated through dyadic splits of existing partitions. These are evaluated and the one that minimizes global error is greedily selected. Second, the new partition is then refit. Although simple, these rules, and refitting in particular, produce large gains over naive adaptive partitioning methods; empirical results are discussed in Section 6.

Most other adaptive partitioning methods use backfitting or pruning to select the tree or partition size. Due to the construction of the CAP estimator, we cannot locally prune and so instead we rely on model selection criteria. We derive a generalized cross-validation method for this setting that is used to select K. This is discussed in Section 5.

3.1 The Algorithm

We now introduce some notation required for convex adaptive partitioning. When presented with data, a partition can be defined over the covariate space (denoted by $\{A_1,\ldots,A_K\}$, with $A_k \subseteq \mathcal{X}$) or over the observation space (denoted by $\{C_1,\ldots,C_K\}$, with $C_k \subseteq \{1,\ldots,n\}$). The observation partition is defined from the covariate partition,

$$C_k = \{i : \mathbf{x}_i \in A_k\}, \quad k = 1,\ldots,K.$$

The relationship between these is shown in Figure 1. CAP proposes and searches over a set of models, $M_1,\ldots,M_K$. A model M_k is defined by: 1) the covariate partition $\{A_1,\ldots,A_K\}$, 2) the corresponding observation partition, $\{C_1,\ldots,C_K\}$, and 3) the hyperplanes $(\alpha_j,\beta_j)_{j=1}^K$ fit to those partitions.

The CAP algorithm progressively refines the partition until each subset cannot be split without one subset having fewer than a minimal number of observations, n_{min}. This value is chosen to balance increasing model complexity against accurate local model fit and computational complexity. When a relatively small number of observations is used to fit local linear models, the local models tend to fit noise. This is particularly problematic with linear models, which can predict extreme values based on overfit models. The issue is aggravated when the estimator is defined as a max over local linear models, which can be dominated by a few extreme values; it can cause instability in the estimator of Magnani and Boyd (2009). Therefore, we choose a conservative value for n_{min}, which

admits logarithmic partition growth,

$$n_{min} = \min\left\{\frac{n}{D\log(n)}, 2(d+1)\right\}.$$

Here D is a log scaling factor, which acts to change the base of the log operator. We briefly outline the CAP algorithm below.

3.1.1 CONVEX ADAPTIVE PARTITIONING (CAP)

1. **Initialize.** Set $K = 1$; place all observations into a single observation subset, $C_1 = \{1, \ldots, n\}$; $A_1 = \mathcal{X}$; this defines model M_1.

2. **Split.** Refine partition by splitting a subset.

   a. *Generate candidate splits.* Generate candidate model $\hat{M}_{kj\ell}$ by 1) fixing a subset k, 2) fixing a dimension j, 3) dyadically dividing the data in subset k and dimensions j according to knot a_ℓ. This is done for L knots, all p dimensions and K subsets.

   b. *Select split.* Choose the model M_{K+1} from the candidates that minimizes global mean squared error on the training set and satisfies $\min_k |C_k| \geq n_{min}$. Set $K = K + 1$.

3. **Refit.** Use the partition induced by the hyperplanes to generate model M'_K. Set $M_K = M'_K$ if for every subset C'_k in M'_K, $|C'_k| \geq n_{min}$.

4. **Stopping conditions.** If for every subset C_k in M_k, $|C_k| < 2n_{min}$, stop fitting and proceed to step 5. Otherwise, go to step 2.

5. **Select model size.** Each model M_k creates an estimator,

$$f_k(\mathbf{x}) = \max_{j \in \{1,\ldots,k\}} \alpha_j + \beta_j^T \mathbf{x}.$$

   Use generalized cross-validation on the estimators to select final model M^* from $\{M_k\}_{k=1}^K$.

3.2 Splitting Rules

To split, we create a collection of candidate models by splitting a single subset into two subsets. We create models for every subset and search along every cardinal direction by splitting the data along that direction. For a fixed dimension j and subset k, let x_{min}^{jk} be the minimum value and x_{max}^{jk} be the maximum value of the covariates in this subset and dimension. Let $0 < a_1 < \cdots < a_L < 1$ be a set of evenly spaced knots that represent the proportion between x_{min}^{jk} and x_{max}^{jk}.

We create model $\hat{M}_{jk\ell}$ by 1) fixing subset $k \in \{1, \ldots, K\}$, and 2) fixing dimension $j \in \{1, \ldots, p\}$.

$$x_{min}^{jk} = \min\{x_{ij} : i \in C_k\}, \qquad x_{max}^{jk} = \max\{x_{ij} : i \in C_k\}.$$

Use the weighted average $b_{jk\ell} = a_\ell x_{min}^{jk} + (1 - a_\ell) x_{max}^{jk}$ to split C_k and A_k in dimension j. Set

$$C'_k = \{i : i \in C_k, x_{ij} \leq b_{jk\ell}\}, \qquad C'_{K+1} = \{i : i \in C_k, x_{ij} > b_{jk\ell}\},$$
$$A'_k = \{\mathbf{x} : \mathbf{x} \in A_k, x_j \leq b_{jk\ell}\}, \qquad A'_{K+1} = \{\mathbf{x} : \mathbf{x} \in A_k, x_j > b_{jk\ell}\}.$$

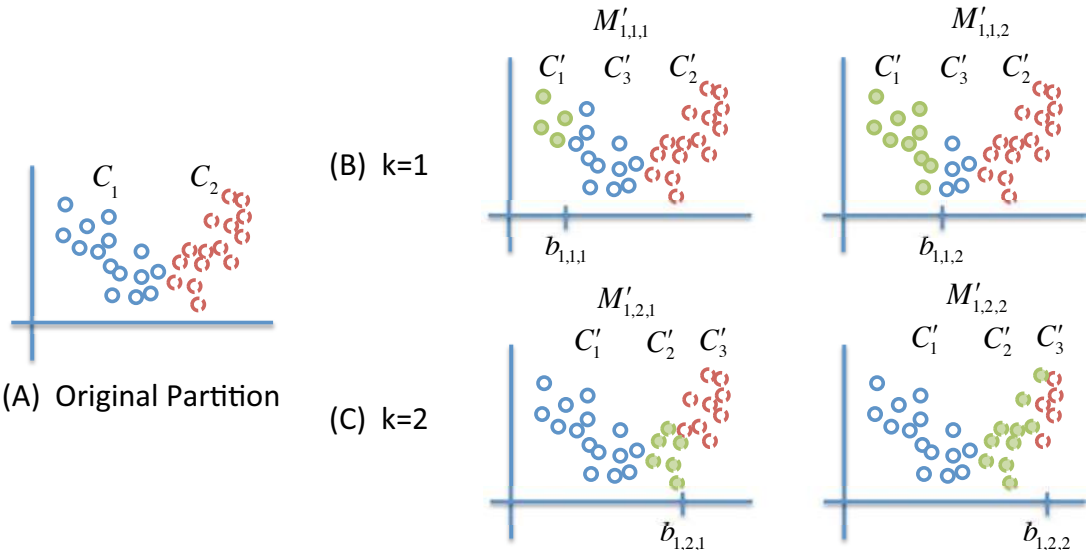

Figure 2: (A) The original observation partition C for M_2, (B) new splits generated from the subset C_1, and (C) new splits generated from the subset C_2. Since there is only one dimension, we fix $j = 1$.

These define new subset and covariate partitions, $C'_{1:K+1}$ and $A'_{1:K+1}$ where $C'_{k'} = C_{k'}$ and $C'_{k'} = C_{k'}$ for $k' \neq k$. See Figure 2 for an example. Fit hyperplanes $(\hat{\alpha}_k, \hat{\beta}_k)_{k=1}^{K+1}$ in each of the subsets. The triplet of observation partition $C'_{1:K+1}$, covariate partition, $A'_{1:K+1}$, and set of hyperplanes $(\hat{\alpha}_k, \hat{\beta}_k)_{k=1}^{K+1}$ defines the model $M'_{jk\ell}$. This is done for $k = 1, \ldots, K$, $j = 1, \ldots, p$ and $\ell = 1, \ldots, L$. After all models are generated, set $K = K + 1$.

We note that any models where $\min_k |C'_k| < n_{min}$ are discarded. If all models are discarded in one subset/dimension pair, we produce a model by splitting on the subset median in that dimension.

3.3 Split Selection

We select the model $M'_{jk\ell}$ that gives the smallest *global* error. Let $(\alpha_i^{jk\ell}, \beta_i^{jk\ell})_{i=1}^{K}$ be the hyperplanes associated with $M'_{jk\ell}$ and let

$$\hat{f}^{jk\ell}(\mathbf{x}) = \max_{i \in \{1,\ldots,K\}} \alpha_i^{jk\ell} + \beta_i^{jk\ell T} \mathbf{x}$$

be its estimator. We set the model M_K to be the one that minimizes global mean squared error,

$$M_K = \left\{ \hat{M}_{jk\ell} : (j,k,\ell) = \arg\min_{j,k,\ell} \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \hat{f}^{jk\ell}(\mathbf{x}_i) \right)^2 \right\}.$$

Set $\hat{f}_K$ to be the minimal estimator. We note that M_K may not be unique, however this seldom occurs in practice.

3.4 Refitting

We refit by using the partition induced by the hyperplanes. Let $(\alpha_{1:K}, \beta_{1:K})$ be the hyperplanes associated with M_K. Refit the partitions by

$$C'_k = \{\mathbf{x}_i : \alpha_k + \beta_k^T \mathbf{x}_i \geq \alpha_j + \beta_j^T \mathbf{x}_i, j \neq k\}$$

for $k = 1, \ldots, K$. The covariate partition, $A'_{1:K}$ is defined in a similar manner. Fit hyperplanes in each of those subsets. Let M'_K be the model generated by the partition $C'_1, \ldots, C'_K$. Set $M_K = M'_K$ if $|C'_k| \geq n_{min}$ for all k.

3.5 Stopping Criteria

Stopping criteria are similar to those in tree-based models (Nobel, 1996; Györfi et al., 2002). That is, the model stops when there are not enough observations within each subset of leaf to generate any further candidate splits,

$$|C_k| \leq 2n_{min}$$

for $k = 1, \ldots, K$. After fitting to termination the final model size, however, is chosen through a pruning method discussed Section 5.

3.6 Tunable Parameters

CAP has two tunable parameters, L and n_{min}. L specifies the number of knots used when generating candidate models for a split. Its value is tied to the smoothness of f_0 and after a certain value, usually 5 to 10 for most functions, higher values of L offer little fitting gain.

We choose a minimal subset size, n_{min}, that admits at most $O(\log(n))$subsets. A parameter D is used to specify a minimum subset size, $n_{min} = n/(D\log(n))$. Here D transforms the base of the logarithm from e into $\exp(1/D)$. We have found that $D = 3$ (implying base ≈ 1.4) is a good choice for most problems.

Increases in either of these parameters increase the computational time. Sensitivity to these parameters, both in terms of predictive error and computational time, is empirically examined in Appendix B.

3.7 Computational Efficiency

Each round of CAP requires $O(dKL)$ regressions to be fit for model proposal. Since observations are moved from one side of a threshold to another within each leaf, an efficient method is to maintain and update parameters and the sum of squares and cross products within each leaf. Alternately, a *QR* decomposition may be maintained and updated for each leaf (Alexander and Grimshaw, 1996). Unlike treed linear models, all linear models need to be refit for each round of CAP.

4. Consistency

Consistency for CAP can be shown in a related manner to consistency for other adaptive partitioning models, like CART (Breiman et al., 1984), treed linear models (Chaudhuri et al., 1994) and other variants (Nobel, 1996; Györfi et al., 2002). We take a two-step approach, first showing consistency for the mean function and first derivatives of a more traditional treed linear model based on CAP under the ℓ_∞ metric and then we use that to show consistency for the CAP estimator itself.

Letting M_n^* be the model for the CAP estimate after n observations, define the discontinuous piecewise linear estimate based on M_n^*,

$$f_n^*(\mathbf{x}) = \sum_{k=1}^{K_n} \left(\alpha_k + \beta_k^T \mathbf{x}\right) \mathbf{1}_{\{\mathbf{x} \in A_k\}},$$

where K_n is the partition size, $A_1, \ldots, A_{K_n}$ are the covariate partitions and $(\alpha_k, \beta_k)_{k=1}^{K_n}$ are the hyperplanes associated with M_n^*. Let $f_n(\mathbf{x})$ be the CAP estimator based on M_n^*,

$$f_n(\mathbf{x}) = \max_{k \in \{1,\ldots,K_n\}} \alpha_k + \beta_k^T \mathbf{x}.$$

Each subset A_k has an associated diameter, $d_{nk} = \sup_{\mathbf{x}_1, \mathbf{x}_2 \in A_k} ||x_1 - x_2||_2$. Define the empirical covariate mean for subset k as $\bar{x}_k = \frac{1}{|C_k|} \sum_{i \in C_k} \mathbf{x}_i$. For $\mathbf{x}_i \in A_k$, define

$$\Gamma_i = \begin{bmatrix} [1, \ldots, 1] \\ d_{nk}^{-1} \left(\mathbf{x}_i - \bar{\mathbf{x}}_k\right) \end{bmatrix}, \qquad G_k = \sum_{i \in C_k} \Gamma_i \Gamma_i^T.$$

Note that $(\alpha_k, \beta_k) = G_k^{-1} \sum_{i \in C_k} \Gamma_i y_i$ whenever G_k is nonsingular.

Let $\mathbf{x}_1, \ldots, \mathbf{x}_n$ be i.i.d. random variables. We make the following assumptions:

A1. $\mathcal{X}$ is compact and f_0 is Lipschitz continuous and continuously differentiable on $\mathcal{X}$ with Lipschitz parameter ζ.

A2. There is an $a > 0$ such that $\mathbb{E}\left[e^{a|Y - f_0(x)|} \,|\, \mathbf{X} = \mathbf{x}\right]$ is bounded on $\mathcal{X}$.

A3. Let λ_k be the smallest eigenvalue of $|C_k|^{-1} G_k$ and $\lambda_n = \min_k \lambda_k$. Then λ_n remains bounded away from 0 in probability as $n \to \infty$.

A4. The diameter of the partition $\max_k d_{nk}^{-1} \to 0$ in probability as $n \to \infty$.

A5. The number of observations in each subset satisfies $\min_{k=1,\ldots,K_n} |C_k| > d_{nk}^{-1} \sqrt{n \log(n)}$ in probability as $n \to \infty$.

Assumptions **A1.** and **A2.** place regularity conditions on f_0 and the noise distribution, respectively. Assumption **A3.** is a regularity condition on the covariate distribution to ensure the uniqueness of the linear estimates. Assumption **A4.** is a condition that can be included in the algorithm and checked along with the subset cardinality, $|C_k|$. If $\mathcal{X}$ is given, it can be computed directly, otherwise it can be approximated using $\{\mathbf{x}_i : i \in C_k\}$. Assumption **A5.** ensures that there are enough observations in the terminal nodes to fit the linear models.

To show consistency of f_n under the ℓ_∞ metric, we first show consistency of f_n^* and its derivatives under the ℓ_∞ metric in Theorem 1. This is similar to Theorem 1 of Chaudhuri et al. (1994) for treed linear models, although we need to modify it to allow partitions with an arbitrarily large number of faces.

Theorem 1 *Suppose that assumptions* **A1.** *through* **A5.** *hold. Then,*

$$\max_{k=1,\ldots,K_n} \sup_{\mathbf{x} \in A_k} \left|\alpha_k + \beta_k^T \mathbf{x} - f_0(\mathbf{x})\right| \to 0, \qquad \max_{k=1,\ldots,K_n} \sup_{\mathbf{x} \in A_k} ||\beta_k - \nabla f_0(\mathbf{x})||_\infty \to 0$$

in probability as $n \to \infty$.

The CAP algorithm is similar to the SUPPORT algorithm of Chaudhuri et al. (1994), except the refitting step of CAP allows partition subsets to be polyhedra with up to K_n faces. Theorem 1 is analogous to Theorem 1 of Chaudhuri et al. (1994); to prove our theorem, we modify parts of the proof in Chaudhuri et al. (1994) that rely on a fixed number of polyhedral faces. The proof is given in Appendix A.

Using the results from Theorem 1, extension to consistency for f_n under the ℓ_∞ metric is fairly simple; this is given in Theorem 2.

Theorem 2 *Suppose that assumptions* **A1.** *through* **A5.** *hold. Then,*

$$\sup_{\mathbf{x}\in\mathcal{X}} |f_n(\mathbf{x}) - f_0(\mathbf{x})| \to 0$$

in probability as $n \to \infty$.

The proof follows immediately from Theorem 1 and some algebra. Details are given in the Appendix A.

5. Generalized Cross-Validation

The terminal model produced by CAP can overfit the data. As a fast approximation to leave-one-out cross-validation, we use generalized cross-validation (GCV) (Golub et al., 1979; Friedman, 1991) to select the best model from all of those produced by CAP, $M_1,\ldots,M_K$. A given model M_K is generated by a collection of K linear models. In linear regression, GCV relies on the following approximation

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f_{-i}(\mathbf{x}_i))^2 = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - f_n(\mathbf{x}_i)}{1-H_{ii}}\right)^2 \approx \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - f_n(\mathbf{x}_i)}{1-\mathrm{Tr}(H)}\right)^2, \tag{4}$$

where H_{ii} is the i^{th} diagonal element of the hat matrix, $\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$, $\hat{f}_{-i}$ is the estimator conditioned on all of the data minus element i. We note that $\mathrm{Tr}(H)$ is sometimes approximated by the degrees of freedom divided by the number of observations.

The model M_K is defined by $C_1,\ldots,C_K$, the partition, and the hyperplanes $(\alpha_k,\beta_k)_{k=1}^K$, which were generated by the partition. Let $(\alpha_k^{(-i)},\beta_k^{(-i)})_{k=1}^K$ be the collection of hyperplanes generated when observation i is removed; notice that if $i \in C_k$, only (α_k,β_k) changes. Let $\hat{f}_{-iK}$ be the estimator for model M_K with observation i removed. Using the derivation in Equation (4),

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{f}_{-iK}(\mathbf{x}_i)\right)^2 &= \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \max_{k\in\{1,\ldots,K\}} \alpha_k^{(-i)} + \beta_k^{(-i)^T}\mathbf{x}_i\right)^2, \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \alpha_{k(i)} - \beta_{k(i)}^T\mathbf{x}_i}{1 - H_{ii}^{k(i)}\mathbf{1}_{\{i\in C_{k(i)}\}}}\right)^2 \\
&\approx \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \alpha_{k(i)} - \beta_{k(i)}^T\mathbf{x}_i}{1 - \mathrm{Tr}(H^{k(i)})\mathbf{1}_{\{i\in C_{k(i)}\}}}\right)^2,
\end{aligned} \tag{5}$$

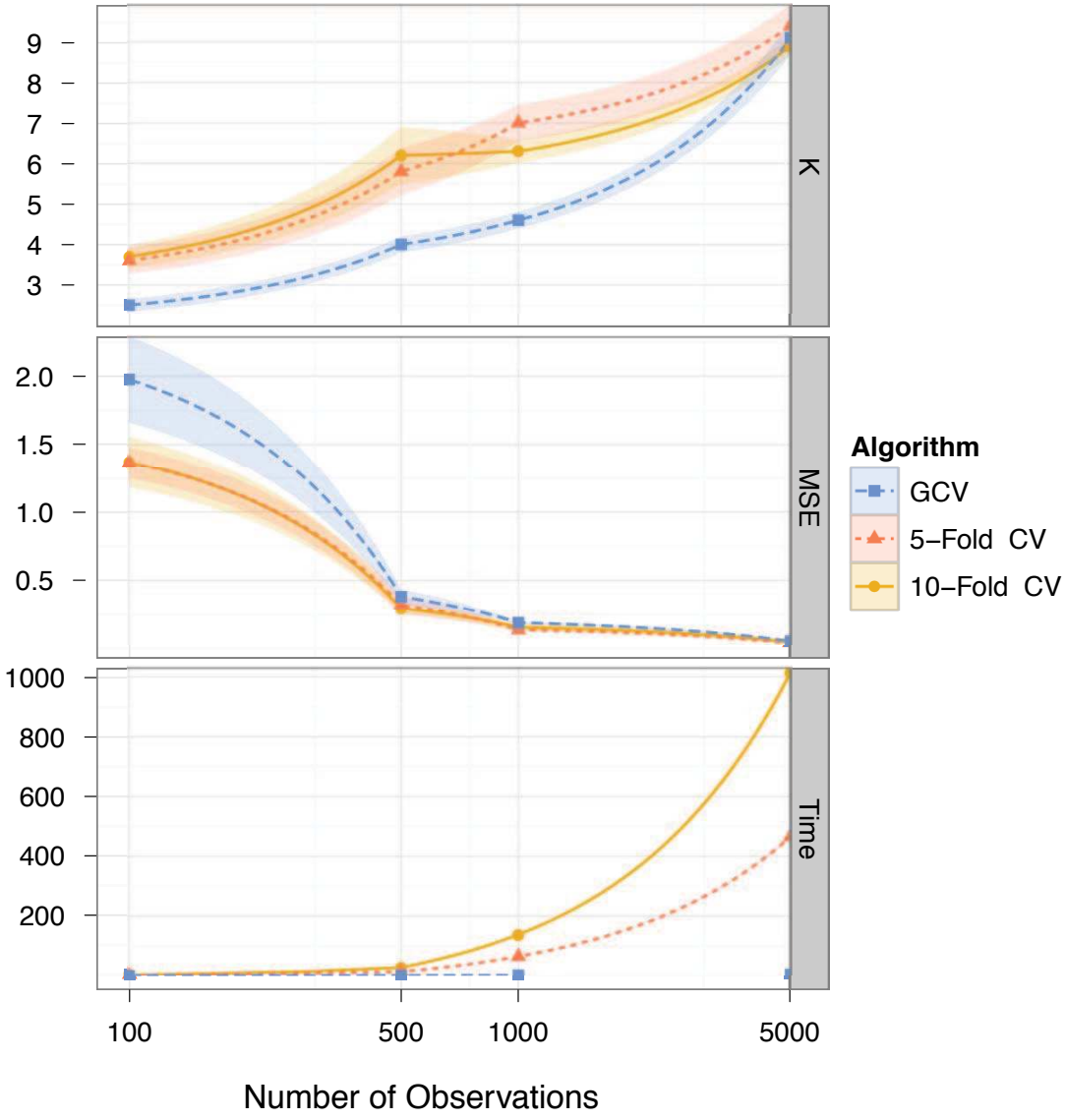

Figure 3: Log-continuous plots for number of observations vs. K (top), MSE (middle), and runtime in seconds (bottom) for GCV, 5-fold and 10-fold cross validation, plus/minus one standard error. Data were generated from 10 i.i.d. training sets with $\mathbf{x} \sim N_5(0, I)$, $y = (x_1 + .5x_2 + x_3)^2 - x_4 + .25x_5^2 + \varepsilon$, and $\varepsilon \sim N(0,1)$.

where, in a slight abuse of notation, H_{ii}^k is the diagonal entry of the hat matrix for subset k corresponding to element i, and

$$k(i) = \arg \max_{k \in \{1,\ldots,K\}} \frac{\alpha_k + \beta_k^T \mathbf{x}_i}{1 - \mathrm{Tr}(H^k)\mathbf{1}_{\{i \in C_k\}}}.$$

To select K, we find the K that minimizes the right hand side of Equation (5). Although more computationally intensive than GCV in linear models, the computational complexity for CAP GCV is similar to that of the CAP split selection step.

We empirically compared GCV selection of K with 5- and 10-fold cross validation selection of K. GCV tends to select a smaller K than full cross validation, particularly on smaller problems. Predictive results, however, are comparable for moderate to large problem sizes ($n \geq 5,000$) while the runtime of GCV is orders of magnitude less than 5- and 10-fold cross validation. We should expect more discrepancy between cross-validation and GCV on smaller problems because GCV relies on an asymptotic approximation. In these cases, full cross validation selection of K may be worthwhile. Representative results are given in Figure 3.

We can use generalized cross-validation to create a more efficient stopping rule for CAP. We note that GCV scores are often unimodal in K. Instead of fully growing the tree, we stop splitting after the score has increased twice in a row. The resulting algorithm is called Fast CAP; details are given in Appendix B.

6. Empirical Analysis

We compare shape constrained and unconstrained regression methods across a set of convex regression problems: two synthetic regression problems, predicting mean weekly wages and value function approximation for pricing basket options.

6.1 Synthetic Regression Problems

We apply CAP to two synthetic regression problems to demonstrate predictive performance and analyze sensitivity to tunable parameters. The first problem has a non-additive structure, high levels of covariate interaction and moderate noise, while the second has a simple univariate structure embedded in a higher dimensional space and low noise. Low noise or noise free problems often occur when a highly complicated convex function needs to be approximated by a simpler one (Magnani and Boyd, 2009).

6.1.1 PROBLEM 1

Here $\mathbf{x} \in \mathbb{R}^5$. Set

$$y = (x_1 + .5x_2 + x_3)^2 - x_4 + .25x_5^2 + \varepsilon,$$

where $\varepsilon \sim N(0,1)$. The covariates are drawn from a 5 dimensional standard Gaussian distribution, $N_5(0,I)$.

6.1.2 PROBLEM 2

Here $\mathbf{x} \in \mathbb{R}^{10}$. Set

$$y = \exp\left(\mathbf{x}^T\mathbf{q}\right) + \varepsilon,$$

where $\mathbf{q}$ was randomly drawn from a Dirichlet(1,...,1) distribution,

$$\mathbf{q} = (0.0680, 0.0160, 0.1707, 0.1513, 0.1790, 0.2097, 0.0548, 0.0337, 0.0377, 0.0791)^T.$$

We set $\varepsilon \sim N(0, 0.1^2)$. The covariates are drawn from a 10 dimensional standard Gaussian distribution, $N_{10}(0,I)$.

6.1.3 PREDICTIVE PERFORMANCE AND RUNTIMES

We compared the performance of CAP and Fast CAP to other regression methods on problems 1 and 2. We implemented the following shape constrained algorithms: the least squares regression (LSE) using `cvx` (Grant and Boyd, 2012, 2008), and the linear refitting algorithm of Magnani and Boyd (2009). The general methods included Gaussian processes (Rasmussen and Williams, 2006) using `gpml` in Matlab, tree regression with constant values in the leaves using `classregtree` in Matlab, multivariate adaptive regression splines (MARS) (Friedman, 1991) using `ARESlab` in Matlab, and support vector machines (SVMs) using the `e1071` package in `R`.

For CAP and Fast CAP, we set the parameters to $D = 3$ and $L = 10$; the sensitivity to these parameters is examined in Appendix C. The parameter K was chosen by GCV for CAP. In Fast CAP, the number of random search directions was set to be $p = \min(d, 10)$. All methods were given a maximum runtime of 90 minutes, after which the results were discarded. Methods were run on 10 random training sets and tested on the same testing set of 10,000 random covariates. Average runtimes and predictive performance are shown in Figure 4.

Non-convex regression methods performed poorly compared to shape restricted methods, particularly in the higher noise setting. Amongst the shape restricted methods, only CAP and Fast CAP had consistently low predictive error. The method of Magnani and Boyd (2009) can become unstable, which is seen in problem 1. Surprisingly, the LSE had high predictive error. This can be attributed to overfitting, particularly in the boundary regions. A demonstration is given in Figure 5. Although CAP and Fast CAP had similar predictive performance, their runtimes often differed by an order of magnitude with the largest differences on the biggest problem sizes. Based on this performance, we would suggest using Fast CAP on larger problems.

We note that the empirical rate of convergence for CAP and Fast CAP is much faster than would be predicted by minimax convergence rates. The results, however, are consistent with rates that adapt to an underlying linear subspace; this is examined in Appendix D.

6.1.4 CAP AND TREED LINEAR MODELS

Treed linear models are a popular method for regression and classification. They can be easily modified to produce a convex regression estimator by taking the maximum over the linear leaves. CAP differs from existing treed linear models in how the partition is refined. First, subset splits are selected based on global reduction of error. Second, the partition is refit after a split is made. To investigate the contributions of each step, we compare to treed linear models generated by: 1) local error reduction as an objective for split selection and no refitting, 2) global error reduction as an objective function for split selection and no refitting, and 3) local error reduction as an objective for split selection along with refitting. All estimators based on treed linear models are generated by taking the maximum over the set of linear models in the leaves. We compared the performance of these methods on problems 1 and 2 over 10 different training sets and a single testing set. Average predictive error is displayed in Figure 6.

Global split selection and refitting are both beneficial, but in different ways. Refitting dramatically reduces predictive error, but can add variance to the estimator in noisy settings. Global split selection modestly reduces predictive error but can reduce variance in noisy settings, like problem 1. The combination of the two produces CAP, which has both low variance and high predictive accuracy.

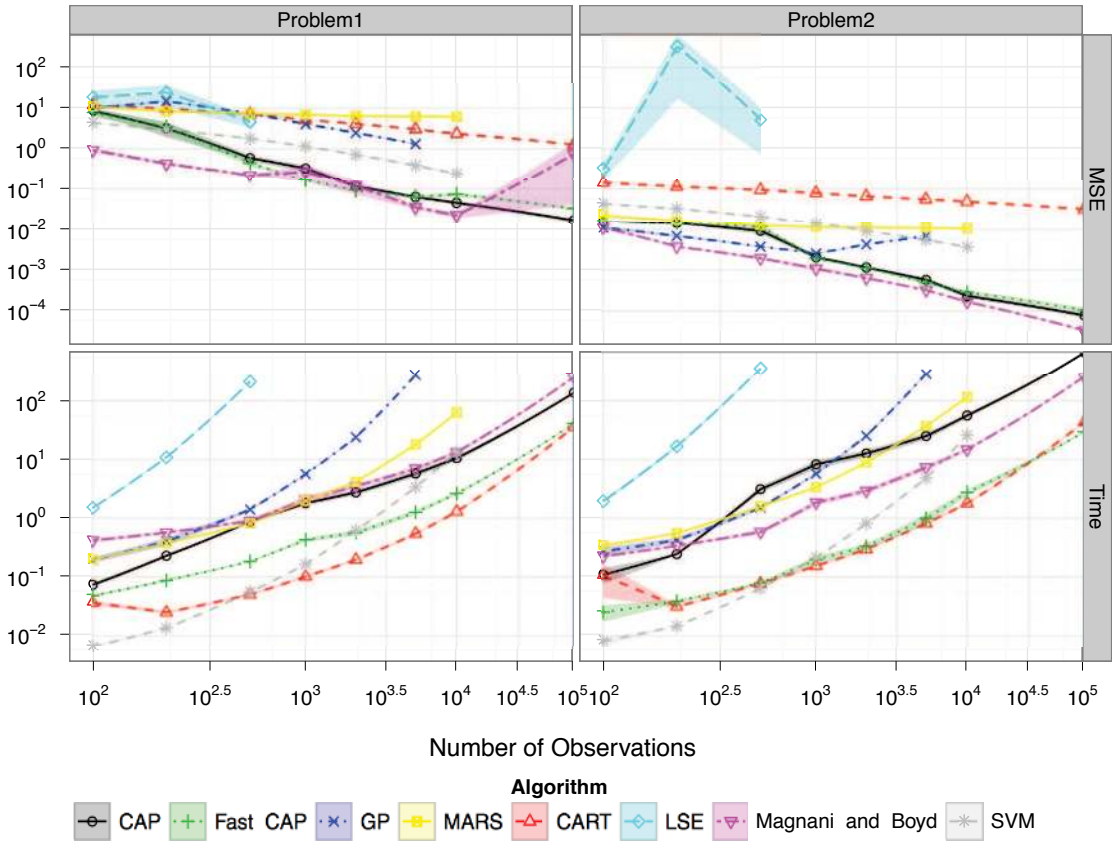

Figure 4: Mean squared error (top) and runtime in seconds (bottom) plus/minus one standard error on problem 1 (left) and problem 2 (right) for CAP, Fast CAP, Gaussian processes, MARS, CART, the least squares estimator, the linear fitting method of Magnani and Boyd (2009) and support vector machines.

6.2 Predicting Weekly Wages

We use shape restricted methods to predict mean weekly wages based on years of education and experience. The data are from the 1988 Current Population Survey (CPS); they originally appeared in Bierens and Ginther (2001) and can be accessed as `ex1029` in the `Sleuth2` package in `R`. The data set contains 25,361 records of weekly wages for full-time, adult, male workers for 1987, along with years experience, years of education, race (either back or white; no others were included in the sample), region, and whether the last job held was part time.

A reasonable assumption for wages is that they are concave in years experience. Each year previously worked should have decreasing returns for average wages until peak earnings are reached, with modest declines afterwards. Indeed, this pattern is seen in Figure 7 when we compare average weekly wages against experience. Wages can also be expected to increase based on education level, but not in a concave or convex fashion. However, concavity can be generated with an exponential transformation of education; this is shown in Figure 7. We therefore used a transformation,

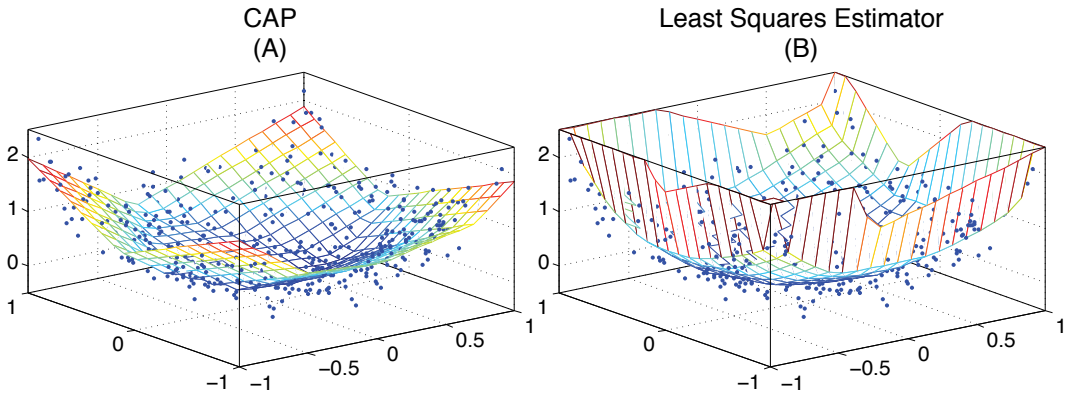

Figure 5: (A) The CAP estimator, and (B) the LSE fit to 500 observations drawn from $y = x_1^2 + x_2^2 + \varepsilon$, where $\varepsilon \sim N(0, 0.25^2)$. The covariates were drawn from a 2 dimensional uniform distribution, Unif$[-1,1]^2$. The LSE was truncated at predicted values of 2.5 for display, although some predicted values reached as high as 4,800 on $[-1,1]^2$.

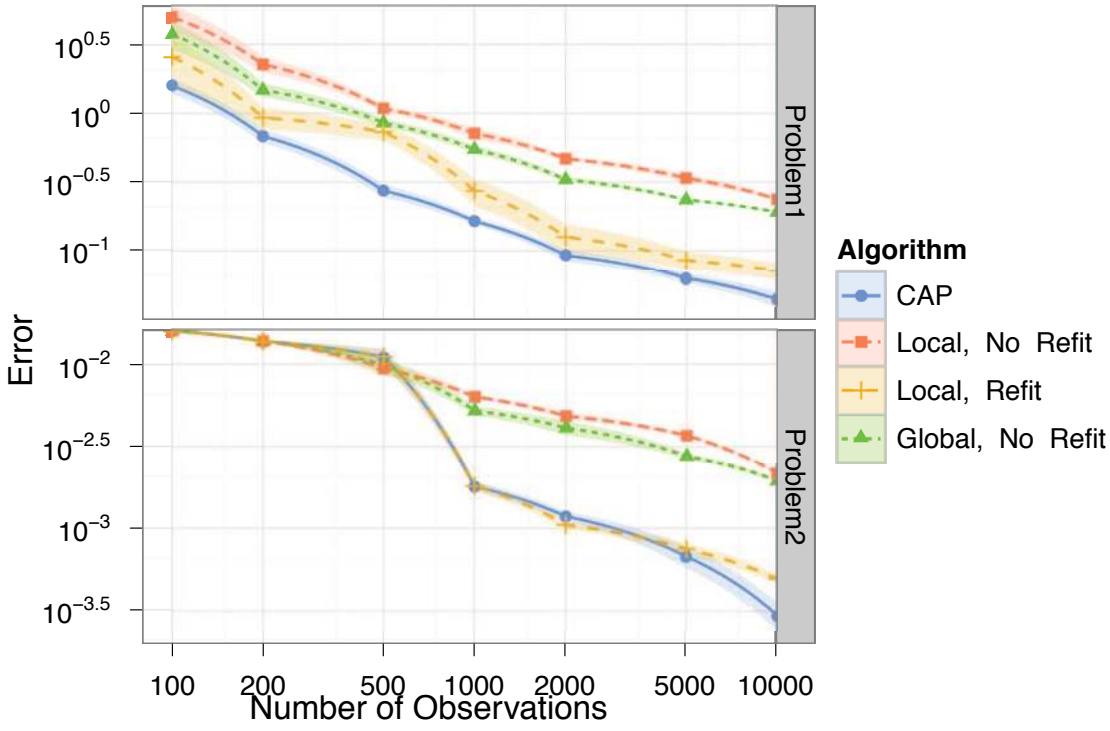

Figure 6: Number of observations (log scale) vs. mean squared error (log scale) plus/minus one standard error for CAP and treed linear models with local split selection, with no refitting; local split selection, with refitting; and global split selection, with no refitting.

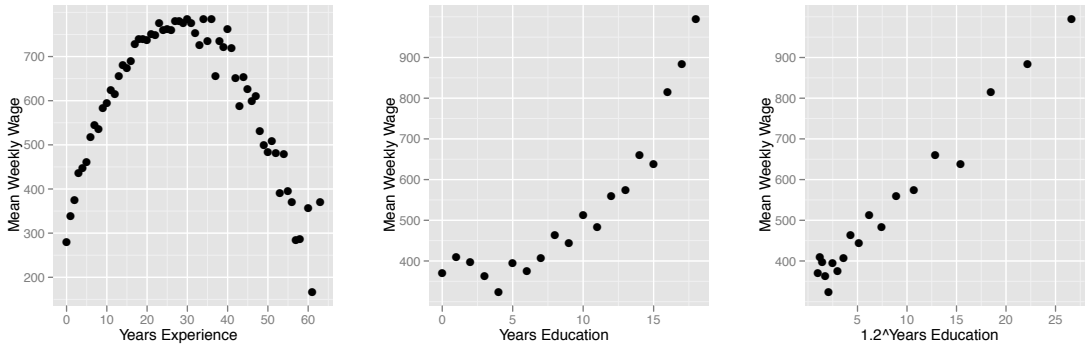

Figure 7: Mean weekly wages vs. years of experience (left), mean weekly wages vs. years of education (center), mean weekly wages vs. $1.2^{\text{years education}}$ (right).

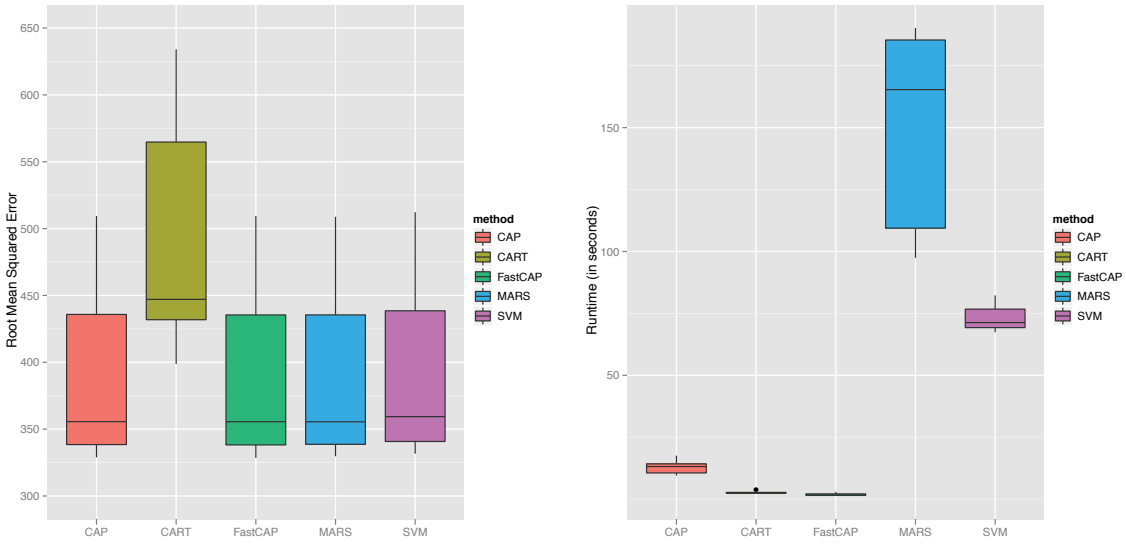

Figure 8: Root mean squared error (RMSE), left, and runtime in seconds, right, for CAP, Fast CAP, CART, MARS, and SVMs for predicting weekly wages based on years experience and years education.

$1.2^{\text{years education}}$, as a covariate. Shape restrictions do not hold with any other covariates, so they are discarded.

We implemented CAP, fast CAP, the linear model of Magnani and Boyd (2009), CART, MARS, and SVMs. Due to the problem size, we did not use Gaussian processes or the least squares estimator. We estimated RMSE through 10-fold cross validation. Results and runtimes are shown in Figure 8 for all methods except Magnani and Boyd (2009). This had a RMSE of 10,156, orders of magnitude larger than other methods, and was hence omitted from the figures.

Method	RMSE	Time
CAP	$\mathbf{385.7 \pm 20.8}$	12.8 ± 0.8
Fast CAP	$\mathbf{385.7 \pm 20.8}$	1.9 ± 0.2
CART	489.6 ± 26.1	2.6 ± 0.2
MARS	385.8 ± 20.7	150.4 ± 12.8
Magnani and Boyd (2009)	10156.0 ± 9765.2	8.0 ± 0.7
SVM	388.9 ± 20.7	72.9 ± 1.6

Table 1: Average RMSE and runtime in seconds, plus/minus one standard error for CAP, Fast CAP, CART, MARS, Magnani and Boyd (2009) and SVMs.

This data set presents difficulties for many methods due to its size ($n > 20,000$) and highly skewed distribution. CAP, Fast CAP, MARS and SVMs all had comparable predictive error rates, while CART produced error rates about 27% higher. The linear fitting method of Magnani and Boyd (2009) occasionally tried to fit outliers with hyperplanes, resulting in about a 2,500% increase in predictive error. This potential instability is one of the largest drawbacks with the method of Magnani and Boyd (2009). In terms of runtimes, Fast CAP and CAP were both significantly faster than any methods that produced comparable results, with runtime reductions of more than 80% over SVMs.

In Figure 9, we compare the predicted functions produced by CAP and SVMs. In areas with small amounts of data, such for people with low education, the SVM produces results that do not match prior information. In the SVM surface, someone with 0 years of experience and 0 years of education is predicted to have about a 150% larger weekly wage than a high school graduate with 0 years of experience and about the same weekly wage as someone with a 4-year college degree and 0 years of experience. By imposing shape constraints, CAP eliminates these types of problems and produces a surface that conforms to prior knowledge.

Unlike the surface produced by SVM regression, the surface produced by CAP is not smooth. A greater degree of smoothness can be added through ensemble methods like bagging (Breiman, 1996) and smearing (Breiman, 2000). Averaging randomized convex estimators produces a new convex estimator; these methods have been explored for approximating objective functions in Hannah and Dunson (2012). A surface produced by smearing CAP is shown on the right in Figure 9. Note that its overall shape is quite similar to the original CAP estimator while most of the sharp edges have been smoothed away.

6.3 Pricing Stock Options

In sequential decision problems, a decision maker takes an action based on a currently observed state of the world based on the current rewards of that action and possible future rewards. Approximate dynamic programming is a modeling method for such problems based on approximating a value-to-go function. Value-to-go functions, or simply "value functions," give the value for each state of the world if all optimal decisions are made subsequently.

Often value functions are known to be convex or concave in the state variable; this is common in options pricing, portfolio optimization and logistics problems. In some situations, such as when a linear program is solved each time period to determine an action, a convex value function *is*

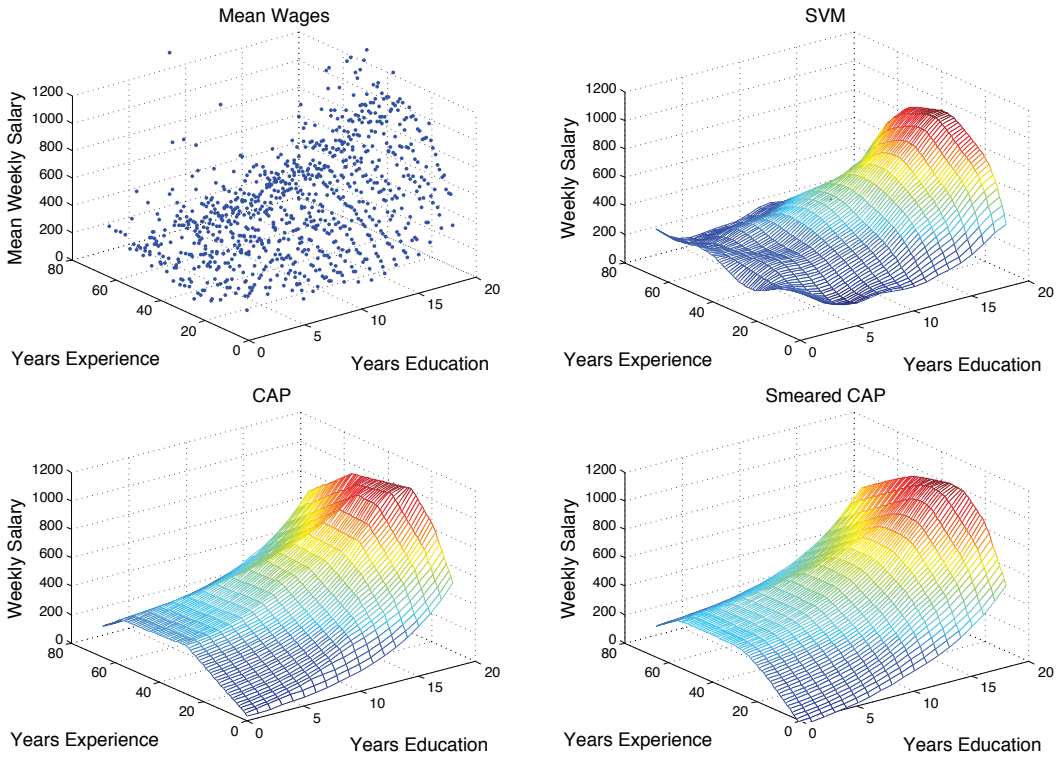

Figure 9: Mean weekly wage based on years of experience and years of education (top left), predicted values using SVM regression (top right), CAP (bottom left), and smeared CAP (bottom right).

required for computational tractability. Convex regression holds great promise for value function approximation in these problems.

To give a simple example for value function approximation, we consider pricing American basket options on the average of M underlying assets. Options give the holder the right—but not the obligation—to buy the underlying asset, in this case the average of M individual assets, for a predetermined strike price R. In an American option, this can be done at any time between the issue date and the maturity date, T. However, American options are notoriously difficult to price, particularly when the underlying asset base is large.

A popular method for pricing American options uses approximate dynamic programming where continuation values are approximated via regression (Carriere, 1996; Tsitsiklis and Van Roy, 1999, 2001; Longstaff and Schwartz, 2001). We summarize these methods as follows; see Glasserman (2004) for a more thorough treatment. The underlying assets are assumed to have the sample path $\{X_1,\ldots,X_T\}$, where $X_t = \{S_1(t),\ldots,S_M(t)\}$ is the set of securities at time t. At each time t, a continuation value function, $\bar{V}_t(X_t)$, is estimated by regressing a value function for the next time period, $\bar{V}_{t+1}(X_{t+1})$, on the current state, X_t. The continuation value is the value of holding the option rather than exercising given the current state of the assets. The value function is defined to

be the max of the current exercise value and the continuation value. Options are exercised when the current exercise value is greater than or equal to the continuation value.

The procedure to estimate the continuation values is as follows (as summarized in Glasserman 2004):

0. Define basket payoff function,

$$h(X_t) = \max\left\{\frac{1}{M}\sum_{k=1}^{M} S_k(t) - R, 0\right\}.$$

1. Sample N independent paths, $\{X_{1j},\ldots,X_{Tj}\}$, $j = 1,\ldots,N$.
2. At time T, set $\bar{V}_T(X_{Tj}) = h(X_{Tj})$.
3. Apply backwards induction: for $t = T-1,\ldots,1$,
   - given $\{\bar{V}_{t+1}(X_{t+1j})\}_{j=1}^N$, regress on $\{X_{tj}\}_{j=1}^N$ to get continuation value estimates $\{\bar{C}_t(X_{tj})\}_{j=1}^N$.
   - set value function,
$$\bar{V}_t(X_{tj}) = \max\left\{h(X_{tj}), \bar{C}_t(X_{tj})\right\}.$$

We use the value function defined by Tsitsiklis and Van Roy (1999).

The regression values are used to create a policy that is implemented on a test set: exercise when the current exercise value is greater than or equal to the estimated continuation value. A good regression model is crucial to creating a good policy.

In previous literature, $\{C_t(X_{tj})\}_{j=1}^N$ has been estimated by regression splines for a single underlying asset (Carriere, 1996), or least squares linear regression on a set of basis functions (Tsitsiklis and Van Roy, 1999; Longstaff and Schwartz, 2001; Glasserman, 2004). Regression on a set of basis functions becomes problematic when X_{tj} is defined over moderate to high dimensional spaces. Well-defined sets of bases such as radial basis functions and polynomials require an exponential number of functions to span the space, while manually selecting basis functions can be quite difficult. Since the expected continuation values are convex in the asset price for basket options, CAP is a simple, nonparametric alternative to these methods.

We compared the following methods: CAP and Fast CAP with $D = 3$, $L = 10$ for both and $P' = \min(M, 10)$, the number of random search directions in Fast CAP; the method of Magnani and Boyd (2009); regression trees with constant leaves using the Matlab function `classregtree`; least squares using the polynomial basis functions

$$(1, S_i(t), S_i^2(t), S_i^3(t), S_i(t)S_j(t), h(X_t)),\ i = 1,\ldots,M,\ j \neq i;$$

ridge regression on the same basis functions with ridge parameter chosen by 10-fold cross-validation each time period from values between 10^{-3} and 10^5.

We compared value function regression methods as follows. We simulated for both $N = 10,000$ and $N = 50,000$ training samples for a 3-month American basket option with a number of underlying assets, M, varying between 1 and 30 using a geometric Brownian motion with a drift of 0.05 and a volatility of 0.10. All assets had correlation 0.5 and starting value 100. The option had strike price 110. Policy values were approximated on 50,000 testing sample paths. An approximate upper bound

was generated using the dual martingale methods of Haugh and Kogan (2004) from value functions generated using polynomial basis functions based on the mean of the assets, $(1, Y_t, Y_t^2, Y_t^3, h(Y_t))$, where $Y_t = 1/M \sum_{i=1}^{M} X_i(t)$, with 2,000 samples. Upper and lower bounds were generated using 5 training and testing sets.

Results are displayed in Figure 10. We found that CAP and Fast CAP gave state of the art performance without the difficulties associated with linear functions, such as choosing basis functions and regularization parameters. We observed a decline in the performance of least squares as the number of assets grew due to overfitting. Ridge regularization greatly improved the least squares performance as the number of assets grew. Tree regression did poorly in all settings, likely due to overfitting in the presence of the non-symmetric error distribution generated by the geometric Brownian motion. These results suggest that CAP is robust even in less than ideal conditions, such as when data have heteroscedastic, non-symmetric error distributions.

Again, we noticed that while the performances of CAP and Fast CAP were comparable, the runtimes were about an order of magnitude different. On the larger problems, runtimes for Fast CAP were similar to those for unregularized least squares. This is likely because the number of covariates in the least squares regression grew like M^2, while all linear regressions in CAP only had M covariates.

7. Conclusions

In this article, we presented convex adaptive partitioning, a computationally efficient, theoretically sound and empirically robust method for regression subject to a convexity constraint. CAP is the first convex regression method to scale to large problems, both in terms of dimensions and number of observations. As such, we believe that it can allow the study of problems that were once thought to be computationally intractable. These include econometrics problems, like estimating consumer preference or production functions in multiple dimensions, approximating complex constraint functions for convex optimization, or creating convex value-to-go functions or response surfaces that can be easily searched in stochastic optimization.

CAP can be extended in a number of ways. First, as demonstrated in Hannah and Dunson (2012), CAP can be used in an ensemble setting—like bagging or smearing—to produce a smoother estimator. Averages of piecewise linear estimators are particularly useful in an optimization setting. They are still piecewise linear and can be searched by a linear program, but have more stable minimum locations than sparse methods like CAP and the linear fitting method of Magnani and Boyd (2009). Second, CAP can be extended to more shape constrained settings, like monotone, concave and semi-convex functions. Monotone, concave functions are concave functions with increasing slopes, which are common in economics. Although CAP is not a general purpose shape constrained inference method, a variant for monotone functions can easily be generated by placing positivity constraints on the parameters of the linear models. Semi-convex functions are convex in some dimensions but unconstrained in others. Variants of CAP could be combined with other nonparametric methods like kernels to produce efficient inference methods for partially convex functions. We believe that the methods supporting the CAP algorithm can bring efficient, theoretically sound inference to a variety of shape constrained problems that are inapproachable with traditional methods.

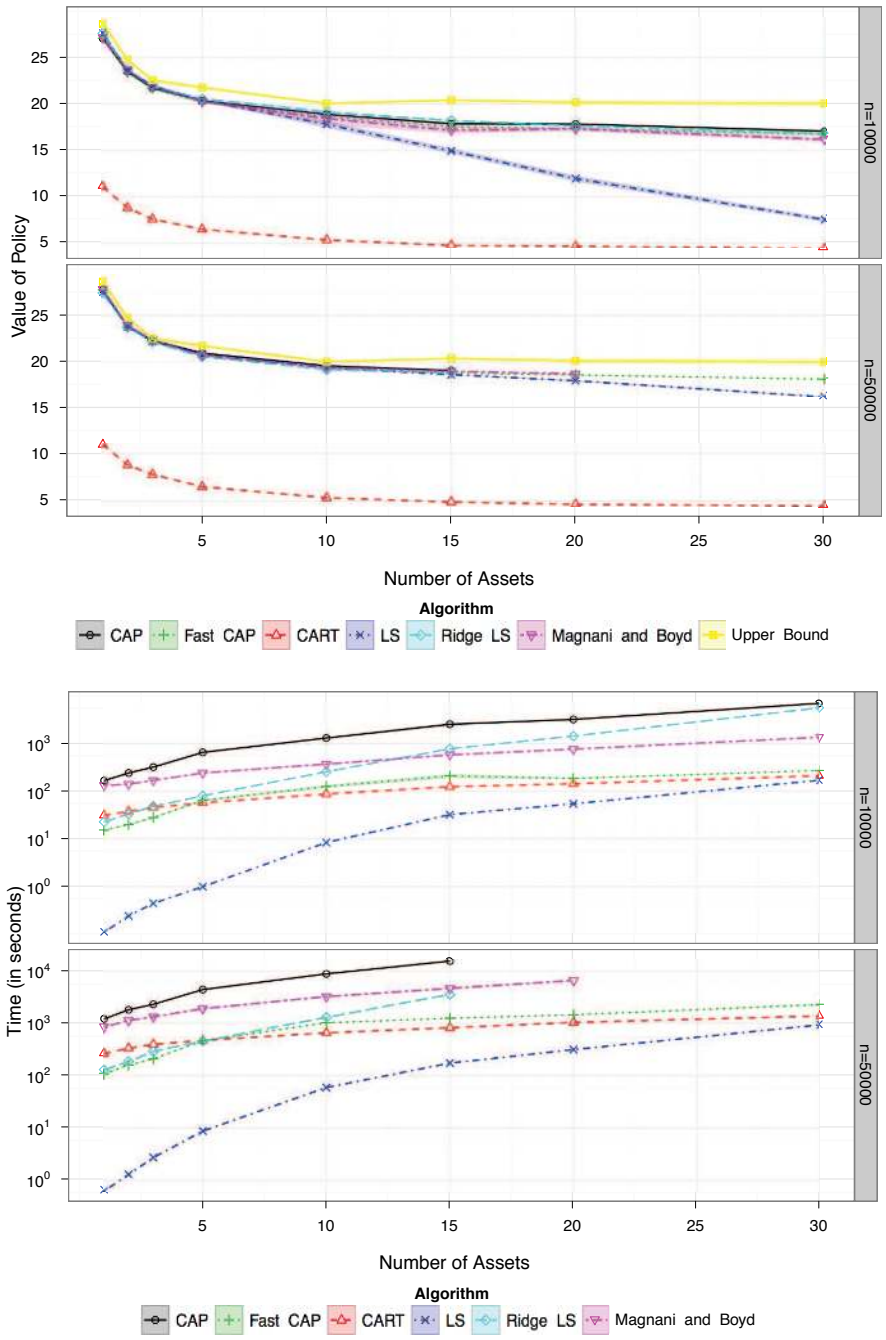

Figure 10: Average values (top) and runtimes (bottom) plus/minus one standard error for pricing America basket options as a function of the number of underlying assets are shown for CAP, Fast CAP, tree regression with constant leaves (CART), the method of Magnani and Boyd (2009), least squares (LS) and ridge regularized least squares value function approximation.

8. Online Supplements

Code for CAP can be downloaded at `http://www.columbia.edu/~lah2178/Research.html` and as an online supplement at the JMLR website.

Acknowledgments

This work was partially supported by grant R01 ES017240 from the National Institute of Environmental Health Sciences (NIEHS) of the National Institutes of Health (NIH). Lauren A. Hannah was partially supported by the Provost's Postdoctoral Fellowship at Duke University.

Appendix A.

The proof of Theorem 1 is essentially identical to the proof of Theorem 1 of Chaudhuri et al. (1994) with a few modifications. The Chaudhuri et al. (1994) results are for an algorithm that splits subsections parallel to axes. By allowing subsets to be determined by the dominating hyperplanes, the subsets are now polyhedral with a maximal number of faces determined by the dimension and maximal number of subsets. To show this, we modify Lemma 12.27 of Breiman et al. (1984).

Proof [Proof of Theorem 1] It is sufficient to show that

$$\max_{k=1,\ldots,K_n} d_{nk}^{-1} \left| [\alpha_k, \beta_k]^t - \Delta(\bar{\mathbf{x}}_k) \right| \to 0$$

in probability, where $\Delta(\bar{\mathbf{x}}_k)$ is the vector of elements $[f_0(\bar{x}_k, d_{nk}^{-1} \frac{\partial}{\partial x_1} f_0(\bar{\mathbf{x}}_k), \ldots, d_{nk}^{-1} \frac{\partial}{\partial x_p} f_0(\bar{\mathbf{x}}_k)]^t$. Let $\alpha_k^0 = \alpha_k - \beta_k^t \bar{\mathbf{x}}_k$. Assumption **A3.** ensures that the matrices D_k for all subsets are nonsingular with probability tending to 1, where $D_k = \sum_{\mathbf{x}_i \in C_k} \Gamma_i \Gamma_i^t$. Letting $Y_i = f_0(\mathbf{x}_i) + \varepsilon_i$, we have

$$[\alpha_k^0, \beta_k]^t = D_k^{-1} \sum_{\mathbf{x}_i \in C_k} \Gamma_i f_0(\mathbf{x}_i) + D_k^{-1} \sum_{\mathbf{x}_i \in C_k} \Gamma_i \varepsilon_i \tag{6}$$

for $k = 1, \ldots, K_n$ with probability tending towards 1. Doing a Taylor expansion of Equation (6), we get

$$[\alpha_k^0, \beta_k]^t - \Delta(\bar{\mathbf{x}}_k) = D_k^{-1} \sum_{i \in C_k} \Gamma_i r(\mathbf{x}_i - \bar{\mathbf{x}}_k) + D_k^{-1} \sum_{i \in C_k} \Gamma_i \varepsilon_i,$$

where $r(\mathbf{x}_i - \bar{\mathbf{x}}_k)$ are the second order and above terms of the Taylor expansion of $f_0(\bar{\mathbf{x}}_k)$. Assumptions **A1.**, **A3.** and **A4.** ensure $\max_{k=1,\ldots,K_n} \left| d_{nk}^{-1} D_k^{-1} \sum_{i \in C_k} \Gamma_i r(\mathbf{x}_i - \bar{\mathbf{x}}_k) \right| \to 0$ in probability as $n \to \infty$. To bound the random error term of Equation (6), we first assume that A_k is fixed. Applying Lemma 12.26 of Breiman et al. (1984) to each component of $d_{nk}^{-1} |C_k|^{-1} \sum_{i \in C_k} \Gamma_i \varepsilon_i$, there exist constants $h_1 > 0$, $h_2 > 0$ and $\gamma_0 > 0$ such that

$$\mathbb{P}\left(d_{nk}^{-1} \left| |C_k|^{-1} \sum_{i \in C_k} \Gamma_i \varepsilon_i \right| > \gamma \right) \leq h_1 e^{-h_2 d_{nk}^2 |C_k| \gamma^2}, \tag{7}$$

whenever $\gamma \leq \gamma_0$. Modifying Lemma 12.27 of Breiman et al. (1984) to account for the greater VC dimension of the subsets, we note **A5.** bounds the number of polyhedral faces for each subset to be

$K_n(p+2)$. Following the proof of 12.27, for m_n such that $m_n/\log(n) \to \infty$, we use Equation (7) to show

$$\begin{aligned}\mathbb{P}\left(\left|[\alpha_k^0, \beta_k]^t - \Delta(\bar{\mathbf{x}}_k)\right| \geq \gamma | \mathbf{x}_{1:n}\right) &\leq h_1 e^{-h_2\gamma^2 n d_{nk}^2 |C_k|/n}, \\ &\leq h_1 e^{-h_2\gamma^2 m_n \log(n)}, \\ &= h_1 n^{-h_2\gamma^2 m_n}\end{aligned}$$

on the event $d_{nk}^2|C_k|/n \geq m_n \log(n)/n$. Using the VC dimension of the partition,

$$\begin{aligned}&\mathbb{P}\left(\left|[\alpha_k^0, \beta_k]^t - \Delta(\bar{\mathbf{x}}_k)\right| \geq \gamma \text{ for each } A_k \text{ and } d_{nk}^2|C_k| \geq m_n D \log(n)\right) \\ &\quad \leq h_1 2^{1+K_n(p+2)} n^{K_n(p+2)-h_2\gamma^2 m_n}.\end{aligned}$$

Since **A5.** ensures that $m_n \to \infty$, the result holds.

■

Proof [Proof of Theorem 2] Fix $\varepsilon > 0$; let $d_{\mathcal{X}}$ be the diameter of $\mathcal{X}$. Choose N such that for every $n \geq N$

$$\mathbb{P}\left\{\max_{k=1,\ldots,K_n} \sup_{\mathbf{x}\in A_k} \left|\alpha_k + \beta_k^T \mathbf{x} - f_0(\mathbf{x})\right| > \frac{\varepsilon}{\zeta d_{\mathcal{X}}}\right\} < \varepsilon/2,$$

$$\mathbb{P}\left\{\max_{k=1,\ldots,K_n} \sup_{\mathbf{x}\in A_k} ||\beta_k - \nabla f_0(\mathbf{x})||_\infty > \frac{\varepsilon}{\zeta d_{\mathcal{X}}}\right\} < \varepsilon/2.$$

Fix a δ net over $\mathcal{X}$ such that at least one point of the net sits in A_k for each $k = 1, \ldots, K$. Let n_δ be the number of points in the net and let $\mathbf{x}_i^\delta$ be a point. Then,

$$\begin{aligned}\mathbb{P}\left\{\sup_{x\in\mathcal{X}} |f_n(\mathbf{x}) - f_0(\mathbf{x})| > \varepsilon\right\} &= \mathbb{P}\left\{\sup_{x\in\mathcal{X}} \left|\max_{k=1,\ldots,K_n} \alpha_k + \beta_k^T \mathbf{x} - f_0(\mathbf{x})\right| > \varepsilon\right\}, \\ &\leq \mathbb{P}\left\{\max_{i=1,\ldots,n_\delta} \left|\max_{k=1,\ldots,K_n} \alpha_k + \beta_k^T \mathbf{x}_i^\delta - f_0(\mathbf{x}_i^\delta)\right| > \frac{\varepsilon}{\zeta}\right\}, \\ &\leq \mathbb{P}\left\{\max_{i=1,\ldots,n_\delta} \left|\sum_{k=1}^{K_n} \left(\alpha_k + \beta_k^T \mathbf{x}_i^\delta\right) \mathbf{1}_{\{x_i^\delta \in A_k\}} - f_0(\mathbf{x}_i^\delta)\right| > \frac{\varepsilon}{\zeta d_{\mathcal{X}}}\right\}, \\ &< \varepsilon.\end{aligned}$$

■

Appendix B.

The CAP algorithm offers two main computational bottlenecks. First, it searches over all cardinal directions, and only cardinal directions, to produce candidate models. Second, it keeps generating models until no subsets can be split without one having less than the minimum number of observations. In most cases, the optimal number of components is much lower than the terminal number of components.

To alleviate the first problem, we suggest using P' random projections as a basis for search. Using ideas similar to compressive sensing, each projection $\mathbf{g}_j \sim N_p(0,I)$ for $j = 1,\ldots,P'$. Then we search along the direction $\mathbf{g}_j^T\mathbf{x}$ rather than x_j. When we expect the true function to live in a lower dimensional space, as is the case with superfluous covariates, we can set $P' < p$.

We solve the second problem by modifying the stopping rule. Instead of fully growing the tree until each subset has less than $2n/(2\log(n))$ observations, we use generalized cross-validation. We grow the tree until the generalized cross-validation value has increased in two consecutive iterations or each subset has less than $2n/(2\log(n))$ observations. As the generalized cross-validation error is usually concave in K, this heuristic often offers a good fit at a fraction of the computational expense of the full CAP algorithm.

The Fast CAP algorithm has the potential to substantially reduce the $\log(n)^2$ factor by halting the model generation long before K reaches $D\log(n)$. Since every feasible partition is searched for splitting, the computational complexity grows as k gets larger.

The Fast CAP algorithm is summarized as follows.

B.1 Fast Convex Adaptive Partitioning (Fast CAP)

1. **Initialize.** As in CAP.

2. **Split.**

   a. *Generate candidate splits.* Generate candidate model $\hat{M}_{jk\ell}$ by 1) fixing a subset k, 2) generating a random direction j with $\mathbf{g}_j \sim N_p(0,I)$, and 3) dividing the data as follows:

   - set $x_{min}^{jk} = \min\{\mathbf{g}_j^T\mathbf{x}_i : i \in C_k\}$, $x_{max}^{jk} = \max\{\mathbf{g}_j^T\mathbf{x}_i : i \in C_k\}$ and $b_{jk\ell} = a_\ell x_{min}^{jk} + (1 - a_\ell)x_{max}^{jk}$
   - set

   $$C_k' = \{i : i \in C_k, \mathbf{g}_j^T\mathbf{x}_i \le b_{jk\ell}\}, \qquad C_{K+1}' = \{i : i \in C_k, \mathbf{g}_j^T\mathbf{x}_i > b_{jk\ell}\},$$
   $$A_k' = \{\mathbf{x} : \mathbf{x} \in A_k, \mathbf{g}_j^T\mathbf{x} \le b_{jk\ell}\}, \qquad A_{K+1}' = \{\mathbf{x} : \mathbf{x} \in A_k, \mathbf{g}_j^T\mathbf{x} > b_{jk\ell}\}.$$

   Then new hyperplanes are fit to each of the new subsets. This is done for L knots, P' dimensions and K subsets.

   b. *Select split.* As in CAP.

3. **Refit.** As in CAP.

4. **Stopping conditions.** Let $GCV(M_K)$ be the generalized cross-validation error for model M_K. Stop if $GCV(M_K) > GCV(M_{K-1})$ and $GCV(M_{K-1}) > GCV(M_{K-2})$ of if $|C_k| < 2n_{min}$ for $k = 1,\ldots,K$. Then select final model as in CAP.

Appendix C.

In this subsection, we empirically examine the effects of the two tunable parameters, the log factor, D, and the number of knots, L. The log factor controls the minimal number of elements in each subset by setting $|C_k| \ge n/(D\log(n))$, and hence it controls the number of subsets, K, at least for large enough n. Increasing D allows the potential accuracy of the estimator to increase, but at the

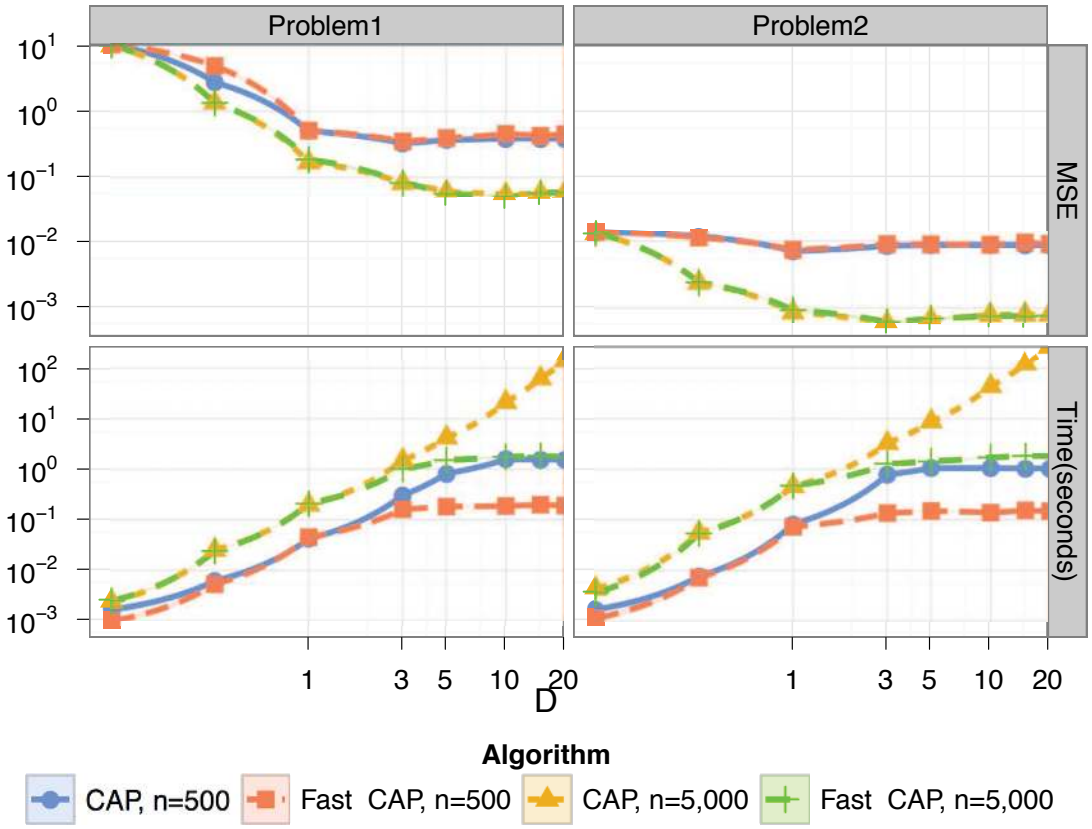

Figure 11: Log factor D (log scale) vs. mean squared error (log scale) for CAP and Fast CAP (top). Log factor D (log scale) vs. runtime in seconds (log scale) (bottom). Both methods were run on problem 1 (left) and problem 2 (right) with $n = 500$ and $n = 5,000$. Lines are mean value and shading represents one standard error.

cost of greater computational time due to the increase in possible values for K and the larger number of possibly admissible sets generated in the splitting step of CAP.

We compared values for D ranging from 0.1 to 20 on problems 1 and 2 with sample sizes of $n = 500$ and $n = 5,000$ over 100 training sets and one testing set. Results are displayed in Figure 11. Note that error may not be strictly decreasing with D because different subsets are proposed under each value. Additionally, Fast CAP is a randomized algorithm so variance in error rate and runtime is to be expected.

Empirically, once $D \geq 1$, there was little substantive error reduction in the models, but the runtime increased as $O(D^2)$ for the full CAP algorithm. Since D controls the maximum partition size, $K_n = D\log(n)$, and a linear regression is fit $K\log(K)$ times, the expected increase in the runtime should only be $O(D\log(D))$. We believe that the extra empirical growth comes from an increased number of feasible candidate splits. In the Fast CAP algorithm, which terminates after generalized cross-validation gains cease to be made, we see runtimes leveling off with higher values of D. Based

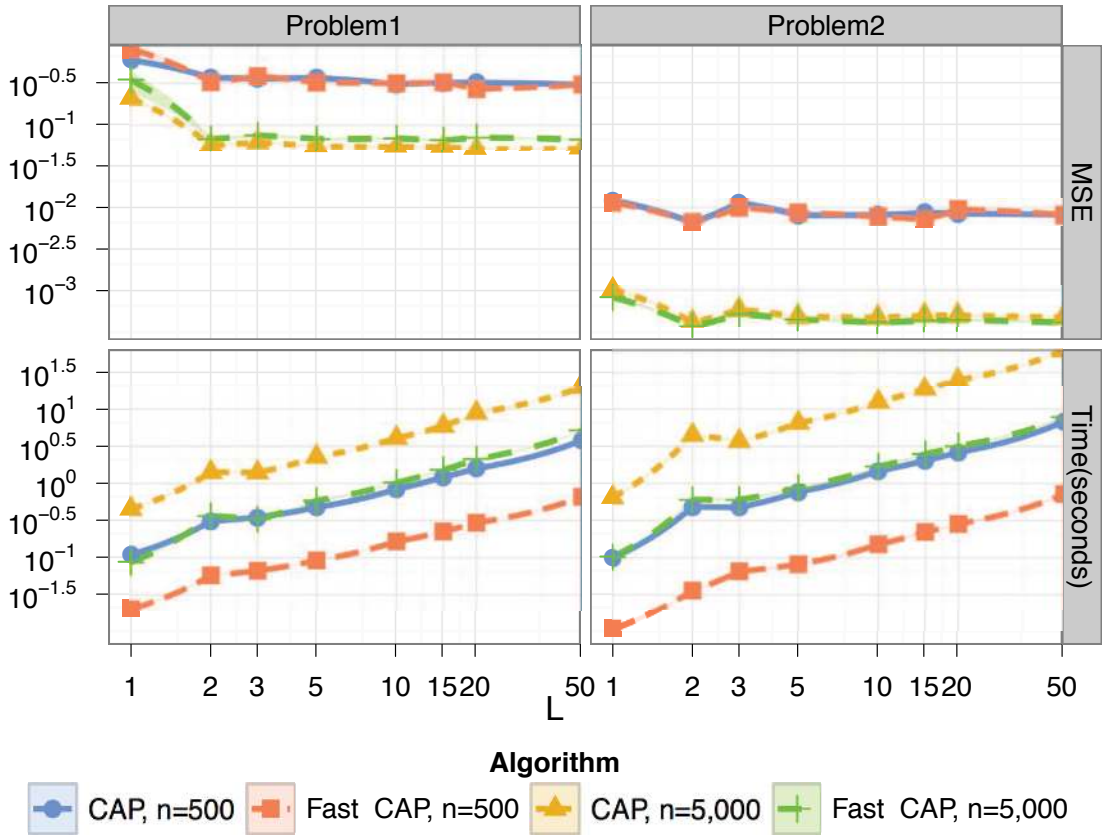

Figure 12: Number of knots L (log scale) vs. mean squared error (log scale) for CAP and Fast CAP (top). Number of knots L (log scale) vs. runtime in seconds (log scale) (bottom). Both methods were run on problem 1 (left) and problem 2 (right) with $n = 500$ and $n = 5,000$. Lines are mean value and shading represents one standard error.

on these results, we believe that setting $D = 3$ offers a good balance between fit and computational expense.

The number of knots, L, determines how many possible subsets will be examined during the splitting step. Like D, an increase in L offers a better fit at the expense of increased computation. We compared values for L ranging from 1 to 50 on problems 1 and 2 with sample sizes of $n = 500$ and $n = 5,000$ over 100 training sets and 1 testing set. Results are displayed in Figure 12.

The changes in fit and runtime are less dramatic with L than they are with D. After $L = 3$, the predictive error rates almost completely stabilized. Runtime increased as $O(L)$ as expected. Due to the minimal increase in computation, we feel that $L = 10$ is a good choice for most settings.

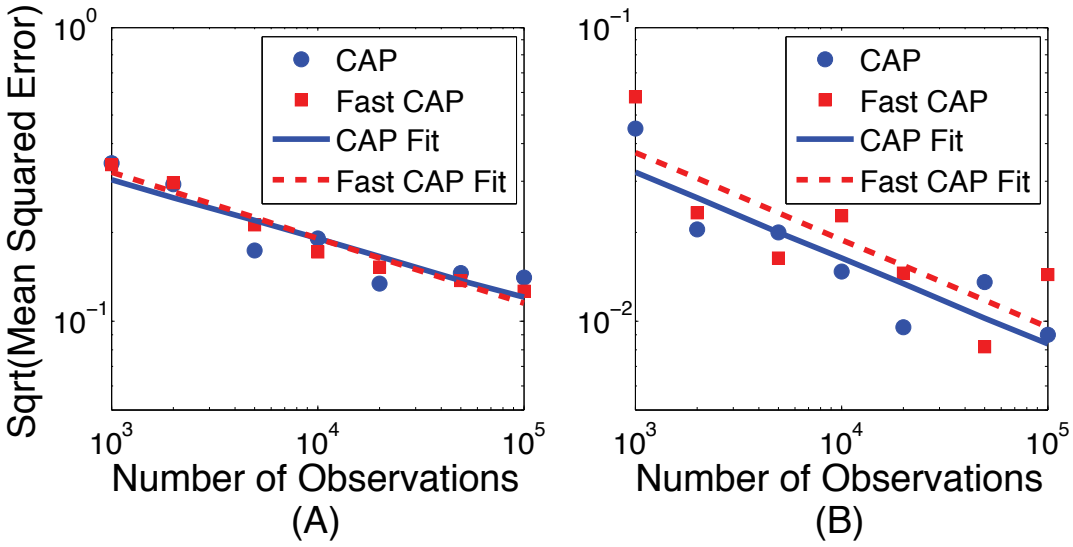

Figure 13: Number of observations n (log scale) vs. square root of mean squared error (log scale) for problem 1 (A) and problem 2 (B). Linear models are fit to find the empirical rate of convergence.

Appendix D.

Although theoretical rates of convergence are not yet available for CAP, we are able to empirically examine them. Rates of convergence for multivariate convex regression have only been studied in two articles of which we are aware. Aguilera et al. (2011) studied rates of convergence for an estimator that is created by first smoothing the data, then evaluating the smoothed data over an ε-net, and finally convexifying the net of smoothed data by taking the convex hull. They showed that the convexify step preserved the rates of the smoothing step. For most smoothing algorithms, these are minimax nonparametric rates, $n^{-\frac{1}{p+2}}$ with respect to the empirical ℓ_2 norm.

Hannah and Dunson (2011) showed adaptive rates for a Bayesian model that places a prior over the set of all piecewise linear functions. Specifically, they showed that if the true mean function f_0 actually maps a d-dimensional linear subspace of $\mathcal{X}$ to $\mathbb{R}$, that is

$$f_0(\mathbf{x}) = g_0(\mathbf{x}\mathbf{A}), \qquad \mathbf{A} \in \mathbb{R}^{p\times d},$$

then their model achieves rates of $\log^{-1}(n)n^{-\frac{1}{d+2}}$ with respect to the empirical ℓ_2 norm. Empirically, we see these types of adaptive rates with CAP.

In Figure 13, we plotted the number of observations against the square root of the mean squared error in a log-log plot for problems 1 and 2. We then fitted a linear model for both CAP and Fast CAP. For problem 1, $p = 5$ but $d = 3$, due to the sum in the quadratic term. Likewise, for problem 2, $p = 10$ but $d = 1$ because it is an exponential of a linear combination. Under standard nonparametric rates, we would expect the slope of the linear model to be $-\frac{1}{7}$ for problem 1 and $-\frac{1}{12}$ for problem 2. However, we see slopes closer to $-\frac{1}{5}$ and $-\frac{1}{3}$ for problems 1 and 2, respectively; values are given

Method	Problem 1	Problem 2
Expected: Rates in p	-0.1429	-0.0833
Expected: Rates in d	-0.2000	-0.3333
Empirical: CAP	-0.2003	-0.2919
Empirical: Fast CAP	-0.2234	-0.2969

Table 2: Slopes for linear models fit to $\log(n)$ vs. $\log(\sqrt{MSE})$ in Figure 13. Expected slopes are given when: 1) rates are with respect to full dimensionality, p, and 2) rates are with respect to dimensionality of linear subspace, d. Empirical slopes are fit to mean squared error generated by CAP and Fast CAP. Note that all empirical slopes are closest to those for linear subspace rates rather than those for full dimensionality rates.

in Table 2. These results strongly imply that CAP achieves adaptive convergence rates of the type shown by Hannah and Dunson (2011) for problems 1 and 2.

References

Néstor Aguilera and Pedro Morin. Approximating optimization problems over convex functions. *Numerische Mathematik*, 111(1):1–34, 2008.

Néstor Aguilera and Pedro Morin. On convex functions and the finite element method. *SIAM Journal on Numerical Analysis*, 47(1):3139–3157, 2009.

Néstor Aguilera, Liliana Forzani, and Pedro Morin. On uniform consistent estimators for convex regression. *Journal of Nonparametric Statistics*, 23(4):897–908, 2011.

Yacine Aīt-Sahalia and Jefferson Duarte. Nonparametric option pricing under shape restrictions. *Journal of Econometrics*, 116(1–2):9–47, 2003.

William P. Alexander and Scott D. Grimshaw. Treed regression. *Journal of Computational and Graphical Statistics*, 5(2):156–175, 1996.

Gad Allon, Michael Beenstock, Steven Hackman, Ury Passy, and Alexander Shapiro. Nonparametric estimation of concave production technologies by entropic methods. *Journal of Applied Econometrics*, 22(4):795–816, 2007.

Herman J. Bierens and Donna K. Ginther. Integrated conditional moment testing of quantile regression models. *Empirical Economics*, 26(1):307–324, 2001.

Melanie Birke and Holger Dette. Estimating a convex function in nonparametric regression. *Scandinavian Journal of Statistics*, 34(2):384–404, 2007.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, United Kingdom, 2004.

Stephen Boyd, Seung-Jean Kim, Lieven Vandenberghe, and Arash Hassibi. A tutorial on geometric programming. *Optimization and Engineering*, 8(1):67–127, 2007.

Leo Breiman. Hinging hyperplanes for regression, classification, and function approximation. *IEEE Transactions on Information Theory*, 39(3):999–1013, 1993.

Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

Leo Breiman. Randomizing outputs to increase prediction accuracy. *Machine Learning*, 40(3): 229–242, 2000.

Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and Regression Trees*. Chapman & Hall/CRC, Boca Raton, Florida, 1984.

Hugh D. Brunk. Maximum likelihood estimates of monotone parameters. *The Annals of Mathematical Statistics*, 26(4):607–616, 1955.

Jacques F. Carriere. Valuation of the early-exercise price for options using simulations and nonparametric regression. *Insurance: Mathematics and Economics*, 19(1):19–30, 1996.

I-Shou Chang, Li-Chu Chien, Chao A. Hsiung, Chi-Chung Wen, and Yuh-Jenn Wu. Shape restricted regression with random Bernstein polynomials. *IMS Lecture Notes-Monograph Series*, 54:187–202, 2007.

Probal Chaudhuri, Min-Ching Huang, Wei-Yin Loh, and Ruji Yao. Piecewise-polynomial regression trees. *Statistica Sinica*, 4(1):143–167, 1994.

Probal Chaudhuri, Wen-Da Lo, Wei-Yin Loh, and Ching-Ching Yang. Generalized regression trees. *Statistica Sinica*, 5(1):641–666, 1995.

Madeleine Cule and Richard Samworth. Theoretical properties of the log-concave maximum likelihood estimator of a multidimensional density. *Electronic Journal of Statistics*, 4:254–270, 2010.

Madeleine Cule, Richard Samworth, and Michael Stewart. Maximum likelihood estimation of a multi-dimensional log-concave density. *Journal of the Royal Statistical Society, Series B*, 72(5): 545–607, 2010.

Warren Dent. A note on least squares fitting of functions constrained to be either nonnegative, nondecreasing or convex. *Management Science*, 20(1):130–132, 1973.

Alin Dobra and Johannes Gehrke. SECRET: a scalable linear regression tree algorithm. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 481–487, Berlin, Germany, 2002. ACM.

Richard L. Dykstra. An algorithm for restricted least squares regression. *Journal of the American Statistical Association*, 78(384):837–842, 1983.

Donald A. S. Fraser and Hélène Massam. A mixed primal-dual bases algorithm for regression under inequality constraints. Application to concave regression. *Scandinavian Journal of Statistics*, 16 (1):65–74, 1989.

Jerome H. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1): 1–141, 1991.

Paul Glasserman. *Monte Carlo Methods in Financial Engineering*. Springer Verlag, New York, New York, 2004.

Gene H. Golub, Michael Heath, and Grace Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.0 beta. `http://cvxr.com/cvx`, September 2012.

Piet Groeneboom, Geurt Jongbloed, and Jon A. Wellner. Estimation of a convex function: characterizations and asymptotic theory. *Annals of Statistics*, 29(6):1653–1698, 2001.

László Györfi, Michael Kohler, Adam Krzyżak, and Harro Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer, New York, New York, 2002.

Peter Hall and Li-Shan Huang. Nonparametric kernel regression subject to monotonicity constraints. *The Annals of Statistics*, 29(3):624–647, 2001.

Lauren A. Hannah and David B. Dunson. Bayesian nonparametric multivariate convex regression. Technical Report arXiv:1109.0322v1, Department of Statistical Science, Duke University, Durham, North Carolina, 2011.

Lauren A. Hannah and David B. Dunson. Ensemble methods for convex regression with applications to geometric programming based circuit design. In *Proceedings of the 29th Annual International Conference on Machine Learning*, Edinburgh, United Kingdom, 2012.

D. L. Hanson and Gordon Pledger. Consistency in concave regression. *The Annals of Statistics*, 4 (6):1038–1050, 1976.

Martin B. Haugh and Leonid Kogan. Pricing American options: a duality approach. *Operations Research*, 52(2):258–270, 2004.

Daniel J. Henderson and Christopher F. Parmeter. Imposing economic constraints in nonparametric regression: survey, implementation and extension. In Q. Li and J. S. Racine, editors, *Nonparametric Econometric Methods (Advances in Econometrics)*, volume 25, pages 433–469. Emerald Publishing Group Limited, Bingley, United Kingdom, 2009.

Clifford Hildreth. Point estimates of ordinates of concave functions. *Journal of the American Statistical Association*, 49(267):598–619, 1954.

Charles A. Holloway. On the estimation of convex functions. *Operations Research*, 27(2):401–407, 1979.

Arezou Keshavarz, Yang Wang, and Stephen Boyd. Imputing a convex objective function. In *Proceedings of the IEEE International Symposium on Intelligent Control*, pages 613–619, Denver, Colorado, 2011.

Jintae Kim, Jaeseo Lee, Lieven Vandenberghe, and C.K.-Ken Yang. Techniques for improving the accuracy of geometric-programming based analog circuit design optimization. In *Proceedings of the IEEE International Conference on Computer Aided Design*, pages 863–870, San Jose, California, 2004.

Farinaz Koushanfar, Mehrdad Majzoobi, and Miodrag Potkonjak. Nonparametric combinatorial regression for shape constrained modeling. *IEEE Transactions on Signal Processing*, 58(2):626–637, 2010.

Timo Kuosmanen. Representation theorem for convex nonparametric least squares. *Econometrics Journal*, 11(2):308–325, 2008.

Timo Kuosmanen and Andrew L. Johnson. Data envelopment analysis as nonparametric least-squares regression. *Operations Research*, 58(1):149–160, 2010.

Eunji Lim. Response surface computation via simulation in the presence of convexity constraints. In *Proceedings of the 2010 Winter Simulation Conference*, pages 1246–1254, Baltimore, Maryland, 2010.

Eunji Lim and Peter W. Glynn. Consistency of multi-dimensional convex regression. *Operations Research*, 60(1):196–208, 2012.

Francis A. Longstaff and Eduardo S. Schwartz. Valuing American options by simulation: a simple least-squares approach. *Review of Financial Studies*, 14(1):113–147, 2001.

Alessandro Magnani and Stephen Boyd. Convex piecewise-linear fitting. *Optimization and Engineering*, 10(1):1–17, 2009.

Enno Mammen. Nonparametric regression under qualitative smoothness assumptions. *The Annals of Statistics*, 19(2):741–759, 1991.

Mary C. Meyer. Inference using shape-restricted regression splines. *Annals of Applied Statistics*, 2 (3):1013–1033, 2008.

Mary C. Meyer, Amber J. Hackstadt, and Jennifer A. Hoeting. Bayesian estimation and inference for generalised partial linear models using shape-restricted splines. *Journal of Nonparametric Statistics*, 23(4):867–884, 2011.

Renato D. C. Monteiro and Ilan Adler. Interior path following primal-dual algorithms. Part II: convex quadratic programming. *Mathematical Programming*, 44(1–3):43–66, 1989.

Brian Neelon and David B. Dunson. Bayesian isotonic regression and trend analysis. *Biometrics*, 60(2):398–406, 2004.

Andrew Nobel. Histogram regression estimation using data-dependent partitions. *The Annals of Statistics*, 24(3):1084–1105, 1996.

Duncan Potts and Claude Sammut. Incremental learning of linear model trees. *Machine Learning*, 61(1):5–48, 2005.

Warren B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley-Interscience, Hoboken, New Jersey, 2007.

J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, Inc., San Mateo, California, 1993.

Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, Massachusetts, 2006.

Sanghamitra Roy, Weijen Chen, Charlie Chung-Ping Chen, and Yu Hen Hu. Numerically convex forms and their application in gate sizing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(9):1637–1647, 2007.

Dominic Schuhmacher and Lutz Dümbgen. Consistency of multivariate log-concave density estimators. *Statistics & Probability Letters*, 80(5-6):376–380, 2010.

Emilio Seijo and Bodhisattva Sen. Nonparametric least squares estimation of a multivariate convex regression function. *The Annals of Statistics*, 39(3):1580–1607, 2011.

Thomas S. Shively, Thomqw W. Sager, and Stephen G. Walker. A Bayesian approach to nonparametric monotone function estimation. *Journal of the Royal Statistical Society, Series B*, 71 (1):159–175, 2009.

Thomas S. Shively, Stephen G. Walker, and Paul Damien. Nonparametric function estimation subject to monotonicity, convexity and other shape constraints. *Journal of Econometrics*, 161(2): 166–181, 2011.

Huseyin Topaloglu and Warren B. Powell. An algorithm for approximating piecewise linear concave functions from sample gradients. *Operations Research Letters*, 31(1):66–76, 2003.

Alejandro Toriello, George Nemhauser, and Martin Savelsbergh. Decomposing inventory routing problems with approximate value functions. *Naval Research Logistics*, 57(8):718–727, 2010.

John N. Tsitsiklis and Benjamin Van Roy. Optimal stopping of Markov processes: Hilbert space theory, approximation algorithms, and an application to pricing high-dimensional financial derivatives. *IEEE Transactions on Automatic Control*, 44(10):1840–1851, 1999.

John N. Tsitsiklis and Benjamin Van Roy. Regression methods for pricing complex American-style options. *IEEE Transactions on Neural Networks*, 12(4):694–703, 2001.

Berwin A. Turlach. Shape constrained smoothing using smoothing splines. *Computational Statistics*, 20(1):81–103, 2005.

Hal R. Varian. The nonparametric approach to demand analysis. *Econometrica*, 50(4):945–973, 1982.

Hal R. Varian. The nonparametric approach to production analysis. *Econometrica*, 52(3):579–597, 1984.

Yongqiao Wang and He Ni. Multivariate convex support vector regression with semidefinite programming. *Knowledge-Based Systems*, 30:87–94, 2012.

C. F. Jeff Wu. Some algorithms for concave and isotonic regression. In S. H. Zanakis and J. S. Rustagi, editors, *Studies in the Management Sciences*, volume 19, pages 105–116. North-Holland, Amsterdam, 1982.