Efficient Linear Programming Algorithm for Functional Component Pursuit

Yonggang Yao
Department of Statistics
The Ohio State University
(joint with Dr. Yoonkyung Lee)

October 11, 2007
Outline

Introduction
  - Regularization
  - LASSO Penalty

Kernel Selection
  - Structured Regularization
  - Functional Component Pursuit
  - Solution Path

Linear Programming
  - Parametric-cost LP
  - Simplex Algorithm

Example of Classification
  - $\theta$-step for MSVM
  - Simulation Study

Conclusion and Future Works
Regularization

- Regularization is one of the popular methods for modeling and prediction, which combines the information in the data with other limitations on model spaces. (The idea of regularity can be found in Tikhonov (1943).)
- A regularization problem minimizes:

\[ \mathcal{L}(f) + \lambda J(f) \] w.r.t. \( f \in \mathcal{F}, \)

where \( f \) is an element in the model space \( \mathcal{F} \), \( \mathcal{L}(f) \) is the empirical risk of model \( f \), \( J(f) \) is the penalty of \( f \) due to the additional information, and \( \lambda \) is the tuning parameter.
The penalty is often based on roughness, norm, sparsity, or margin associated with model $f$.

Many penalty components can be explained as the attributes of certain prior distributions in the Bayesian methods (e.g., the solutions of many regularization can be viewed as posterior modes).
To simultaneously achieve model fitting and variable selection for linear regression problem, Tibshirani (1996) proposed the LASSO method, an $l_1$-norm penalty regression method, which fits models by minimizing

$$
\frac{1}{n} \sum_{i=1}^{n} \|y_i - x_i \beta\|_2^2 + \lambda \|\beta\|_1 \text{ w.r.t. } \beta \in \mathbb{R}^p,
$$

(1)

where $p$ is the number of predictor variables.

- The LASSO regression can select predictor variables by shrinking the coefficients of other predictor variables to zero.
- The $l_1$-norm penalty $\|\beta\|_1$ is often called LASSO penalty.
**Figure:** Estimation plots for LASSO (left) and RIDGE (right) regression.
Roughness Penalty

- Roughness of a continuous function $f$ often refers to:

$$J(f) = \int_\chi [f^{(m)}(t)]^2 dt \text{ for some integer } m > 0$$

or a positive linear combination of the quantities in this form.

- Under some regular conditions, the quantity can be expressed as the norm associated with certain Hilbert space $\mathcal{H}$, such that

$$\int_\chi [f^{(m)}(t)]^2 dt = < f, f >_{\mathcal{H}}.$$

- The class of regularization problems with roughness penalty often take the form of minimizing:

$$\frac{1}{n} \sum_{i=1}^{n} L[y_i, f(x_i)] + \lambda < f, f >_{\mathcal{H}} \text{ w.r.t. } f \in \mathcal{H}.$$

- $\mathcal{H}$ is also called a feature space.
• A reproducing kernel Hilbert space (RKHS) is characterized by its reproducing kernel function, $K(\cdot, \cdot)$, such that, for any $f \in \mathcal{H}$, we have
\[
    f(t) = \langle K(\cdot, t), f(\cdot) \rangle_{\mathcal{H}}.
\]
• By the representor theorem, the previous regularization problem is equivalent to minimizing
\[
    \frac{1}{n} \sum_{i=1}^{n} L[y_i, c'K_i] + \lambda c'Kc \text{ w.r.t. } c \in \mathcal{R}^n
\]
where $K_i := [K(x_1, x_i), \cdots, K(x_n, x_i)]'$ and $K := [K_1, \cdots, K_n]$. 

Examples of RKHS Methods

- Examples of the applications include
  1. Smoothing Spline ANOVA with square loss (Wahba 1990)
  2. Support Vector Machine (SVM) with hinge loss (Vapnik 1998)
  3. Nonparametric Quantile Regression (QR) with check loss (Koenker 2005)
Kernel Selection

- In practice, the kernel function $K(\cdot, \cdot)$ is often pre-specified before formulating the objective function in (2).
- Since kernel determines the feature space $\mathcal{H}$, kernel selection also amounts to **feature selection** under this scenario.
- By further parameterizing the kernel structure, many current nonparametric regularization methods can be extended to achieve the objectives of both model-fitting and feature selection.
- Consider the kernel space which is a non-negative linear combination of a set of pre-specified kernel functions \( \{K_1, \cdots, K_d\} \):

\[
\mathcal{K} := \left\{ K = \sum_{v=1}^{d} \theta_v K_v : \theta := (\theta_1, \cdots, \theta_d)' \geq 0, \theta \in \mathbb{R}^d \right\}. \tag{3}
\]
Structured Regularization

- COmponent Selection and Smoothing Operator (COSSO) method (Lin and Zhang 2006) provides an approach to the kernel selection problem by minimizing:

\[
\frac{1}{n} \sum_{i=1}^{n} L[y_i, f(x_i)] + \lambda \sum_{v=1}^{d} \theta_v^{-1} < f, f >_{\mathcal{H}_v} + \lambda \theta \sum_{v=1}^{d} \theta_v
\]

with respect to \( f \in \mathcal{H} := 1 \oplus \left[ \bigoplus_{v=1}^{d} \mathcal{H}_v \right] \), where \( < f, f >_{\mathcal{H}_v} \) refers to the inner product of \( \mathcal{H}_v \) characterized by \( K_v \).

- The structured regularization problem is equivalent to minimizing

\[
\frac{1}{n} \sum_{i=1}^{n} L[y_i, c'] \sum_{v=1}^{d} \theta_v K_{vi}] + \lambda \sum_{v=1}^{d} \theta_v c' K_v c + \lambda \theta \sum_{v=1}^{d} \theta_v \quad \text{(4)}
\]

with respect to \( c \in \mathcal{R}^n \) and \( \theta \geq 0 \), where \( K_{vi} := [K_v(x_1, x_i), \cdots, K_v(x_n, x_i)]' \), and \( K_v := [K_{v1}, \cdots, K_{vn}] \).

- COSSO applies the LASSO penalty to the coefficients of the functional components.
• To solve the problem in (4), one can alternately update \( c \) and \( \theta \) till convergence for fixed \( \lambda \) and \( \lambda_\theta \).
• \( c \)-step, for fixed \( \lambda_\theta \) and \( \theta \), solve the ordinary regularization problem;
• \( \theta \)-step, for fixed \( \lambda \) and \( c \), solve another regularization problem with LASSO penalty.
• Updating \( c \) and \( \theta \) a few times is sufficient to achieve convergence.
• \( \theta \)-step amounts to shrinkage and selection of functional components.
• The functional component pursuit is a nonparametric generalization of basis pursuit (Chen et al. 2001).
• The functional Component Pursuit can be embodied by adopting
  ▶ hinge loss \( (1 - y_i f(x_i))_+ \) with SVM,
  ▶ check loss \( \rho_{\tau}(t) = \tau t^+ + (1 - \tau)t^- \) with QR,
  ▶ \( \sup_{1 \leq t \leq p} |X'(Y - X\beta)| \) with Dantzig selector
as \( \mathcal{L} \) in the structure regularization problem. (See Candes and Tao 2007 for Dantzig selector.)
• The three cases are all of convex and piecewise linear \( \mathcal{L} \).
• The penalty associated with kernel functions is LASSO penalty, which is also piecewise linear.
• With a piecewise linear \( \mathcal{L} \), the \( \theta \)-step is actually a parametric cost Linear Programming (LP) problem.
• Every regularization problem gives rise to a continuum of optimization problems indexed by the tuning parameter $\lambda$.
• Solution path refers to the sequence of the optimal models $f$ as a function of $\lambda$.
• Goodness of $f$ depends on the value of $\lambda$, the choice of which is a model selection problem.
• In general, it is difficult to accurately generate the solution path due to computation complexity.
• There are some fast solution-path algorithms for LARS (Efron et al. 2004), SVM solution path (Hastie et al. 2004), and Multi-category SVM solution path (Lee and Cui 2006), etc..
Linear Programming

- Linear Programming (LP) is one of the milestones of the optimization theory (Kantorovich 1939; Dantzig 1947).
- One of the standard forms of LP is

\[
\begin{aligned}
\min_{z \in \mathbb{R}^N} & \quad c'z \\
\text{s.t.} & \quad Az = b, \\
& \quad z \geq 0
\end{aligned}
\]

(5)

where \( z \) is an \( N \)-vector of variables, \( c \) is a fixed \( N \)-vector, \( b \) is a fixed \( M \)-vector, and \( A \) is an \( M \times N \) fixed matrix with rank-\( M \).

- LP problem can be solved by the Simplex algorithm.
LP Terminology

• A set \( B^* := \{B_1^*, \ldots, B_M^*\} \subset N := \{1, \ldots, N\} \) is called a basic index set, if \( A_{B^*} := [A_{B_1^*}, \ldots, A_{B_M^*}] \) is invertible, where \( A_{B_i^*} \) is the \( B_i^* \)th column vector of \( A \) for \( i \in M := \{1, \ldots, M\} \).

• \( z^* \in \mathbb{R}^N \) is called the basic solution associated with \( B^* \), if \( z^* \) is determined by

\[
\begin{align*}
  z_{B^*}^* := (z_{B_1^*}^*, \ldots, z_{B_M^*}^*)' &= A_{B^*}^{-1}b \\
  z_j^* &= 0 \text{ for } j \in N \setminus B^*.
\end{align*}
\]

• A basic index set \( B^* \) is also called an optimal basic index set if

\[
A_{B^*}^{-1}b \geq 0 \text{ and } \left[ c - A' (A_{B^*}^{-1})' c_{B^*} \right] \geq 0
\]

• Let \( B^* \) be an optimal basic index set and \( z^* \) be its associated basic solution, then \( z^* \) is an optimal solution for the LP problem.

• Remark: the standard LP problem can be solved by finding its optimal basic index set.
Geometry of Linear Programming

Figure: Illustration of the Standard Linear Programming
The standard form of a parametric-cost LP is defined as
\[
\min_{z \in \mathbb{R}^N} (c + \lambda a)'z \quad \text{s.t.} \quad Az = b \quad \text{and} \quad z \geq 0
\] (6)

The solution path of (6) is a stepwise function:
\[
z(\lambda) = \begin{cases} 
  z^0 & \text{for } \lambda > \lambda_0; \\
  z^l & \text{for } \lambda_l < \lambda < \lambda_{l-1} \text{ and } l = 1, \cdots, J; \\
  \tau z^l + (1 - \tau)z^{l+1} & \text{for } \lambda = \lambda_l, \tau \in [0, 1] \text{ and } l = 0, \cdots, J - 1.
\end{cases}
\]
For fixed $\lambda^* \geq 0$, let $B^*$ be an optimal basic index set of the problem in (6) at $\lambda = \lambda^*$. Define

$$
\lambda := \max \left\{ j : \tilde{a}_j^* > 0; j \in \mathcal{N} \setminus B^* \right\} \left( -\frac{\tilde{c}_j^*}{\tilde{a}_j^*} \right)
$$

and

$$
\bar{\lambda} := \min \left\{ j : \tilde{a}_j^* < 0; j \in \mathcal{N} \setminus B^* \right\} \left( -\frac{\tilde{c}_j^*}{\tilde{a}_j^*} \right)
$$

where $\tilde{a}_j^* := a_j - a'_{B^*} A_{B^*}^{-1} A_j$ and $\tilde{c}_j^* := c_j - c'_{B^*} A_{B^*}^{-1} A_j$ for $j \in \mathcal{N}$. Then, $B^*$ is an optimal basic index set of (6) for $\lambda \in [\lambda, \bar{\lambda}]$. 
Simpelex Algorithm for Parametric-cost LP

1. Initialize the optimal basic index set at $\lambda_{-1} = \infty$ with $B^0$.
2. Given $B^l$, the $l$th optimal basic index set at $\lambda = \lambda_{l-1}$, determine the solution $z^l$ by $z^l_{B^l} = A^{-1}_B b$ and $z^l_j = 0$ for $j \in \mathcal{N} \setminus B^l$.
3. Find the entering index

$$j^l = \arg \max_{j : \tilde{\alpha}_j^l > 0; j \in \mathcal{N} \setminus B^l} \left(-\frac{\tilde{\zeta}_j^l}{\tilde{\alpha}_j^l}\right).$$

4. Find the leaving index

$$i^l = \arg \min_{i \in \{j : d_j^l < 0, j \in B^l\}} \left(-\frac{z_i^l}{d_i^l}\right).$$

If there are multiple indices, choose one of them.
5. Update the optimal basic index set to $B^{l+1} = B^l \cup \{j^l\} \setminus \{i^l\}$.
6. Terminate the algorithm if $\tilde{\zeta}_{j^l}^l \geq 0$ or equivalently $\lambda_l \leq 0$. Otherwise, repeat 2 – 5.
Refinement of the Simplex Algorithm

- The previous Simplex algorithm may have cycling problem due to some possible degenerate joint solution. (Cycling problem means \( z^l = z^{l+1} = \cdots \) and \( \lambda_l = \lambda_{l+1} = \cdots \), such that the algorithm terminating condition can never be satisfied.)
- Tableau-simplex algorithm with anti-cycling property is more suitable for general settings, while its computational complexity is much higher than the simple Simplex algorithm.
- By utilizing the structure of the \( \theta \)-step for MSVM, the computational complexity of the Tableau-simplex algorithm could be reduced to achieve efficient computation. See Yao and Lee (2007) for more discussion on fast Tableau-simplex algorithm.
Formulation of Multi-category SVM

Let \( f^j(x) := \beta^j_0 + \sum_{i=1}^{n} \sum_{v=1}^{d} \theta_v K_v(x, x_i) \beta^j_i \). By the representor theorem, the original regularization problem of \( k \)-category SVM (Lee et al. 2004) becomes:

\[
\frac{1}{n} \sum_{j=1}^{k} (L^j)' \left( y^j - \beta^j_0 1 - \sum_{v=1}^{d} \theta_v K_v \beta^j \right) + \lambda \frac{1}{2} \sum_{j=1}^{k} (\beta^j)' \left( \sum_{v=1}^{d} \theta_v K_v \right) \beta^j + \lambda_\theta \sum_{v=1}^{d} \theta_v \\
\text{s.t.} \quad \theta_v \geq 0; \ v = 1, \cdots, d.
\]

where \( \lambda \geq 0 \) and \( \lambda_\theta \geq 0 \) are tuning parameters, \( L^j \) is a fixed weight vector for the loss of \( j \)-th category, \( y^j \) is the indicator vector of the \( j \)-th category for training data, \( \beta^j_0 \) and \( \beta^j := (\beta^j_1, \cdots, \beta^j_n)' \) contain the \( j \)-th-category variable coefficients, \( K_v \) is the \( v \)-th pre-specified kernel function, and \( \theta_v \) is the coefficient of \( K_v \) for \( j = 1, \cdots, k \) and \( v = 1, \cdots, d \).
\( \theta \)-step of Multi-category SVM

- Let \( \zeta^j := y^j - \beta_0^j \mathbf{1} - \sum_{v=1}^{d} \theta_v \mathbf{K}_v \beta^j; j = 1, \cdots, k \), and

\[
\begin{aligned}
\mathbf{X} &:= \begin{bmatrix}
\mathbf{K}_1 \beta^1, & \cdots, & \mathbf{K}_d \beta^1 \\
& \vdots & \vdots \\
\mathbf{K}_1 \beta^k, & \cdots, & \mathbf{K}_d \beta^k
\end{bmatrix} \\
\mathbf{L} &:= \frac{1}{n} \left( (\mathbf{L}^1)' , \cdots , (\mathbf{L}^k)' \right)' \\
g &:= (g_1, \cdots, g_d)' \\
\text{with } g_v &:= \frac{\lambda}{2} \sum_{j=1}^{k} (\beta^j)' \mathbf{K}_v \beta^j \text{ for } v = 1, \cdots, d, \\
\zeta &:= \left( (\zeta^1)' , \cdots , (\zeta^k)' \right)' , \\
\theta &:= (\theta_1, \cdots, \theta_d)' .
\end{aligned}
\]
• We can match the symbols in (8) and (6), which gives

\[
\begin{align*}
\mathbf{z} &:= (\theta', (\zeta^+)', (\zeta^-)')' \\
\mathbf{c} &:= (g', 0, L')'
\end{align*}
\]

\[
\begin{align*}
\mathbf{a} &:= (1', 0', 0')'
\end{align*}
\]

\[
\begin{align*}
\mathbf{A} &:= (X, I, -I')'
\end{align*}
\]

\[
\begin{align*}
\mathbf{b} &:= \left( (y^1 - \beta_0^1 \mathbf{1}')', \ldots, (y^k - \beta_0^k \mathbf{1}')' \right)'
\end{align*}
\]

with \( M = nk \) and \( N = (d + 2nk) \).

Then, the equivalent form in (8) becomes the standard parametric - cost LP form defined in (6).
Simulation Study

• Following Lee et al. (2006), we consider a three-category toy example.
• \( \mathbf{x}_i := (x_{i1}, x_{i2}) \sim \text{Uniform}[0, 1] \times \text{Uniform}[0, 1] \) i.i.d. for \( i = 1, \ldots, n; \)
• The true kernel is a two-way interaction spline kernel:

\[
K(\mathbf{x}_i, \mathbf{x}_j) = \theta_1 K(x_{i1}, x_{j1}) + \theta_2 K(x_{i2}, x_{j2}) + \theta_3 K(x_{i1}, x_{j1}) K(x_{i2}, x_{j2});
\]

• Set \( \theta := (\theta_1, \theta_2, \theta_3) = (30, 6, 0). \)
• Set \( \mathbf{c} \) to be a fixed \( n \times 3 \) matrix;
• Simulate \( y_i \) by setting \( y_i \) to be the index of the largest element in the \( i \)th row of the matrix \( [K\mathbf{c} + \sigma \mathbf{\epsilon}]_{n \times 3} \) (e.g., \( y_i = 2 \) if the \( i \)th row is \( (.12, .5, -.34) \)), where \( K \) is the kernel matrix due to the true kernel, and \( \mathbf{\epsilon} \) is a \( n \times 3 \) standard Gaussian white noise matrix (i.e., the elements in \( \mathbf{\epsilon} \) are independently and identically distributed from \( N(0, 1) \)).
First c-step

- Assume $\theta = (1, 1, 1)$ and minimize the objective function in (8) with respect to $c$ only.
- By using 5-fold Cross Validation (CV) with respect to hinge loss, we choose the optimal $c$ corresponding to $\lambda = 2^{-21}$.

![Graph showing relationship between log2(λ) and cv.hinge values]
• Set \( c \) to be the one generated by the former \( c \)-step.
• Use Tableau-simplex method to generate the solution path with respect to \( \lambda_\theta \).
By using 5-repetition-5-fold Cross Validation (CV) with respect to hinge loss, we choose the optimal $\theta$ corresponding to

$$-\log_2(\lambda_\theta) = 3.45.$$
Second c-step

- Set $\theta$ to be the one generated by the former $\theta$-step and minimize the objective function in (8) with respect to $c$ only.
- By using 5-fold Cross Validation (CV) with respect to hinge loss, we choose the optimal $c$ corresponding to $\lambda = 2^{-22}$. The black and brown curves in the plot below are the CV hinge respectively corresponding to the first and second c-step.
Illustration of Classification

True Bayes Rule

Bayes Rule from Second c-step
• Functional component pursuit is a way to improve the current applications of the reproducing kernel Hilbert space theory.
• Solution path methods can facilitate model fitting and tuning.
• Parametric LP algorithm sheds a new light on the solution path methods for variable and feature selection.
• Similar strategy can be applied based on parametric quadratic programming.
• Model selection issue remains to be addressed.
• New algorithms can be generated for quantile regression and Dantzig selector.
References


