Sparse Optimization

Juan C. De los Reyes

Centro de Modelización Matemática (MODEMAT)
Escuela Politécnica Nacional, Quito-Ecuador

CIMPA School: Mathematical Modelling and Numerical Simulation in Medicine
La Habana, 2023

Outline

Starting concept

Mode Mat

Priciple of parsimony-Ockham's razor

"Entities should not be multiplied unnecessarily"

One should not go looking for more complex explanations when there is a simpler one.

Starting concept

Priciple of parsimony-Ockham's razor

"Entities should not be multiplied unnecessarily"

One should not go looking for more complex explanations when there is a simpler one.

From the dictionary

Sparse: present only in small amounts; less than necessary or normal; thinly covering an area; not thick or full.

En castellano: disperso, esparcido, ralo.

Starting concept

Priciple of parsimony-Ockham's razor

"Entities should not be multiplied unnecessarily"

One should not go looking for more complex explanations when there is a simpler one.

From the dictionary

Sparse: present only in small amounts; less than necessary or normal; thinly covering an area; not thick or full.

En castellano: disperso, esparcido, ralo.

- There are several optimization problems were sparse solutions are required (e.g., in machine learning, data acquisition, image restoration, etc.)

Starting concept

Priciple of parsimony-Ockham's razor

"Entities should not be multiplied unnecessarily"

One should not go looking for more complex explanations when there is a simpler one.

From the dictionary

Sparse: present only in small amounts; less than necessary or normal; thinly covering an area; not thick or full.

En castellano: disperso, esparcido, ralo.

- There are several optimization problems were sparse solutions are required (e.g., in machine learning, data acquisition, image restoration, etc.)
- Recently, sparsity has also been considered in PDE constrained optimization problems.

Starting concept

Priciple of parsimony-Ockham's razor

"Entities should not be multiplied unnecessarily"

One should not go looking for more complex explanations when there is a simpler one.

From the dictionary

Sparse: present only in small amounts; less than necessary or normal; thinly covering an area; not thick or full.

En castellano: disperso, esparcido, ralo.

- There are several optimization problems were sparse solutions are required (e.g., in machine learning, data acquisition, image restoration, etc.)
- Recently, sparsity has also been considered in PDE constrained optimization problems.
- In recent years, a huge amount of new literature emerged on the subject.

Motivation

What does sparse optimization mean?

- many of the values of the decision variables are zero in case of vectors: solutions easy to interpret
- small support in case of functions: allows the localization of the action of the control

Motivation

What does sparse optimization mean?

- many of the values of the decision variables are zero in case of vectors: solutions easy to interpret
- small support in case of functions: allows the localization of the action of the control

Tools for dealing with such problems

- Large-scale optimization
- Nonsmooth optimization
- Application-specific knowledge

Outline

Linear regression

Classical linear regression model

A : matrix of individuals and features, i.e., $a_{i,j}$ is the value of attribute j of individual i.

u : is the decision vector with all the coefficients

y : is the dependent vector

Goal

Find the optimal coefficient vector $\bar{u} \in \mathbb{R}^n$ such that

$$\bar{u} = \arg\min_{u \in \mathbb{R}^n} \frac{1}{2} \|Au - y\|_2^2.$$

Linear regression

Example

Suppose we have a large database of clients with several $n \gg 1$ attributes (e.g., salary, age, years of education, number of shirts, etc.), and a dependent variable y (e.g., income). By minimizing the least squares cost

$$\|Au - y\|_2^2,$$

we get a coefficient vector $\bar{u} = (\bar{u}_1, \ldots, \bar{u}_n)$ that best fits the data. The vector acts also as predictor in case of new clients.

Linear regression
Example

Suppose we have a large database of clients with several $n \gg 1$ attributes (e.g., salary, age, years of education, number of shirts, etc.), and a dependent variable y (e.g., income). By minimizing the least squares cost

$$\|Au - y\|_2^2,$$

we get a coefficient vector $\bar{u} = (\bar{u}_1, \ldots, \bar{u}_n)$ that best fits the data. The vector acts also as predictor in case of new clients.

- How to predict the income of a new individual?
- Do we need to collect all $n \gg 1$ attribute information for the new clients?

Lasso

How to obtain a sparse predictor vector?

The idea consists in solving a least squares problem with an additional bound on an appropriate norm of the vector, i.e.,

$$\min_{u \in \mathbb{R}^n} \frac{1}{2} \|Au - y\|_2^2$$

$$\text{subject to: } \|u\|_0 \leq M$$

where $\|u\|_0$ counts the number of nonzero entries of u.

Problem of combinatorial nature

Lasso

How to obtain a sparse predictor vector?

The idea consists in solving a least squares problem with an additional bound on an appropriate norm of the vector, i.e.,

$$\min_{u \in \mathbb{R}^n} \frac{1}{2} \|Au - y\|_2^2$$

subject to: $\sout{\|u\|_0 \leq M}$ $\|u\|_1 \leq \epsilon,$

where $\|u\|_1 = \sum_{i=1}^{n} |u_i|$.

Lasso

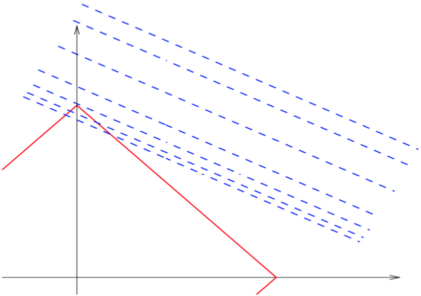
How to obtain a sparse predictor vector?

The idea consists in solving a least squares problem with an additional bound on an appropriate norm of the vector, i.e.,

$$\min_{u \in \mathbb{R}^n} \frac{1}{2} \|Au - y\|_2^2$$

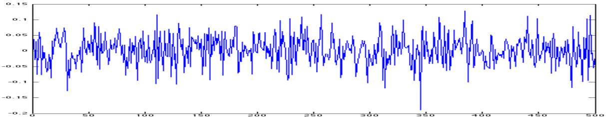
subject to: ~~$\|u\|_0 \leq M$~~ $\|u\|_1 \leq \epsilon,$

where $\|u\|_1 = \sum_{i=1}^{n} |u_i|$.



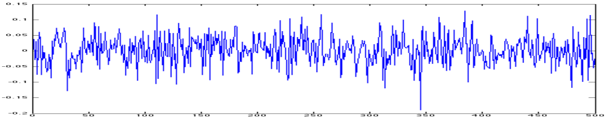
Lasso

We randomly generate an attribute matrix of size 1000×500 and a dependent variable y of length 1000. Solving (with MATLAB LSQLIN function) the classical least squares problem with get a full coefficient vector

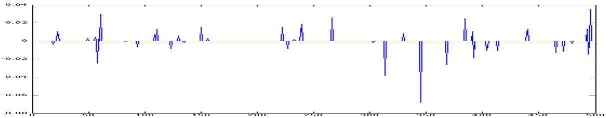


Lasso

We randomly generate an attribute matrix of size 1000×500 and a dependent variable y of length 1000. Solving (with MATLAB LSQLIN function) the classical least squares problem with get a full coefficient vector

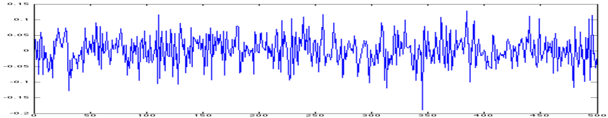


On the other hand, solving the Lasso problem, with a sparsity constraint, we get a sparse predictor

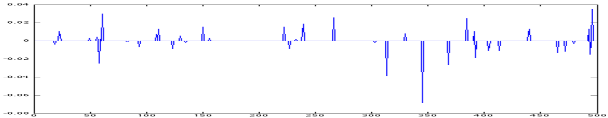


Lasso

We randomly generate an attribute matrix of size 1000×500 and a dependent variable y of length 1000. Solving (with MATLAB LSQLIN function) the classical least squares problem with get a full coefficient vector



On the other hand, solving the Lasso problem, with a sparsity constraint, we get a sparse predictor



Much less information has to be collected for a new individual in order to predict its behaviour.

Outline

Speech recognition

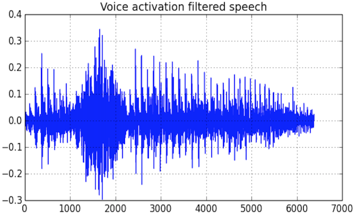
Mode
Mat

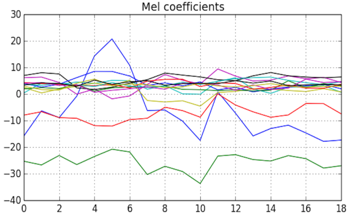
Training point

Vector of features for a 10ms frame of speech and a label representing the phonetic state.

Training point

Vector of features for a 10ms frame of speech and a label representing the phonetic state.

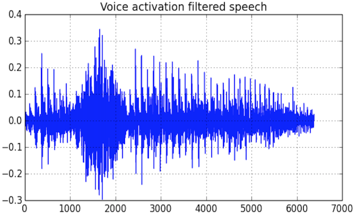


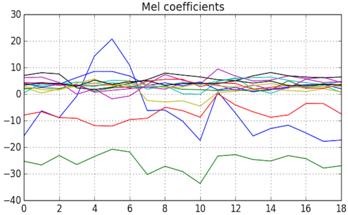


Speech recognition

Training point

Vector of features for a 10ms frame of speech and a label representing the phonetic state.





Goal

Maximize the conditional probability of the correct phonetic state, given an observed features vector.

- Simple logistic regression yields the probability of an event, given a prediction vector u:

$$p(y=1) = \frac{\exp\left(u^T a\right)}{1+\exp\left(u^T a\right)}$$

- Simple logistic regression yields the probability of an event, given a prediction vector u:

$$p(y=1) = \frac{\exp\left(u^T a\right)}{1+\exp\left(u^T a\right)}$$

- Speech recognition is based on multinomial logistic regression

$$p(y_j = k) = \frac{\exp u_k^T a_j}{\sum_{i=1}^{K} \exp u_i^T a_j}.$$

- Simple logistic regression yields the probability of an event, given a prediction vector u:

$$p(y=1) = \frac{\exp\left(u^T a\right)}{1+\exp\left(u^T a\right)}$$

- Speech recognition is based on multinomial logistic regression

$$p(y_j = k) = \frac{\exp u_k^T a_j}{\sum_{i=1}^{K} \exp u_i^T a_j}.$$

ModeMat

Optimization problem

$$\min_{u} \ j(u) = -\frac{1}{m}\sum_{j=1}^{m} \log \frac{\exp u_{y_j}^T z_j}{\sum_{i\in C} \exp u_i^T z_j} + \beta \|u\|_1$$

where:

C : set of labels
z_j : feature vector for point j
u_i : parameter subvector for class label i
m : number of training points
y_j : class label associated with data point j

Speech recognition

ModeMat

Optimization problem

$$\min_u j(u) = -\frac{1}{m} \sum_{j=1}^{m} \log \frac{\exp u_{y_j}^T z_j}{\sum_{i \in C} \exp u_i^T z_j} + \beta \|u\|_1$$

where:

C : set of labels
z_j : feature vector for point j
u_i : parameter subvector for class label i
m : number of training points
y_j : class label associated with data point j

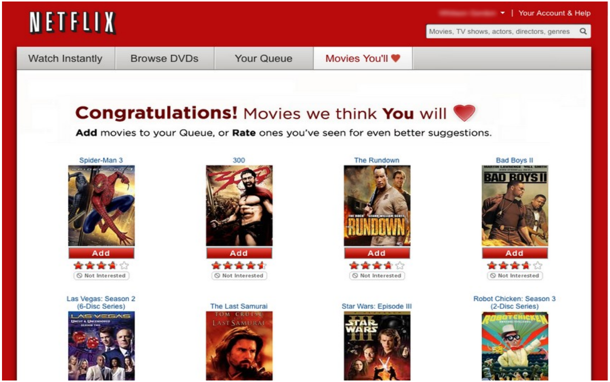
- Problems are usually of very large scale
- Subsampling is mandatory in this context
- Important to combine efficient optimization with stochastic approaches

Outline

Matrix completion

The Netflix Prize:

In 2006 Netflix offered a US$1,000,000 prize for an algorithm that substantially improves the accuracy of predictions about how much someone is going to enjoy a movie based on their movie preferences.

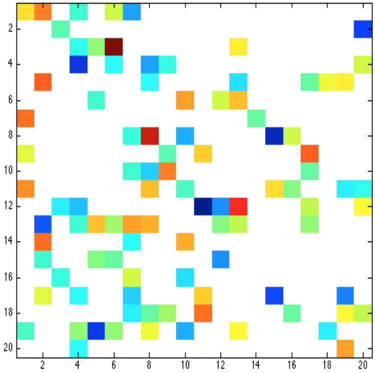


Matrix completion

Goal: fill the zero elements of a sparse matrix, based on the observed non-zero entries.

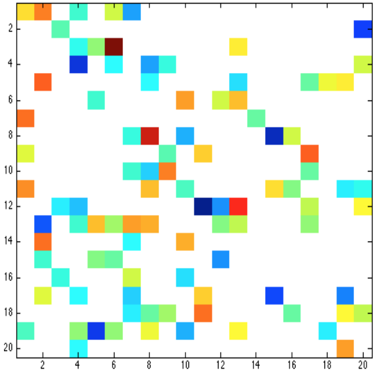
Matrix completion

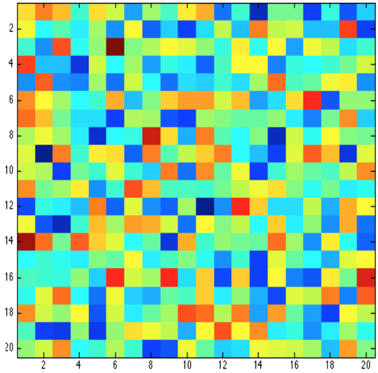
Goal: fill the zero elements of a sparse matrix, based on the observed non-zero entries.



Matrix completion

Goal: fill the zero elements of a sparse matrix, based on the observed non-zero entries.





Matrix completion

Hypothesis

- There are only few factors that determine the movie preferences of users.
- The observed non-zero entries of the matrix are uniformly distributed (at least one observation per row and one observation per column).

Mathematically, the problem can be stated in the following form:

$$\min_{X} \ \text{rank}(X)$$
$$\text{subject to: } X_{i,j} = M_{i,j}, \quad (i,j) \in \Omega,$$

with Ω the set of locations corresponding to observed entries.

Matrix completion

Hypothesis

- There are only few factors that determine the movie preferences of users.
- The observed non-zero entries of the matrix are uniformly distributed (at least one observation per row and one observation per column).

Mathematically, the problem can be stated in the following form:

$$\min_{X} \ \text{rank}(X)$$
$$\text{subject to: } X_{i,j} = M_{i,j}, \quad (i,j) \in \Omega,$$

with Ω the set of locations corresponding to observed entries.

Drawback: Any solution algorithm requires too much time to compute an exact solution.

Matrix completion

Important Property. If a matrix has rank r, then it has exactly r nonzero singular values.

Matrix completion

Important Property. If a matrix has rank r, then it has exactly r nonzero singular values.

Alternative idea

Instead of using the rank of X, one can consider the nuclear norm minimization, i.e.,

$$\min_X \|X\|_*$$
$$\text{subject to: } X_{i,j} = M_{i,j}, \quad (i,j) \in \Omega,$$

where $\|X\|_* = \sum_{k=1}^{n} \sigma_k(X)$, where $\sigma_k(X)$ is the k^{th} largest singular value of X.

Matrix completion

Important Property. If a matrix has rank r, then it has exactly r nonzero singular values.

Alternative idea

Instead of using the rank of X, one can consider the nuclear norm minimization, i.e.,

$$\min_X \|X\|_*$$
$$\text{subject to: } X_{i,j} = M_{i,j}, \quad (i,j) \in \Omega,$$

where $\|X\|_* = \sum_{k=1}^{n} \sigma_k(X)$, where $\sigma_k(X)$ is the k^{th} largest singular value of X.

Observation

The relation between $rank(X)$ and $\|X\|_*$ for matrices is similar to the relation between the l_0-norm and the l_1-norm for vectors.

A theoretical result

Theorem

Let M be an $n_1 \times n_2$ matrix of rank r and put $n = \max(n_1, n_2)$. Suppose we observe m entries of M with locations sampled uniformly at random. Then there are constants C and c such that if

$$m \geq Cn^{5/4}\ r\ \log n,$$

the minimizer to the matrix completion problem is unique and equal to M with probability at least $1 - cn^{-3}$, that is to say, the semidefinite program recovers all the entries of M with no error.

Emmanuel J. Candès, Benjamin Recht
Exact Matrix Completion via Convex Optimization. *Foundations of Computational Mathematics*. Volume 9, pp 717-772, 2009.

A theoretical result

Theorem

Let M be an $n_1 \times n_2$ matrix of rank r and put $n = \max(n_1, n_2)$. Suppose we observe m entries of M with locations sampled uniformly at random. Then there are constants C and c such that if

$$m \geq C n^{5/4} \; r \; \log n,$$

the minimizer to the matrix completion problem is unique and equal to M with probability at least $1 - cn^{-3}$, that is to say, the semidefinite program recovers all the entries of M with no error.

Emmanuel J. Candès, Benjamin Recht
Exact Matrix Completion via Convex Optimization. *Foundations of Computational Mathematics*. Volume 9, pp 717-772, 2009.

Solution methods. Semidefinite programming algorithms.

Outline

Sparse optimal control

Controlling population dynamics

$$\min J(y,u) = \varphi(y,u) + \frac{\lambda}{2}\|u\|_V^2 + \beta\|u\|_{L^1(\Omega)}$$

subject to :

$$\frac{\partial y(x,t)}{\partial t} - \nu\Delta y(x,t) = ry(x,t)\left(1 - \frac{y(x,t)}{\kappa}\right) - u(x)y(x,t)$$

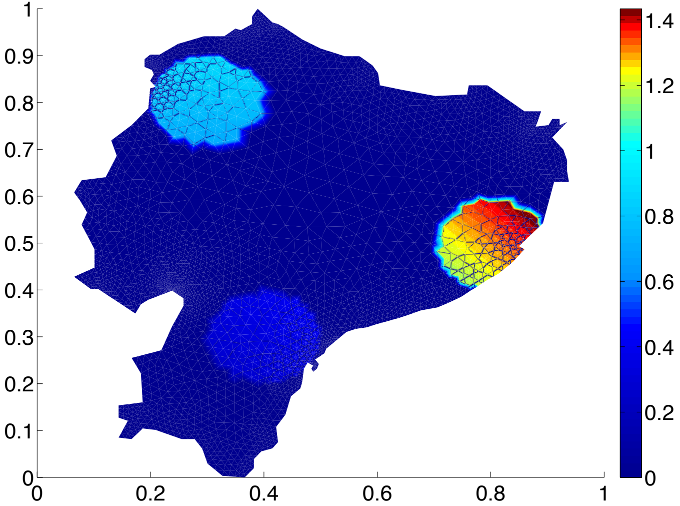
$+$ boundary conditions $+$ initial conditions

ν : diffusion parameter u : mortality rate to be controlled

r : growth rate κ : environmental capacity

φ represents the fumigation strategy.

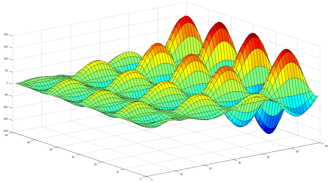
Localized fumigation

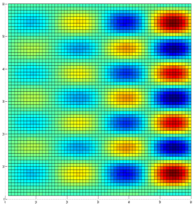


An optimal control example

L^2-term only

$$(P)\begin{cases} \min_{y,u} \frac{1}{2} \parallel y - y_d \parallel^2_{L^2(\Omega)} + \frac{\lambda}{2} \parallel u \parallel^2_{L^2(\Omega)} \\ \text{s.t.} \\ \qquad -\Delta y = u + f \quad \text{in } \Omega \\ \qquad y = 0 \quad \text{on } \Gamma \end{cases}$$

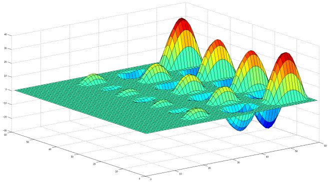


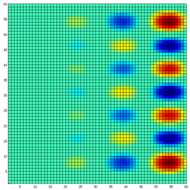


An optimal control example

With additional L^1-term

$$
(P)\begin{cases}
\min\limits_{y,u} \dfrac{1}{2} \| y - y_d \|^2_{L^2(\Omega)} + \dfrac{\lambda}{2} \| u \|^2_{L^2(\Omega)} + \beta \| u \|_{L^1(\Omega)} \\
\text{s.t.} \\
\qquad -\Delta y = u + f \quad \text{in } \Omega \\
\qquad y = 0 \quad \text{on } \Gamma
\end{cases}
$$





Outline

Outline

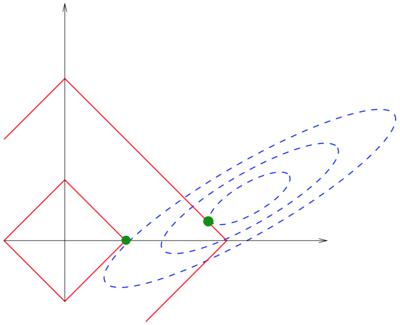
Outline

Lasso revisited

Why does it work?

$$\min_{u \in \mathbb{R}^n} \frac{1}{2}\|Au - y\|_2^2$$

subject to: $\|u\|_1 \leq \epsilon.$



Lasso revisited

Alternative formulations

- As unconstrained problem:

$$\min_{u \in \mathbb{R}^n} \frac{1}{2}\|Au - y\|_2^2 + \beta\|u\|_1 \qquad (1)$$

- With the least squares term as constraint:

$$\min_{u \in \mathbb{R}^n} \|u\|_1$$

$$\text{subject to: } \|Au - y\|_2 \leq \epsilon$$

Lasso revisited

Alternative formulations

- As unconstrained problem:

$$\min_{u\in\mathbb{R}^n} \frac{1}{2}\|Au-y\|_2^2 + \beta\|u\|_1 \tag{1}$$

- With the least squares term as constraint:

$$\min_{u\in\mathbb{R}^n} \|u\|_1$$
$$\text{subject to: } \|Au-y\|_2 \leq \epsilon$$

We focus on unconstrained optimization problems like (1)

Outline

Abstract result

Let J be a convex function and consider the optimization problem

$$\min_{u} J(u)$$

Abstract result

Let J be a convex function and consider the optimization problem

$$\min_u J(u)$$

Defining the subdifferential by

$$\partial J(u) := \{\phi \in \mathbb{R}^m : \phi^T(v-u) \leq J(v) - J(u)\}$$

we obtain the following general result.

Theorem

For any convex function $J : \mathbb{R}^n \to \mathbb{R}$, if a point $\bar{u} \in \mathbb{R}^n$ is a global minimum of J if and only if $0 \in \partial J(\bar{u})$ holds.

Abstract result

Let J be a convex function and consider the optimization problem

$$\min_u J(u)$$

Defining the subdifferential by

$$\partial J(u) := \{\phi \in \mathbb{R}^m : \phi^T(v-u) \leq J(v) - J(u)\}$$

we obtain the following general result.

Theorem

For any convex function $J : \mathbb{R}^n \to \mathbb{R}$, if a point $\bar{u} \in \mathbb{R}^n$ is a global minimum of J if and only if $0 \in \partial J(\bar{u})$ holds.

If g is differentiable, then $\partial J(u) = \{\nabla J(u)\}$.

Optimization problem

More structure

We focus on the unconstrained optimization problem:

$$\min_{u\in\mathbb{R}^n} J(u) = f(u) + \beta\|u\|_1 \tag{P}$$

where f is convex and differentiable.

Theorem

Let $j_1 : \mathbb{R}^n \to \mathbb{R}$ be differentiable and $j_2 : \mathbb{R}^n \to \mathbb{R}$ convex and continuous. If $\bar{u} \in \mathbb{R}^n$ is an optimal solution to

$$\min_{u\in U} j_1(u) + j_2(u),$$

then it satisfies the following optimality condition:

$$j_1'(\bar{u})(v-\bar{u}) + j_2(v) - j_2(\bar{u}) \geq 0, \text{ for all } v \in \mathbb{R}^n.$$

Proof

j_1 convex and differentiable, j_2 convex continuous

$$j_1(\bar{u}) + j_2(\bar{u}) \leq j_1(w) + j_2(w), \forall w$$

Taking $w = \bar{u} + t(v - \bar{u})$, $0 < t \leq 1$,

$$\begin{aligned} 0 &\leq j_1(\bar{u} + t(v - \bar{u})) - j_1(\bar{u}) + j_2(\bar{u} + t(v - \bar{u})) - j_2(\bar{u}) \\ &\leq j_1(\bar{u} + t(v - \bar{u})) - j_1(\bar{u}) + t\, j_2(v) + (1 - t)\, j_2(\bar{u}) - j_2(\bar{u}) \end{aligned}$$

Dividing by t and taking the limit:

$$\begin{aligned} &0 \leq \frac{j_1(\bar{u} + t(v - \bar{u})) - j_1(\bar{u})}{t} + j_2(v) - j_2(\bar{u}) \\ \Longrightarrow\ &0 \leq j_1'(\bar{u})(v - \bar{u}) + j_2(v) - j_2(\bar{u}). \end{aligned}$$

Optimality condition

Problem

$$\min_{u \in \mathbb{R}^n} J(u) = f(u) + \beta \|u\|_1 \tag{P}$$

The optimality condition is given by:

$$\nabla f(\bar{u})^T (v - \bar{u}) + \beta \|v\|_1 - \beta \|\bar{u}\|_1 \geq 0, \text{ for all } v \in \mathbb{R}^n,$$

which can be reformulated as

$$-\nabla f(\bar{u}) \in \partial \beta \|\bar{u}\|_1$$

or, equivalently,

$$\begin{aligned}
&\nabla_i f(\bar{u}) + \beta = 0 && \text{if } \bar{u}_i > 0 \\
&\nabla_i f(\bar{u}) - \beta = 0 && \text{if } \bar{u}_i < 0 \\
&0 \in [\nabla_i f(\bar{u}) - \beta, \nabla_i f(\bar{u}) + \beta] && \text{if } \bar{u}_i = 0
\end{aligned}$$

Example

Consider the one dimensional problem

$$\min_{u \in \mathbb{R}} \frac{1}{2}(y-u)^2 + \beta|u|.$$

Since the subgradient of the absolute value function is

$$\partial|u| = \begin{cases} 1 & \text{if } u > 0 \\ [-1,1] & \text{if } u = 0 \\ -1 & \text{if } u < 0, \end{cases}$$

the solution of the problem is given by

$$\bar{u} = \begin{cases} 0 & \text{if } |y| \leq \beta \\ \left(1 - \frac{\beta}{|y|}\right) y & \text{otherwise.} \end{cases}$$

The last operator is called *soft-thresholding*.

Outline

Our problem may be written in the general form:

$$\inf_{u \in V} \mathcal{F}(u) + \mathcal{G}(\Lambda u),$$

where $\mathcal{F} : V = \mathbb{R}^n \to \mathbb{R}$, $\mathcal{G} : Y = \mathbb{R}^n \to \mathbb{R}$ and $\Lambda \in \mathcal{L}(V, Y)$.
Defining the conjugate of a function $h : V \to \mathbb{R}$ by

$$h^*(v^*) = \sup_{v \in V}\{\langle v^*, v \rangle - h(v)\},$$

which is convex function. The dual problem is then given by:

$$\sup_{q^* \in \mathbb{R}^n} -\mathcal{F}^*(-\Lambda^* q^*) - \mathcal{G}^*(q^*),$$

where Λ^* is the adjoint operator of Λ.

Mode Mat

Theorem

Let $\bar{u}$ and $\bar{q}$ be the optimal solutions to the primal and dual problem, respectively. Then there is no duality gap, i.e.,

$$\mathcal{F}(\bar{u}) + \mathcal{G}(\Lambda \bar{u}) = -\mathcal{F}^*(-\Lambda^* q^*) + \mathcal{G}^*(q^*)$$

and both solutions satisfy the following extremality conditions:

$$\begin{aligned} -\Lambda^* \bar{q} &\in \partial \mathcal{F}(\bar{u}) \\ -\bar{q} &\in \partial \mathcal{G}(\Lambda \bar{u}). \end{aligned}$$

The extremality conditions are necessary and sufficient.

Duality for Lasso

Considering the Lasso problem

$$\min_{u \in \mathbb{R}^n} \frac{1}{2}\|Au - y\|_2^2 + \beta\|u\|_1$$

the Fenchel dual problem is given by

$$\min_{q \in \mathbb{R}^n} -\frac{1}{2}\|Au - y\|_2^2 - (q, u)$$

subject to:

$$A^T(Au - y) + q = 0$$

$$|q_i| \leq \beta, \ \forall i$$

Duality for Lasso

Considering the Lasso problem

$$\min_{u \in \mathbb{R}^n} \frac{1}{2}\|Au - y\|_2^2 + \beta\|u\|_1$$

the Fenchel dual problem is given by

$$\min_{q \in \mathbb{R}^n} -\frac{1}{2}\|Au - y\|_2^2 - (q, u)$$

subject to:

$$A^T(Au - y) + q = 0$$

$$|q_i| \leq \beta, \ \forall i$$

and the optimality system by

$$\begin{aligned} & A^T(Au - y) + q^* = 0 & \\ & |q_i^*| \leq \beta & \forall i = 1, \dots, n \\ & q_i^* \, \bar{u}_i = \beta|\bar{u}_i| & \forall i = 1, \dots, n. \end{aligned}$$

Duality for Lasso

By defining the auxiliary dual multiplier

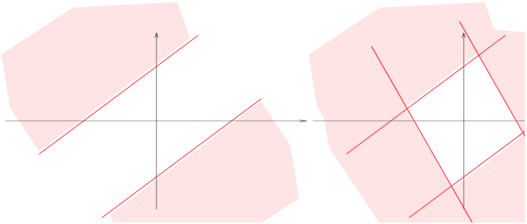
$$\bar{q} := y - Au$$

the dual problem can be rewritten as

$$\min_{q \in \mathbb{R}^m} \|q - y\|_2^2$$

$$\text{subject to: } |A^T q| \leq \beta.$$

- The number of active faces of the constraint set corresponds to the number of nonzero entries of u.



Optimality system

The optimality system for our case is given by

$$
\begin{aligned}
&A\bar{u} - y + \bar{q} = 0 && \\
&|(A^T\bar{q})_i| \leq \beta && \forall i = 1, \ldots, n \\
&(A^T\bar{q})_i \, \bar{u}_i = \beta|\bar{u}_i| && \forall i = 1, \ldots, n.
\end{aligned}
$$

where $\bar{q}$ is the dual solution.

Dual information

The dual problem and the resulting optimality system provide important information, which may be of use for the design of solution algorithms.

Outline

Outline

ModeMat

Steespest descent

Let us consider the following minimization problem:

$$\min_{u \in \mathbb{R}^n} f(u),$$

with f continuously differentiable.

The main idea of descent methods consists in finding, at a given iterate u_k, a descent direction g_k, i.e,

$$f(u_k + \alpha_k g_k) < f(u_k) \quad \text{with} \quad \alpha_k > 0.$$

Steepest descent

The most natural choice would be to pick as direction the one that leads to the maximum descent of the objective function (locally), i.e, the one that solves the problem

$$\min_{||g||=1} \nabla f(u)^\top g \quad \text{minimization of the linear model of } f$$

Theorem

Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable and $u \in \mathbb{R}^n$ such that $\nabla f(u) \neq 0$. Then the optimization problem has a unique solution given by

$$g = -\frac{\nabla f(u)}{\|\nabla f(u)\|}$$

Consequently, any direction of the form

$$g_k = -\alpha_k \nabla f(u_k), \quad \alpha_k > 0 \tag{2}$$

corresponds to a ”steepest descent” direction.

Line search

Once the descent direction is determined, is important to know how far to move in such direction, i.e, which parameter $\alpha_k > 0$ should be used. The ideal choice would be

$$\alpha_k = \arg\min_{\alpha>0} f(u_k + \alpha g_k)$$

Line search

Once the descent direction is determined, is important to know how far to move in such direction, i.e, which parameter $\alpha_k > 0$ should be used. The ideal choice would be

$$\alpha_k = \arg\min_{\alpha > 0} f(u_k + \alpha g_k)$$

This is, however, not possible in practice!

Line search

Once the descent direction is determined, is important to know how far to move in such direction, i.e, which parameter $\alpha_k > 0$ should be used. The ideal choice would be

$$\alpha_k = \arg\min_{\alpha > 0} f(u_k + \alpha g_k)$$

This is, however, not possible in practice!

In general the following *feasibility* condition is required to get convergence:

$$f(u_k + \alpha_k g_k) - f(u_k) \xrightarrow[k \to \infty]{} 0 \Longrightarrow \frac{\nabla f(u_k)^\top g_k}{||g_k||} \xrightarrow[k \to \infty]{} 0$$

Armijo's line search rule

A popular line search strategy is the Armijo rule, which consists in the following: given a descent direction g_k of f at u_k, choose $\alpha_k \in \{1, \frac{1}{2}, \frac{1}{4} \cdots\}$ such that

$$f(u_k + \alpha_k g_k) - f(x_k) \leq \gamma \alpha_k \nabla f(u_k)^\top g_k,$$

where $\gamma \in (0,1)$ is a given constant.

- There exists an interval of feasible steps.
- Armijo's rule satisfies the feasibility condition.

Sketch

Outline

Subgradient descent

Given the optimization problem

$$\min_u J(u),$$

with J convex, the main idea of subgradient methods consists in choosing an element of the subgradient to construct a direction in which to advance in order to improve the cost function value.

Subgradient descent

Given the optimization problem

$$\min_u J(u),$$

with J convex, the main idea of subgradient methods consists in choosing an element of the subgradient to construct a direction in which to advance in order to improve the cost function value.

The iterations for the sparse optimization are given by

$$u_{k+1} = u_k - \alpha_k \underbrace{\left(\nabla f(u_k) + \beta s\right)}_{=: g_k}, \text{ with } s \in \partial \|u_k\|_1$$

Subgradient descent

Given the optimization problem

$$\min_u J(u),$$

with J convex, the main idea of subgradient methods consists in choosing an element of the subgradient to construct a direction in which to advance in order to improve the cost function value.

The iterations for the sparse optimization are given by

$$u_{k+1} = u_k - \alpha_k \underbrace{(\nabla f(u_k) + \beta s)}_{=:g_k}, \text{ with } s \in \partial \|u_k\|_1$$

Historical note

Subgradient methods were developed in the 60's and 70's.

N. Z. Shor.
Minimization Methods for Non-differentiable Functions. Springer Verlag, 1985.

Subgradient descent

Line search rules

- Constant step size: $\alpha_k = \alpha$, constant independent of k.
- Constant step length: $\alpha_k = \frac{\alpha}{\|g_k\|_2}$
- Square summable but not summable:

$$\sum_{k=1}^{\infty} \alpha_k^2 < \infty \quad \sum_{k=1}^{\infty} \alpha_k = \infty.$$

  A prototypical example is $\alpha_k = \frac{\alpha}{k}$.
- Nonsummable diminishing:

$$\lim_{k \to \infty} \alpha_k = 0 \quad \sum_{k=1}^{\infty} \alpha_k = \infty.$$

  A prototypical example is $\alpha_k = \frac{\alpha}{\sqrt{k}}$.

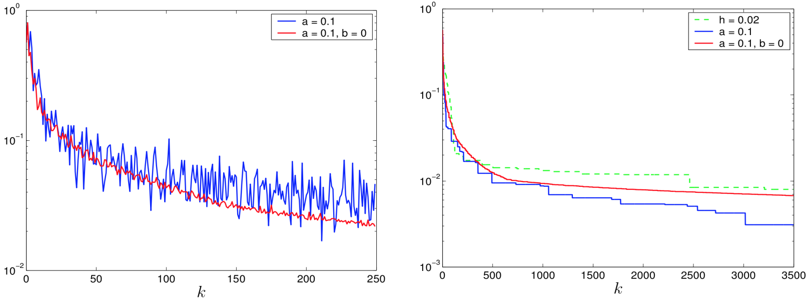
Subgradient descent

Computational results

Numerical results for

$$\min_u \left[\max_{i=1,\ldots,m} (a_i^T u + b_i) \right],$$

with different line search rules.



Subgradient descent

Properties

- Unlike the steepest descent method, there is no guaranteed descent at each iteration.
- The iterates converge globally with

$$J(u_k) - J(\bar{u}) = O(\frac{1}{\sqrt{k}})$$

- Usually convergence is very slow
- The problem structure is not exploited

Outline

Iterative Shrinkage-Thresholding Algorithm (ISTA)

$$\min_{u \in \mathbb{R}^n} J(u) = f(u) + \beta \|u\|_1 \tag{P}$$

An important operator is the so called proximal operator defined for a convex function $J : \mathbb{R}^n \to \mathbb{R}$

$$Prox_J(v) = \arg\min_u \left\{ J(u) + \frac{1}{2}\|u - v\|_2^2 \right\}$$

Basic idea

Solve at each iteration the linearized problem

$$\min_u f(u_k) + \nabla f(u_k)^T (u - u_k) + \beta \|u\|_1 + \frac{L}{2}\|u - u_k\|_2^2,$$

or, equivalently,

$$\min_u \frac{1}{2}\|u - (u_k - \frac{1}{L}\nabla f(u_k))\|_2^2 + \frac{\beta}{L}\|u\|_1 \tag{MinProx}$$

where $L > 0$ is an upper bound for ∇f (usually unknown).

Line search for L

Increase the value of L until

$$f(u_L) \leq f(u_k) + \nabla f(u_k)^T(u_L - u_k) + \frac{L}{2}\|u_L - u_k\|_2^2$$

where u_L is the solution of (MinProx).

Some properties

- The method converges globally with a rate of $O(\frac{1}{k})$.
- There are accelerated versions of the proximal algorithm with convergence rate $O(\frac{1}{k^2})$.
- Accelerated version do not necessarily lead to descent directions.

Proximal methods

Proximal operator

The efficiency of proximal methods depends on the fast computation of the proximal operator

$$Prox_{\beta\|\cdot\|_1}(w) = \arg\min_u \left\{ \frac{1}{2}\|w-u\|_2^2 + \beta\|u\|_1 \right\},$$

since the iteration is given by

$$u_{k+1} = Prox_{\frac{\beta}{L}\|\cdot\|_1}(u_k - \frac{1}{L}\nabla f(u_k)).$$

Thanks to the optimality conditions, the proximal operator can be computed through

$$\left[Prox_{\beta\|\cdot\|_1}(w)\right]_j = \left(1 - \frac{\beta}{|w_j|}\right)_+ w_j,$$

where $(x)_+ := \max(0, x)$.
Componentwise, the proximal operator is the soft-thresholding operator.

Fast Iterative Shrinkage-Thresholding Algorithm
(FISTA)

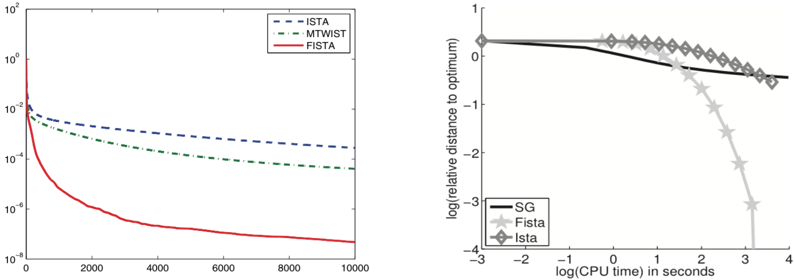
The fast version of the Iterative Shrinkage-Thresholding Algorithm consists in choosing, instead of the previous iterate u_k, a clever linear combination of the previous two iterates.

Initialize u_0, $t_0 = 1$ and $u_1 = Prox_{\frac{\beta}{L}\|\cdot\|_1}(u_0 - \frac{1}{L}\nabla f(u_0))$.
while stoping criteria is false **do**
  Compute $t_k = \frac{1+\sqrt{1+4t_{k-1}^2}}{2}$.
  Compute $y_k = u_{k-1} - \left(\frac{1-t_{k-1}}{t_k}\right)(u_k - u_{k-1})$
  Update $u_{k+1} = Prox_{\frac{\beta}{L}\|\cdot\|_1}(y_k - \frac{1}{L}\nabla f(y_k))$.
  $k \leftarrow k+1$.
end while

Properties

- The method arised from the complexity analysis of ISTA.
- While ISTA has convergence of order $O(k^{-1})$, FISTA has convergence rate of order $O(k^{-2})$.

A. Beck, M- Teboulle.
A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems
SIAM J. Imaging Sciences, Vol. 2, pp. 183-202, 2009.



Outline

Coordinate descent method

Lasso

By selecting a coordinate j, this method is based on the sequential coordinate-wise solution of

$$\min_{u_j} \nabla_j f(u^k)(u_j - u_j^k) + \frac{1}{2}\nabla^2_{jj} f(u^k)(u_j - u_j^k)^2 + \beta |u_j|$$

where $\nabla_j f(u) = A_j^T(Au - y)$ and $\nabla^2_{jj} f(u) = A_j^T A_j$.

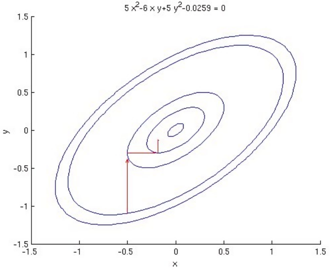


Figure: Coordinate descend iterations

Coordinate descent method

Lasso

By means of the proximal operator with $L = \nabla^2_{jj} f(u^k)$, the solution can be expressed in close form as

$$u_j^* = Prox_{\frac{\beta}{L}|\cdot|}\left(u_j^k - \frac{\nabla_j f(u_j^k)}{\nabla^2_{jj} f(u^k)}\right),$$

i.e., u_j^* is obtained by solving the unregularized problem with respect to coordinate j and soft-thresholding it.

If f is not a least squares term, the solution has not a direct closed form. However, we can still compute the solution to the quadratic model

$$u_j^* = \arg\min_{u_j} \nabla_j f(u^k)(u_j - u_j^k) + \frac{1}{2}\nabla_{jj}^2 f(u^k)(u_j - u_j^k)^2 + \beta|u_j|$$

and combine it with an Armijo line search: Choose $\alpha \in (0,1)$ such that

$$J(u^k + \alpha d e_j) - J(u^k) \leq \sigma\alpha(\nabla_j f(u^k)d + |u_j^k + d| - |u_j^k|)$$

where $\sigma > 0$ and $d = u_j^* - u_j^k$.

Coordinate descent method

Basic algorithm

Initialize u_0,

while stoping criteria is false **do**

  CHOSE $j \in \{1, 2, \ldots, n\}$

  COMPUTE $u_j^* = Prox_{\frac{\beta}{L}|\cdot|}\left(u_j^k - \frac{\nabla_j f(u_j^k)}{\nabla_{jj}^2 f(u^k)}\right)$,

  UPDATE $u^{k+1} = u^k + (u_j^* - u_j^k)e_j$, for some $\alpha_k \in (0, 1)$

  $k \leftarrow k + 1$.

end while

Coordinate descent method

Choosing coordinates

In this framework, there is a lot of freedom in choosing the index j.

- Cyclic fashion coordinates: $i_0 = 1$,

  $$i_k + 1 = (i_k \mod n) + 1, \quad k = 0, 1, 2, \ldots$$

  Every $T \geq n$ iterations each component must be modified at least once: $\cup_{j=0}^{T} i_k - j = 1, 2, \ldots, n$
- Randomized coordinates : not necessarily with equal probability. For example, i_k is choosen with uniform probability in the set $\{1, 2, \ldots, n\}$, independent of the choices of previous iterations.

Convergence result for randomized CDM for Lasso

Assumptions and notations

- f is strongly convex and Lipschitz continuously differentiable

$$f(\alpha u + (1-\alpha)v) \leq \alpha f(u) + (1-\alpha)f(v) - \frac{1}{2}\sigma\alpha(1-\alpha)\left\|u - v\right\|_2^2, \forall\, u, v$$

  if f is twice continuously differentiable, f is strongly convex iff $\nabla^2 f(u)$ is positive definite for all u

- (Componentwise Lipschitz constants) $\forall i$, $\exists L_i$ such that

$$|\nabla_i f(u + te_i) - \nabla_i f(u)| \leq L_i|t|, \quad \forall u, \forall t \in \mathbb{R}$$

$$L_{max} = \max_i L_i$$

CDM for Lasso

$$\min_{u_j} \nabla_j f(u^k)(u_j - u_j^k) + \frac{1}{2\alpha_k}(u_j - u_j^k)^2 + \beta|u_j|$$

1: Initialize u_0,
2: **while** stoping criteria is false **do**
3: CHOSE $i_k \in \{1, 2, \ldots, n\}$
4: COMPUTE $u_{i_k}^* = \arg\min_u (u - u_{i_k}^k)\nabla_i f(u^k) + \frac{1}{2\alpha_k}(u_{i_k} - u^*)^2 + \beta|u_{i_k}|$, for some $\alpha_k \in (0, 1)$
5: UPDATE $u^{k+1} = u^k + (u_{i_k}^* - u_{i_k}^k)e_{i_k}$,
6: $k \leftarrow k + 1$.
7: **end while**

Convergence result for randomized CDM for Lasso

Theorem

With the last assumptions at hand, let us suppose that the coordinate index i_k in CDM-Algorithm are chosen independently for each k with uniform probability from the set $\{1, 2, \ldots, n\}$, and that $\alpha_k = 1/L_{max}$. Then for all $k \geq 0$, we have

$$E\left(J(u^k)\right) - J(u^*) \leq \left(1 - \frac{\sigma}{nL_{max}}\right)^k \left(J(u^0) - J(u^*)\right)$$

Outline

Mode Mat

Let us consider the optimization problem

$$\min_{u\in\Omega} \; J(u),$$

with $\Omega := \{v \in \mathbb{R}^n : a_i \leq v_i \leq b_i\}$ and f continuously differentiable. The optimality condition is then given by

$$\nabla J(\bar{u})^T(v - \bar{u}) \geq 0, \quad \forall v \in \Omega$$

or, equivalently, as

$$\bar{u} = \mathcal{P}(\bar{u} - \lambda \nabla J(\bar{u})), \quad \forall \lambda > 0$$

where $\mathcal{P}(u)_i = \min(\max(u_i, a_i), b_i)$.

Projected gradient

Nonlinear programming

The idea of the projected gradient method consists in using the optimality condition iteratively:

$$u_{k+1} = \mathcal{P}(u_k - \alpha_k \nabla J(u_k)),$$

where $\alpha_k > 0$ is a line search parameter. Sketch
The line-search parameter is chosen according to the projected Armijo rule: choose the largest $\alpha_k \in \{1, \frac{1}{2}, \frac{1}{4}, \dots\}$ for which

$$J(\mathcal{P}(u_k - \alpha_k \nabla J(u_k))) - J(u_k) \leq -\frac{\sigma}{\alpha_k} \|\mathcal{P}(u_k - \alpha_k \nabla J(u_k)) - u_k\|^2,$$

where $\sigma \in (0,1)$ is a given constant.

Accelerated projection methods

The application of projection methods considering other type of directions $d_k = -H_k^{-1}\nabla J(u_k)$ is by no means standard. For Newton directions

$$d_k = -(\nabla^2 J(u_k))^{-1}\nabla J(u_k),$$

for instance, the application of the projected method may not lead to descent in the objective function. To solve this problem, the reduced Hessian

$$(\nabla_R^2 J(u))_{ij} = \begin{cases} \delta_{ij} & \text{if } i \in A(u) \text{ or } j \in A(u) \\ (\nabla^2 J(u))_{ij} & \text{otherwise} \end{cases}$$

where $A(u)$ denotes the set of active indexes, may be used instead of the full second order matrix.

Reformulation of Lasso

By using the decomposition

$$u = u^+ - u^-$$

with $u^+ = \max(0, u)$ and $u^- = |\min(0, u)|$ we obtain the equivalent Lasso optimization problem:

$$\min_{u^+ \geq 0, u^- \geq 0} J(u^+, u^-) = \frac{1}{2}\|A(u^+ - u^-) - y\|_2^2 + \beta 1^t u^+ + \beta 1^t u^-$$

The gradient of the function is given by

$$\begin{pmatrix} \nabla_{u^+} J(u^+, u^-) \\ \nabla_{u^-} J(u^+, u^-) \end{pmatrix} = \begin{pmatrix} A^T A(u^+ - u^-) - A^T y + \beta 1 \\ -A^T A(u^+ - u^-) + A^T y + \beta 1 \end{pmatrix}$$

and the projected iteration is given by

$$\begin{pmatrix} u_{k+1}^+ \\ u_{k+1}^- \end{pmatrix} = \mathcal{P} \begin{pmatrix} u_k^+ - \alpha \nabla_{u^+} J(u^+, u^-) \\ u_k^+ - \alpha \nabla_{u^-} J(u^+, u^-) \end{pmatrix}$$

where $\mathcal{P}(y) := \max(0, y)$.

Summary of projection methods

Nonlinear programming

ModeMat

Properties

- Several developed nonlinear programming toolboxes can be used.
- For directions different from the projected descent, some effort has to be inverted in the construction of the Hessian approximation.

Drawbacks

- The number of optimization variables doubles, causing memory problems, as well as slowing convergence of all available toolboxes.
- The specific structure of the problem is not taken into account.

Outline

Outline

Mode Mat

Reformulation of optimality system

$$
\begin{aligned}
0 &= \nabla_i f(\bar{u}) + \beta && \text{for } i \in \bar{\mathcal{P}}, \\
0 &= \nabla_i f(\bar{u}) - \beta && \text{for } i \in \bar{\mathcal{N}}, \\
0 &\in [\nabla_i f(\bar{u}) - \beta, \nabla_i f(\bar{u}) + \beta] && \text{for } i \in \bar{\mathcal{A}},
\end{aligned}
$$

where the index sets $\bar{\mathcal{P}}$, $\bar{\mathcal{N}}$ and $\bar{\mathcal{A}}$ are defined as

$$
\bar{\mathcal{P}} = \{i : \bar{u}_i > 0\}, \quad \bar{\mathcal{N}} = \{i : \bar{u}_i < 0\}, \quad \text{and } \bar{\mathcal{A}} = \{i : \bar{u}_i = 0\}.
$$

Reformulation of optimality system

$$
\begin{aligned}
0 &= \nabla_i f(\bar{u}) + \beta && \text{for } i \in \bar{\mathcal{P}},\\
0 &= \nabla_i f(\bar{u}) - \beta && \text{for } i \in \bar{\mathcal{N}},\\
0 &\in [\nabla_i f(\bar{u}) - \beta, \nabla_i f(\bar{u}) + \beta] && \text{for } i \in \bar{\mathcal{A}},
\end{aligned}
$$

where the index sets $\bar{\mathcal{P}}$, $\bar{\mathcal{N}}$ and $\bar{\mathcal{A}}$ are defined as

$$\bar{\mathcal{P}} = \{i : \bar{u}_i > 0\}, \quad \bar{\mathcal{N}} = \{i : \bar{u}_i < 0\}, \quad \text{and } \bar{\mathcal{A}} = \{i : \bar{u}_i = 0\}.$$

The system can be equivalently written as $F(u) = 0$, with

$$F_i(u) = \max\left\{\min\{\tau(\nabla_i f(u) + \beta), u_i\}, \tau(\nabla_i f(u) - \beta)\right\},$$

where τ is any positive constant.

How to solve the system efficiently?

Semismooth Newton method

Definition (Newton differentiability)

If there exists a neighborhood $N(\bar{u}) \subset S$ and a family of mappings $G : N(\bar{u}) \to \mathcal{L}(X, Y)$ such that

$$\lim_{\|h\|_X \to 0} \frac{\|\mathcal{F}(\bar{u} + h) - \mathcal{F}(\bar{u}) - G(\bar{u} + h)(h)\|_Y}{\|h\|_X} = 0,$$

then $\mathcal{F}$ is called Newton differentiable at $\bar{u}$.

Semismooth Newton method

Mode Mat

Definition (Newton differentiability)

If there exists a neighborhood $N(\bar{u}) \subset S$ and a family of mappings $G : N(\bar{u}) \to \mathcal{L}(X, Y)$ such that

$$\lim_{\|h\|_X \to 0} \frac{\|\mathcal{F}(\bar{u}+h) - \mathcal{F}(\bar{u}) - G(\bar{u}+h)(h)\|_Y}{\|h\|_X} = 0,$$

then $\mathcal{F}$ is called Newton differentiable at $\bar{u}$.

Semi-smooth Newton step

If F is Newton differentiable, a Newton type update can be obtained as

$$G(u^k)d = -F(u^k), \qquad u^{k+1} = u^k + d,$$

where G stands for the generalized Jacobian of F.

Consider the absolute value function

$$f(x) = |x|$$

The function is not differentiable at 0. However, by using the generalized derivative

$$g(x) = \begin{cases} -1 & \text{if } x < 0, \\ 1 & \text{if } x \geq 0. \end{cases}$$

we obtain for the case $x = 0$:

1. if $h > 0$: $\quad \big||x+h| - |x| - |h|\big| = 0,$
2. if $h < 0$: $\quad \big||x+h| - |x| + |h|\big| = |-x-h-x+h| = 0.$

Consequently,

$$\lim_{h \to 0} \frac{1}{|h|} |f(x+h) - f(x) - g(x+h)h| = 0$$

and $|\cdot|$ is Newton differentiable.

Superlinear convergence

Theorem

Let $\bar{x}$ be a solution to $F(\bar{x}) = 0$, with F Newton differentiable in an open neighbourhood V containing $\bar{x}$. If

$$\|G(x)^{-1}\|_{\mathcal{L}(Z,X)} \leq C,$$

for some constant $C > 0$ and all $x \in V$, then the Newton iteration

$$x_{k+1} = x_k - G(x_k)^{-1}F(x_k)$$

converges superlinearly to $\bar{x}$ provided that $\|x_0 - \bar{x}\|_X$ is sufficiently small.

Differentiability of the *max* function

Mode Mat

The mapping $y \mapsto max(0, y)$ from $\mathbb{R}^n \to \mathbb{R}^n$ with

$$g(y) = \begin{cases} 1 \text{ if } y \geq 0 \\ 0 \text{ if } y < 0 \end{cases}$$

as generalized derivative, is Newton differentiable.

Green light for solving the system

$$F_i(u) = \max\{\min\{\tau(\nabla_i f(u) + \beta), u_i\}, \tau(\nabla_i f(u) - \beta)\} = 0, \forall i$$

with a generalized Newton method.

By defining the following index sets:

$$
\begin{aligned}
\mathcal{N}^k &:= \left\{i \colon u_i^k \le \tau\left(\nabla_i f(u^k) - \beta\right)\right\},\\
\mathcal{A}^k &:= \left\{i \colon \tau\left(\nabla_i f(u^k) - \beta\right) \le u_i^k \le \tau\left(\nabla_i f(u^k) + \beta\right)\right\},\\
\mathcal{P}^k &:= \left\{i \colon u_i^k \ge \tau\left(\nabla_i f(u^k) + \beta\right)\right\},
\end{aligned}
$$

the Newton updates can also be written in the following form:

$$
\begin{aligned}
e_i^T d &= -u_i^k, & i \in \mathcal{A}^k \setminus \left(\mathcal{N}^k \cup \mathcal{P}^k\right)\\
\nabla^2_{i:} f(u^k) d &= -\left(\nabla_i f(u^k) + \beta\right), & i \in \mathcal{P}^k \setminus \mathcal{A}^k\\
\nabla^2_{i:} f(u^k) d &= -\left(\nabla_i f(u^k) - \beta\right), & i \in \mathcal{N}^k \setminus \mathcal{A}^k\\
\left(\delta_i \nabla^2_{i:} f(u^k) + (1-\delta_i) e_i^T\right) d &= -\tau\left(\nabla_i f(u^k) - \beta\right), & i \in \mathcal{N}^k \cap \mathcal{A}^k\\
\left(\delta_i \nabla^2_{i:} f(u^k) + (1-\delta_i) e_i^T\right) d &= -\tau\left(\nabla_i f(u^k) + \beta\right), & i \in \mathcal{P}^k \cap \mathcal{A}^k
\end{aligned}
$$

and set $u^{k+1} = u^k + d$, where $\nabla^2_{i:} f(x)$ stands for the i–th row of the Hessian and e_i is the canonical vector of $\mathbb{R}^m$

Properties

For different choices of τ and δ known efficient methods are obtained:

- For $\delta_i = 0$ and $\tau = \alpha^k$ (the steplength), a second order version of the ISTA algorithm is obtained.

Properties

For different choices of τ and δ known efficient methods are obtained:

- For $\delta_i = 0$ and $\tau = \alpha^k$ (the steplength), a second order version of the ISTA algorithm is obtained.
- For τ sufficiently small such that

$$\text{sign}\left(u_i^k - \tau\left(\nabla_i f(u^k) + \text{sign}(u_i^k)\beta\right)\right) = \text{sign}(u_i^k),\ \forall i : u_i^k \neq 0.$$

and

$$\delta_i = 0, \qquad \text{for all } i \in \left(\mathcal{N}^k \cap \mathcal{A}^k\right) \cup \left(\mathcal{P}^k \cap \mathcal{A}^k\right)$$

the orthantwise NW-CG method is obtained.

Properties

For different choices of τ and δ known efficient methods are obtained:

- For $\delta_i = 0$ and $\tau = \alpha^k$ (the steplength), a second order version of the ISTA algorithm is obtained.
- For τ sufficiently small such that

$$\text{sign}\left(u_i^k - \tau\left(\nabla_i f(u^k) + \text{sign}(u_i^k)\beta\right)\right) = \text{sign}(u_i^k), \ \forall i : u_i^k \neq 0.$$

  and

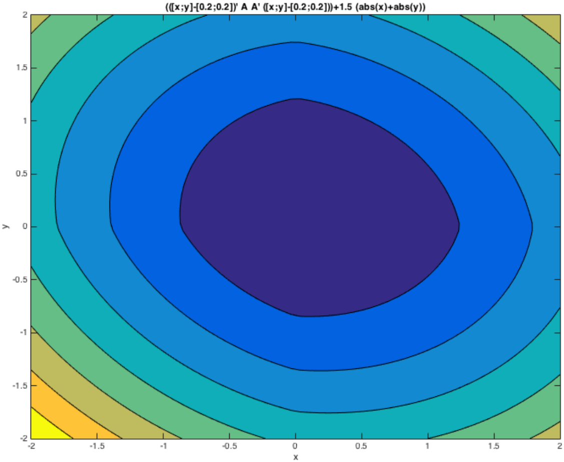
$$\delta_i = 0, \qquad \text{for all } i \in \left(\mathcal{N}^k \cap \mathcal{A}^k\right) \cup \left(\mathcal{P}^k \cap \mathcal{A}^k\right)$$

  the orthantwise NW-CG method is obtained.
- For the choice $\tau_i = \delta_i = \frac{1}{\gamma+1}$, the enriched second order method is obtained.

Outline

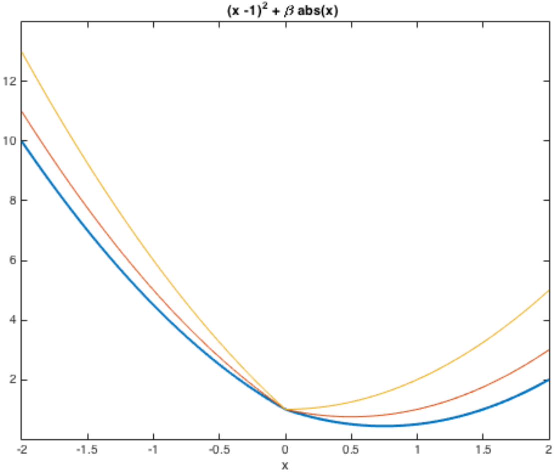
The effect of ℓ_1–norm penalization

$$\min_u \frac{1}{2} \|Au - y\|_2^2 + \beta \|u\|_1$$



The effect of ℓ_1–norm penalization

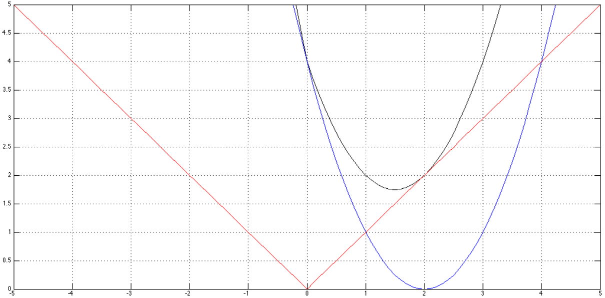
$$\min_u \frac{1}{2} \|Au - y\|_2^2 + \beta \|u\|_1$$



Orthantwise directions

The revival of subgradients

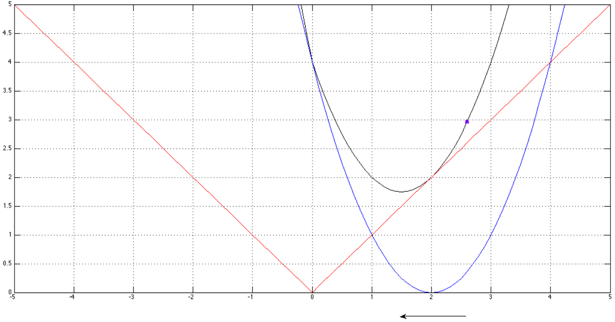
The choice of the minimum norm subgradient element gives rise to the so-called *orthantwise directions.*



$j(u)$, $f(u)$ (regular part), ℓ^1-norm

Orthantwise directions

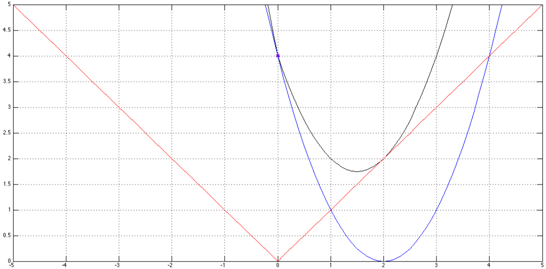
Example



If $f'(u) > 0$ and $sign(u) = 1$ then move along the negative direction.

Orthantwise directions

Example



If $u = 0$ and $f'(u) < 0$, then

- if $f'(u) + 1 < 0$, move along the positive direction,
- if $f'(u) + 1 \geq 0$, stay at 0.

Orthantwise directions

Definition

$$z_i = \begin{cases} 1 & \text{si } u_i > 0 \\ -1 & \text{if } u_i < 0 \\ 1 & \text{if } u_i = 0 \text{ y } \nabla_i f(u) < -\beta \\ -1 & \text{if } u_i = 0 \text{ y } \nabla_i f(u) > \beta \\ 0 & \text{otherwise} \end{cases}$$

Defined orthant

$$\Omega_k := \{d \colon \text{sign}(d_i) = \text{sign}(z_i)\}$$

Orthant-wise methods

Phases

- Identification of the orthant where the optimization step takes place.

Orthant-wise methods

Phases

- Identification of the orthant where the optimization step takes place.
- Computation of a descent direction in the identified orthant using second order information.

Orthant-wise methods

Phases

- Identification of the orthant where the optimization step takes place.
- Computation of a descent direction in the identified orthant using second order information.
- Projected line–search to guarantee that the iteration stays in the same orthant.

Orthant-wise methods

Phases

- Identification of the orthant where the optimization step takes place.
- Computation of a descent direction in the identified orthant using second order information.
- Projected line–search to guarantee that the iteration stays in the same orthant.
- Orthantwise directions correspond to minimum norm subgradient elements.

Orthant-wise methods

Phases

- Identification of the orthant where the optimization step takes place.
- Computation of a descent direction in the identified orthant using second order information.
- Projected line–search to guarantee that the iteration stays in the same orthant.
- Orthantwise directions correspond to minimum norm subgradient elements.

Is this fast?

OWL-QN (Andrew-Gao (2007))

Orthantwise limited memory quasi-Newton method

- Directions

$$v_k = \widetilde{\nabla}_i J(u^k) = \begin{cases} \nabla_i f(u^k) + \beta \text{sign}(u_i^k) & \text{if } u_i^k \neq 0 \\ \nabla_i f(u^k) + \beta & \text{if } u_i^k = 0 \text{ and } \nabla_i f(u^k) < -\beta \\ \nabla_i f(u^k) - \beta & \text{if } u_i^k = 0 \text{ and } \nabla_i f(u^k) > \beta \\ 0 & \text{otherwise} \end{cases}$$

OWL-QN (Andrew-Gao (2007))

Orthantwise limited memory quasi-Newton method

- Directions

$$v_k = \widetilde{\nabla}_i J(u^k) = \begin{cases} \nabla_i f(u^k) + \beta \text{sign}(u_i^k) & \text{if } u_i^k \neq 0 \\ \nabla_i f(u^k) + \beta & \text{if } u_i^k = 0 \text{ and } \nabla_i f(u^k) < -\beta \\ \nabla_i f(u^k) - \beta & \text{if } u_i^k = 0 \text{ and } \nabla_i f(u^k) > \beta \\ 0 & \text{otherwise} \end{cases}$$

- Multiplying by limited memory inverse Hessian (or solving the BFGS full system) approximation of the regular part

$$d^k = B_k^{-1} v^k$$

OWL-QN (Andrew-Gao (2007))

Orthantwise limited memory quasi-Newton method

ModeMat

- Directions

$$v_k = \widetilde{\nabla}_i J(u^k) = \begin{cases} \nabla_i f(u^k) + \beta \text{sign}(u_i^k) & \text{if } u_i^k \neq 0 \\ \nabla_i f(u^k) + \beta & \text{if } u_i^k = 0 \text{ and } \nabla_i f(u^k) < -\beta \\ \nabla_i f(u^k) - \beta & \text{if } u_i^k = 0 \text{ and } \nabla_i f(u^k) > \beta \\ 0 & \text{otherwise} \end{cases}$$

- Multiplying by limited memory inverse Hessian (or solving the BFGS full system) approximation of the regular part

$$d^k = B_k^{-1} v^k$$

- Projection: preserve components if signs coincide; otherwise set to 0.

$$p^k = \mathcal{P}(d^k, v^k),$$

where $\mathcal{P}_i(x,y) = \begin{cases} x_i & \text{if } sign(x_i) = sign(y_i) \\ 0 & \text{otherwise.} \end{cases}$

Iteration

Resulting iteration

$$u^{k+1} \leftarrow \mathcal{P}_{\mathcal{O}}(u^k + \alpha_k p^k)$$

where:

$$\mathcal{P}_{\mathcal{O}}(u_i) = \begin{cases} u_i & \text{if sign}(u_i) = \text{sign}(z_i) \\ 0 & \text{otherwise.} \end{cases}$$

and α_k is chosen according to the line search rule:

$$J(\mathcal{P}_{\mathcal{O}}(u^k + \alpha p^k)) \leq J(u^k) - \sigma (v^k)^T [\mathcal{P}_{\mathcal{O}}(u^k + \alpha p^k) - u^k]$$

NW–CG (Byrd et al. (2012))

Orthantwise Newton-CG algorithm

Steepest descent type direction:

$$\widetilde{\nabla}_i J(u^k) = \begin{cases} \nabla_i f(u^k) + \beta \text{sign}(u_i^k) & \text{if } u_i^k \neq 0 \\ \nabla_i f(u^k) + \beta & \text{if } u_i^k = 0 \text{ and } \nabla_i f(u^k) < -\beta \\ \nabla_i f(u^k) - \beta & \text{if } u_i^k = 0 \text{ and } \nabla_i f(u^k) > \beta \\ 0 & \text{otherwise} \end{cases}$$

NW–CG (Byrd et al. (2012))

Orthantwise Newton-CG algorithm

ModeMat

Steepest descent type direction:

$$\widetilde{\nabla}_i J(u^k) = \begin{cases} \nabla_i f(u^k) + \beta \text{sign}(u_i^k) & \text{if } u_i^k \neq 0 \\ \nabla_i f(u^k) + \beta & \text{if } u_i^k = 0 \text{ and } \nabla_i f(u^k) < -\beta \\ \nabla_i f(u^k) - \beta & \text{if } u_i^k = 0 \text{ and } \nabla_i f(u^k) > \beta \\ 0 & \text{otherwise} \end{cases}$$

or, equivalently, $\widetilde{\nabla}_i J(u) = \nabla_i f(u) + \beta z_i$, for all *meaningful* components with

$$z_i = \begin{cases} 1 & \text{si } u_i > 0 \\ -1 & \text{if } u_i < 0 \\ 1 & \text{if } u_i = 0 \text{ y } \nabla_i f(u) < -\beta \\ -1 & \text{if } u_i = 0 \text{ y } \nabla_i f(u) > \beta \\ 0 & \text{otherwise} \end{cases}$$

NW–CG (Byrd et al. (2012))

Orthantwise Newton-CG algorithm

Steepest descent type direction:

$$\widetilde{\nabla}_i J(u^k) = \begin{cases} \nabla_i f(u^k) + \beta \text{sign}(u_i^k) & \text{if } u_i^k \neq 0 \\ \nabla_i f(u^k) + \beta & \text{if } u_i^k = 0 \text{ and } \nabla_i f(u^k) < -\beta \\ \nabla_i f(u^k) - \beta & \text{if } u_i^k = 0 \text{ and } \nabla_i f(u^k) > \beta \\ 0 & \text{otherwise} \end{cases}$$

or, equivalently, $\widetilde{\nabla}_i J(u) = \nabla_i f(u) + \beta z_i$, for all *meaningful* components with

$$z_i = \begin{cases} 1 & \text{si } u_i > 0 \\ -1 & \text{if } u_i < 0 \\ 1 & \text{if } u_i = 0 \text{ y } \nabla_i f(u) < -\beta \\ -1 & \text{if } u_i = 0 \text{ y } \nabla_i f(u) > \beta \\ 0 & \text{otherwise} \end{cases}$$

Defined orthant: $\Omega_k := \{d \colon \text{sign}(d_i) = \text{sign}(z_i)\}$

Subspace minimization

Define the strong active set as $A_k := \{i : z_i^k = 0\}$

Subspace minimization

Define the strong active set as $A_k := \{i : z_i^k = 0\}$

Subspace minimization

$$\min_{d \in \mathbb{R}^n} J(u_k) + \widetilde{\nabla} J(u^k)^T d + \frac{1}{2} d^T B_k d$$

sujeto a: $d_i = 0, \;$ for $i \in A_k$.

Subspace minimization

Define the strong active set as $A_k := \{i : z_i^k = 0\}$

Subspace minimization

$$\min_{d \in \mathbb{R}^n} J(u_k) + \widetilde{\nabla} J(u^k)^T d + \frac{1}{2} d^T B_k d$$

sujeto a: $d_i = 0,$ for $i \in A_k$.

CG solution of the linear system

$$\left[Y_k^T B_k Y_k\right] d^Y = -Y_k^T \widetilde{\nabla} J(u^k),$$

where Y_k is a basis spanning the set of free variables. The increment is given by $d_k = Y_k d^Y$.

Subspace minimization

Define the strong active set as $A_k := \{i : z_i^k = 0\}$

Subspace minimization

$$\min_{d \in \mathbb{R}^n} \; J(u_k) + \widetilde{\nabla} J(u^k)^T d + \frac{1}{2} d^T B_k d$$

sujeto a: $d_i = 0$, for $i \in A_k$.

CG solution of the linear system

$$\left[Y_k^T B_k Y_k\right] d^Y = -Y_k^T \widetilde{\nabla} J(u^k),$$

where Y_k is a basis spanning the set of free variables. The increment is given by $d_k = Y_k d^Y$.

Set $u_{k+1} = u_k + \alpha_k d^k$, where α_k is chosen according to

$$J(\mathcal{P}_{\mathcal{O}}(u^k + \alpha d^k)) \leq J(u^k) - \sigma \widetilde{\nabla} J(u^k)^T [\mathcal{P}_{\mathcal{O}}(u^k + \alpha d^k) - u^k]$$

Enriched Hessian information

joint work: J.C. De los Reyes, E. Loayza and P. Merino

Idea: Incorporate more information on the second order matrix.

$$u^{k+1} \to \mathcal{P}_{\mathcal{O}} \left[u^k - \alpha_k \left(B_k + \; ? \; \right)^{-1} \nabla \widetilde{J}(u^k) \right]$$

Enriched Hessian information

joint work: J.C. De los Reyes, E. Loayza and P. Merino

Idea: Incorporate more information on the second order matrix.

$$u^{k+1} \to \mathcal{P}_{\mathcal{O}}\left[u^k - \alpha_k\,(B_k + \,?\,)^{-1}\,\nabla\widetilde{J}(u^k)\right]$$

How to do that?

Enriched Hessian information

joint work: J.C. De los Reyes, E. Loayza and P. Merino

Idea: Incorporate more information on the second order matrix.

$$u^{k+1} \to \mathcal{P}_{\mathcal{O}} \left[u^k - \alpha_k \left(B_k + ? \right)^{-1} \nabla \widetilde{J}(u^k) \right]$$

How to do that?

In a distributional sense the second derivative of the ℓ^1-term is given by Dirac's delta function:

$$\delta(u) = \begin{cases} +\infty & \text{if } u = 0 \\ 0 & \text{otherwise.} \end{cases}$$

Enriched Hessian information

joint work: J.C. De los Reyes, E. Loayza and P. Merino

Mode Mat

Idea: Incorporate more information on the second order matrix.

$$u^{k+1} \to \mathcal{P}_{\mathcal{O}} \left[u^k - \alpha_k \left(B_k + ?\ \right)^{-1} \nabla \widetilde{J}(u^k) \right]$$

How to do that?

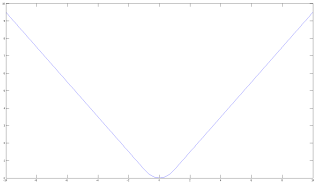
In a distributional sense the second derivative of the ℓ^1-term is given by Dirac's delta function:

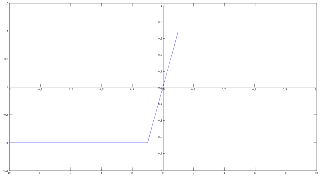
$$\delta(u) = \begin{cases} +\infty & \text{if } u = 0 \\ 0 & \text{otherwise.} \end{cases}$$

Can we use this?

Huber regularization

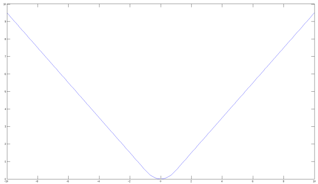
$$h_\gamma(u_i) = \begin{cases} \gamma \frac{u_i^2}{2} & \text{if } |u_i| \leq \frac{1}{\gamma}, \\ |u_i| - \frac{1}{2\gamma} & \text{if } |u_i| > \frac{1}{\gamma}. \end{cases} \qquad \nabla h_\gamma(u_i) = \frac{\gamma u_i}{\max\{1, \gamma |u_i|\}}$$

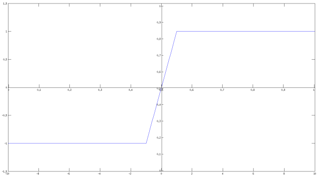




Huber regularization

$$h_\gamma(u_i) = \begin{cases} \gamma \frac{u_i^2}{2} & \text{if } |u_i| \leq \frac{1}{\gamma}, \\ |u_i| - \frac{1}{2\gamma} & \text{if } |u_i| > \frac{1}{\gamma}. \end{cases} \qquad \nabla h_\gamma(u_i) = \frac{\gamma u_i}{\max\{1, \gamma |u_i|\}}$$





Properties

The Huber function is continuously differentiable and has a second generalized derivative.

Weak second order information

$$\left[\nabla^2 h_\gamma(u)\right]_{ii} = \quad \frac{\gamma}{\max\{1, \gamma|u_i|\}} - \gamma^2 \frac{\chi\, u_i \,\mathsf{sign}(u_i)}{\max\{1, \gamma|u_i|\}^2},$$

where χ is the indicator function of the set $\{i : |u_i| > 1/\gamma\}$.

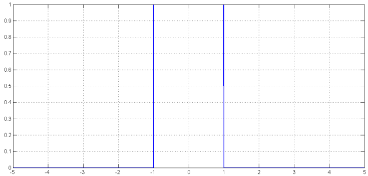
Weak second order information

$$\left[\nabla^2 h_\gamma(u)\right]_{ii} = \frac{\gamma}{\max\{1,\gamma|u_i|\}} - \gamma^2 \frac{\chi\, u_i\, \text{sign}(u_i)}{\max\{1,\gamma|u_i|\}^2},$$

where χ is the indicator function of the set $\{i : |u_i| > 1/\gamma\}$.

From this we have

$$\left(\nabla^2 h_\gamma(u)\right)_{ii} = \begin{cases} \gamma & \text{si } \gamma|u_i| \le 1 \\ 0 & \text{otherwise} \end{cases}$$



Proposed algorithm

Mode Mat

Enriched orthant-wise method

$$u^{k+1} \to \mathcal{P}_{\mathcal{O}}\left[u^k - \alpha_k\left[\left(B_k + \nabla^2 h_\gamma(u^k)\right)^{-1}\nabla\tilde{J}(u^k)\right]\right]$$

$$\nabla_i\widetilde{J}(u) = \begin{cases} \nabla_i f(u) + \beta\,\text{sign}(u_i) & \text{if } u_i \neq 0 \\ \nabla_i f(u) + \beta & \text{if } u_i = 0 \text{ and } \nabla_i f(u_i) < -\beta \\ \nabla_i f(u) - \beta & \text{if } u_i = 0 \text{ and } \nabla_i f(u_i) > \beta \\ 0 & \text{otherwise} \end{cases}$$

Line–search step: find the largest $\alpha_k \in [0,1]$ such that

$$J\left(\mathcal{P}_{\mathcal{O}}[u^k + \alpha_k d^k]\right) \leq J(u^k) + \sigma\nabla\tilde{J}(u^k)^T\left(\mathcal{P}_{\mathcal{O}}[u^k + \alpha_k d^k] - u^k\right)$$

Properties of the enriched algorithm

- Orthantwise directions (with projection) are indeed descent directions.

Properties of the enriched algorithm

- Orthantwise directions (with projection) are indeed descent directions.
- Defining the active set by

$$\mathcal{S}^k = \{i : z_i^k = 0\},$$

if u^k is close to $\bar{u}$ and strict complementarity holds, then

$$\mathcal{S}^k \subset \mathcal{S}^{k+1} \subset \mathcal{A}(\bar{u})$$

Properties of the enriched algorithm

- Orthantwise directions (with projection) are indeed descent directions.
- Defining the active set by

$$\mathcal{S}^k = \{i : z_i^k = 0\},$$

if u^k is close to $\bar{u}$ and strict complementarity holds, then

$$\mathcal{S}^k \subset \mathcal{S}^{k+1} \subset \mathcal{A}(\bar{u})$$

Neighborhood is larger in our algorithm.

Properties of the enriched algorithm

- Orthantwise directions (with projection) are indeed descent directions.
- Defining the active set by

$$\mathcal{S}^k = \{i : z_i^k = 0\},$$

if u^k is close to $\bar{u}$ and strict complementarity holds, then

$$\mathcal{S}^k \subset \mathcal{S}^{k+1} \subset \mathcal{A}(\bar{u})$$

Neighborhood is larger in our algorithm.

Practical consequence: Faster identification of active set.

Properties of the enriched algorithm

- Orthantwise directions (with projection) are indeed descent directions.
- Defining the active set by

$$\mathcal{S}^k = \{i : z_i^k = 0\},$$

if u^k is close to $\bar{u}$ and strict complementarity holds, then

$$\mathcal{S}^k \subset \mathcal{S}^{k+1} \subset \mathcal{A}(\bar{u})$$

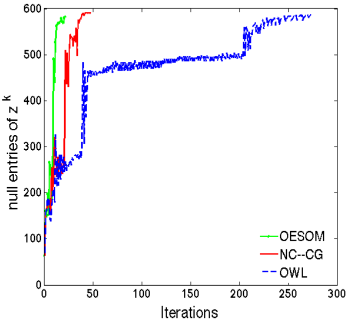
Neighborhood is larger in our algorithm.

Practical consequence: Faster identification of active set.

- Once you get close to zero, you may want to stay there.

Behaviour for PDE-constrained optimization

Comparison of methods



Random quadratic problems

$$\min_{u \in \mathbb{R}^n} \quad \frac{1}{2} u^T Q u + \beta \|u\|_{\ell^1}$$

- Q is generated by the MATLAB function *sprandsym*, ensuring the positive definiteness
- Matrices with 25% of zero entries
- β was generated randomly in the interval $[2.5; n/3]$
- Fail criteria: if convergence is not reached within first 5000 iterations
- We solve 1000 experiments

Performance

Condition number of Q	Algorithms		
	Enriched	NW-CG	OWL
	Number of failures		
Moderate	0	0	0
High	0	260	2
Total	0	260	2

Table: Failures out of a set of 1000 random generated problems.

Performance

Condition number of Q	Algorithms		
	Enriched	NW-CG	OWL
	Number of failures		
Moderate	0	0	0
High	0	260	2
Total	0	260	2

Table: Failures out of a set of 1000 random generated problems.

Algorithm	Mean	Variance
Enriched	4.2970	0.4252
NW–CG	69.3149	9.6458e+04
OWL	3.7154	0.7394

Table: Global performance of the algorithms

Are there drawbacks?

Are there drawbacks?

Main issue: Needs to solve the linear system

$$\left(B_k + \nabla^2 h_\gamma(u^k)\right) d^k = -\widetilde{\nabla} J(u^k)$$

which can be prohibitive for large–scale optimization problems:

- computational power: solve a linear system every step is expensive
- storage: System matrix may need tons of RAM, possibly can not be stored at all

Reduced Oesom

Alternative: incorporate the projection in the building of the second order matrix.
Reorder the iterates

$$d^k = (d^k_{\mathcal{S}^k}, d^k_{I\setminus\mathcal{S}^k})^T$$

Assemble the reduced second order matrix

$$(B^k_R)_{ij} = (B^k)_{ij} + (\nabla^2 h_\gamma(u^k))_{ij}, \quad i \in \mathcal{S}^k, \forall j$$

the following system may be solved:

$$\begin{pmatrix} I \quad 0 \\ \hline B^k_R \end{pmatrix} \begin{pmatrix} d^k_{\mathcal{S}^k} \\ d^k_{I\setminus\mathcal{S}^k} \end{pmatrix} = \begin{pmatrix} -x^k_{\mathcal{S}^k} \\ -\tilde{\nabla}\varphi(x^k)_{I\setminus\mathcal{S}^k} \end{pmatrix}.$$

Reduced Oesom

Alternative: incorporate the projection in the building of the second order matrix.
Reorder the iterates

$$d^k = (d^k_{\mathcal{S}^k}, d^k_{I\setminus\mathcal{S}^k})^T$$

Assemble the reduced second order matrix

$$(B^k_R)_{ij} = (B^k)_{ij} + (\nabla^2 h_\gamma(u^k))_{ij}, \quad i \in \mathcal{S}^k, \forall j$$

the following system may be solved:

$$\begin{pmatrix} I \quad 0 \\ \hline B^k_R \end{pmatrix} \begin{pmatrix} d^k_{\mathcal{S}^k} \\ d^k_{I\setminus\mathcal{S}^k} \end{pmatrix} = \begin{pmatrix} -x^k_{\mathcal{S}^k} \\ -\tilde{\nabla}\varphi(x^k)_{I\setminus\mathcal{S}^k} \end{pmatrix}.$$

Now, second order information is only used for the update of x^k_i, $i \in I\setminus\mathcal{S}^k$

Reduced Oesom

ModeMat

- $\mathcal{S}^k$ tends to be large (sparse solution), therefore the former system can be solved by decoupling.
- B_R^k my be a dense matrix
- Reduced Oesome algorithm can be casted as a Semi-smooth Newton Method by setting $\tau = {}^1/_{(\gamma+1)}$ and γ large, such that

$$\text{sign}\left(x_i^k - \tau\left(\nabla_i f(x^k) + \text{sign}(x_i^k)\beta\right)\right) = \text{sign}(x_i^k) \quad \text{for all } i : x_i^k \neq 0.$$

Outline

Conclusions and perspectives

- Sparse optimization problems are present in a wide variety of application areas, from machine learning to image processing.
- The optimal solutions may be characterized by optimality conditions involving primal and dual variables.
- There is a large class of first order methods that efficiently computes each iteration, although many iterations are needed.
- The inclusion of second-order information (strong and "weak") improves the algorithms performance.
- Semismooth Newton methods provide an alternative for the numerical solution of the optimality condition.

Perspectives

- Alternative line-search rules
- Adaptive choice of different parameters
- Relation to semismooth Newton methods-investigation of further SSN based algorithms
- Development of algorithms for problems involving the l_p-norm, with $1 < p < 2$.
- Development of efficient methods for sparse optimal control problems.
- Several application examples

Bibliography

G. Andrew and J. Gao.
Scalable training of ℓ_1—regularized log-linear models.
In Proceedings of the Twenty Fourth Conference on Machine Learning (ICML), 2007.

A. Beck and M. Teboulle.
A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems.
SIAM Journal on Imaging Sciences, 2(1):183–202, March 2009.

Dimitri P Bertsekas and Sanjoy K Mitter.
Steepest descent for optimization problems with nondifferentiable cost functionals.
Technical report, MIT, Dept. of Electrical Engineering, 1971.

R. Byrd, G. Chin, J. Nocedal, and Y. Wu.
Sample size selection in optimization methods for machine learning.
Mathematical Programming, 134(1), 2011.

R. Byrd, G.M. Chin, J. Nocedal, and F. Oztoprak.
A family of second-order methods for convex ℓ_1—regularized optimization.
Mathematical Programming, pages 1–33, 2012.

E. Chouzenoux, J.C. Pesquet, and A. Repetti.
Variable metric forward–backward algorithm for minimizing the sum of a differentiable function and a convex function.
Journal of Optimization Theory and Applications, 162(1):107–132, 2014.

Juan Carlos De Los Reyes, Estefanıa Loayza, and P Merino.
Second-order orthant-based methods with enriched hessian information for sparse ℓ_1-optimization.
Computational Optimization and Applications, 67(2):225–258, 2017.

K. Fountoulakis and J. Gondzio.
A second-order method for strongly convex ℓ_1-regularization problems.
Mathematical Programming, 156(1):189–219, 2016.

Roland Herzog, Georg Stadler, and Gerd Wachsmuth.
Directional sparsity in optimal control of partial differential equations.
SIAM Journal on Control and Optimization, 50(2):943–963, 2012.

Y. Nesterov.
Gradient methods for minimizing composite functions.
Mathematical Programming, 140(1):125–161, 2013.

Stefan Solntsev, Jorge Nocedal, and Richard H Byrd.
An algorithm for quadratic l1-regularized optimization with a flexible active-set strategy.
Optimization Methods and Software, 30(6):1213–1237, 2015.

S. Sra, S. Nowozin, and S.J. Wright.
Optimization for machine learning.
MIT Press, 2012.

G. Stadler.
Elliptic optimal control problems with L^1-control cost and applications for the placement of control devices.
Comput. Optim. Appl., 44(2):159–181, 2009.

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

Stephen J Wright.
Accelerated block-coordinate relaxation for regularized optimization.
SIAM Journal on Optimization, 22(1):159–186, 2012.

Yangjing Zhang, Ning Zhang, Defeng Sun, and Kim-Chuan Toh.
An efficient hessian based algorithm for solving large-scale sparse group lasso problems.
arXiv preprint arXiv:1712.05910, 2017.