SPARSE SIGNAL ESTIMATION WITH NONLINEAR CONJUGATE GRADIENTS

Goran Marjanovic and Victor Solo

School of Electrical Engineering and Telecommunications
The University of New South Wales Sydney, Australia

ABSTRACT

Many problems in signal processing involve finding sparse solutions to linear systems of equations. The usual way of achieving this involves minimizing a mixed penalty function composed of a quadratic $l_2$ term and a sparse inducing $l_1$ term. Some existing algorithms for minimization include cyclic descent, gradient projection and iterative fixed point methods. Conjugate gradient is well known as a fast algorithm for linear quadratic problems. Here we develop a nonlinear conjugate gradient algorithm for the $l_1$ penalized least squares problem. This new method uses no line search and is found to be very stable. Description of its performance is provided as well as simulations to demonstrate convergence and comparison to another algorithm.

Index Terms— Sparse, conjugate gradient, $l_1$.

1. INTRODUCTION

The need for sparse signal estimation is evident in problems related to signal processing such as compressed sensing, [1], problems in bioinformatics related to gene selection, [2], and in other problems involving statistical inference, [3]. As a result, there has been substantial interest in solving the $l_1$ penalized least squares problem

$$\min_{\beta} \frac{1}{2} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1$$

Typically one has a noisy observation $y_{n \times 1}$ of a signal $X\beta$ where $X_{n \times p}$ is a matrix of regressors and $\beta_{p \times 1}$ a vector of unknown parameters. The scalar $\lambda$ is a non negative penalty parameter, $\|s\|_2$ denotes the Euclidean $l_2$ norm of $s$, and $\|s\|_1 = \sum_i |s_i|$ is the $l_1$ norm of $s$. The presence of the $l_1$ term encourages small components of $\beta$ to become exactly zero, thus promoting sparse solutions, [3]. Several optimization algorithms have been proposed to solve (1). The iterative algorithm in [4] uses a pseudo gradient to update a single component of $\beta$ per iteration. For all non zero components of $\beta$, the corresponding components of the pseudo gradient are the actual gradient components of the penalty in (1). The cyclic descent algorithm in [5] and [6] updates all components of $\beta$ individually per iteration. An updated component of $\beta$ is given by a simple expression derived by keeping all other components constant. The Least Angle Regression, (LARS), algorithm in [7] assumes that the response $y$ is determined by a linear combination of a subset of potential covariates, i.e. columns of $X$ scaled by non zero $\beta$ components with added error. Consequently, the LARS algorithm produces an estimate of which $\beta$ component to include and which to set to zero by increasing $\beta$ values in a direction equiangular to each one’s correlation with the residual. The gradient projection (GP) algorithms in [3] referred to as GPSR (gradient projection for sparse reconstruction) solve (1) by reformulating it as a bound constrained quadratic program, (BCQP), which has defined gradient for every point and its dimension is twice that of the original problem (1). At each iteration, a search is made in the direction of the negative gradient, projecting onto the region imposed by the bounds of the BCQP, and executing a backtracking line search to ensure a sufficient descent is attained in the quadratic penalty of the BCQP. Finally, the algorithm in [8] exploits the existence of a subgradient of the penalty in (1) at $\beta$ that is equal to zero in order to obtain an equation suitable for fixed point iteration. Successive substitutions of $\beta$ in this equation allow a build up of points that converge towards the solution of (1).

The Conjugate gradient, (CG) algorithm is well known in signal processing, [9], and often provides very fast convergence on large problems. There are many extensions of the CG to the nonlinear realm, [10]. Here we developed a method that does not require the use of line search techniques which could be computationally demanding. For nonlinear CG to work however we need differentiability so we approximate $\|\beta\|_1$ in a smooth way e.g. [11], [12], [13]. We call the resultant algorithm $l_1$-CG.

The remainder of the paper is organized as follows. Section 2 describes the application of $l_1$-CG to the smooth approximation of (1). Section 3 contains simulations showing the convergence of $l_1$-CG as well as a comparison to the cyclic descent algorithm from [5]. Finally, concluding remarks are provided in Section 4.
2. NONLINEAR CONJUGATE GRADIENTS

The nonlinear CG method is used to find the local minimum of a nonlinear function using its gradient. The $l_1$-CG is a very simple version of this method as it uses only formulas for the step length rather than more complicated line search techniques. Consequently, it is very easy to implement.

2.1. Smooth approximation of $l_1$

The penalty in (1) consists of a smooth $l_2$ norm and a non smooth $l_1$ norm. As a result, it does not have a defined gradient and Hessian at all points $\beta$ that have at least one zero component. Reformulation of problem (1) is thus necessary and involves replacing the $l_1$ norm by its smooth approximation. There are numerous smooth approximations of $l_1$ norm and a non linear term denoted by the square diagonal matrix built upon the components of $x$. For $\beta \neq 0$, we have that $\tanh(\beta_i) \approx \text{sign}(\beta_i)$ where $\text{sign}(\beta_i) = \partial |\beta_i| / \partial \beta_i$. As a result, for $\beta_i \neq 0$ the $i$th component of $\nabla J(\beta)$ approximates the $i$th component of the existing gradient of the penalty in (1). The Hessian in (5) consists of a constant term and a non linear term dependent on $\beta$ and $T$. As $T >> 1$, its diagonal elements are approximately given by the diagonal elements of $X^T X$ for all $\beta_i \neq 0$. However, for all $\beta_i = 0$ they become $X^T X$ with an offset of $XT$.

Now, define $\lambda_{\max} = \|XTy\|_\infty$, where $\|u\|_\infty$ denotes the $l_\infty$ norm of vector $u$. For all $\lambda \geq \lambda_{\max}$ the optimal solution of (1) and approximately of (3) is 0, [3]. As a result, in this paper we will impose the restriction $\lambda \leq \lambda_{\max}$.

The general nonlinear CG and thus the $l_1$-CG algorithm consists of an update

$$\beta_{(k+1)} = \beta_{(k)} + \alpha_k d_{(k)}$$

where the search direction is given by

$$d_{(k)} = \begin{cases} g_{(k)} & \text{for } k = 0, \\ g_{(k)} + \gamma_k d_{(k-1)} & \text{for } k \geq 1, \end{cases}$$

where $g_{(k)}$ denotes the gradient $\nabla J(\beta_{(k)})$, $\gamma_k$ is a scalar and $\alpha_k$ is a step length. The scalar $\gamma_k$ is given by $\gamma_k = \max(0, \gamma_{PR})$, where $\gamma_{PR}$ is the well known Polak-Ribiére formula, [14], given by

$$\gamma_{PR} = \frac{g_{(k)}^T (g_{(k)} - g_{(k-1)})}{\|g_{(k-1)}\|^2}$$

We now discuss two approaches for the choice of step length.

(a) Newton-Raphson: a pre-determined very simple formula

$$\alpha_k = - \frac{d_{(k)}^T \nabla J(\beta_{(k)})}{d_{(k)}^T H(\beta_{(k)})) d_{(k)}}$$

where $H(\beta_{(k)})$ is the Hessian of $J(\beta)$ in (5).

(b) Sigmoid-Invariant: also a pre-determined formula.

2.2. Line search and orthogonality

Before describing the Sigmoid-Invariant step length we motivate it by discussing exact line search. Subsequently, we will present our Sigmoid-Invariant step length procedure in subsection 2.3.

One expected feature of CG methods is to have the gradient at a particular iteration orthogonal to the previous search direction, i.e. having the orthogonality condition $g_{(k+1)}^T d_{(k)} = 0$ satisfied for all iterations $k$. This is referred to as having exact line search, [15]. By replacing $g_{(k+1)}$ by $\nabla J(\beta_{(k)} + \alpha_k d_{(k)})$ from (4) and simplifying, the orthogonality condition at the $k$th iteration becomes

$$\lambda(\alpha_k) = d_{(k)}^T X^T (X \beta_{(k)} - y)$$

$$+ \alpha_k d_{(k)}^T X^T X d_{(k)}$$

$$+ \lambda d_{(k)}^T \tanh(T \beta_{(k)} + \alpha_k T d_{(k)}) = 0$$

where $\tanh(x_{p \times 1}) = (\tanh(x_1), \tanh(x_2), \ldots, \tanh(x_p))^T$. Solving equation (10) with standard methods such as Newton’s method is not appropriate due to the highly sensitive nature of $\Lambda$ with respect to $\alpha_k$. Let us denote the step length that satisfies (10) by $\alpha^*_{k}$ and the gradient of $\Lambda$ by $\Lambda'$. For some plausible values of parameters $T$, $\lambda$, $y$, $X$ and $d_{(k)}$, the value of $\Lambda'$ in a large neighbourhood surrounding $\alpha^*_{k}$ can become extremely small. As a result, one cannot expect good convergence when an initial guess is not properly chosen, [16]. There exist other plausible values of the same parameters for which $\Lambda'$ at $\alpha^*_{k}$ is almost infinite but becomes extremely small.
in a very small neighbourhood of $\alpha_k^+$. In this case, $\Lambda$ mimics the behaviour of a $\text{sign}$ function in [16]. Newton’s method in this scenario fails to converge and so is not applicable. By applying any other method one needs to take into account that the error between the estimated step length and $\alpha_k^+$ needs to be exceptionally small as any small deviance could result in possibly large deviance of $\Lambda(\alpha_k^+)$ from zero.

Let us denote the total number of iterations made when implementing $l_1$-CG with Newton-Raphson step length by $N_{NR}$. In many implementations this step length did not satisfy equation (10) for a large fraction of $N_{NR}$ as $\lambda \rightarrow \lambda_{\text{max}}$, eventuating in slower convergence than for smaller $\lambda$ values. It was also noticed that $N_{NR}$ began to randomly fluctuate between implementations.

### 2.3. Sigmoid-invariant step length

Introducing the Sigmoid-Invariant in place of the Newton-Raphson step length allows a much slower rise in the total number of iterations made with suppressed fluctuations for large $\lambda \leq \lambda_{\text{max}}$. This is because the new step length actually makes use of equation (10) unlike the previous one. It is given by the following simple pre-determined formula

\[
\alpha_k = \begin{cases} 
\phi_1 & \text{if } |\Lambda(\phi_1)| \leq |\Lambda(\phi_2)| \text{ and } \phi_1 > 0, \\
\phi_2 & \text{if } |\Lambda(\phi_1)| \geq |\Lambda(\phi_2)| \text{ and } \phi_2 > 0,
\end{cases} \quad (11)
\]

where $\Lambda(\phi)$ is given by (10) and

\[
\begin{align*}
\phi_1 &= -\frac{d_i^{\alpha_k} \nabla f(\beta(x_k))}{d_i^{\alpha_k} H(\beta(x_k)) d_i^{\alpha_k}}, \\
\phi_2 &= \frac{1}{2} [a(k) + b(k) + \text{sign}(\Lambda(a(k))) I(\alpha_k)], \\
a(k) &= -\frac{d_i^{\alpha_k} X^T (X \beta(k) y + \lambda |d_i|) d_i^{\alpha_k} \| d_i \|_1 + \delta}{d_i^{\alpha_k} X^T X d_i^{\alpha_k}}, \\
b(k) &= -\frac{d_i^{\alpha_k} X^T (X \beta(k) y + \lambda |d_i|) d_i^{\alpha_k} \| d_i \|_1 + \delta}{d_i^{\alpha_k} X^T X d_i^{\alpha_k}}, \\
\text{sign}(x) &= \begin{cases} 
1 & \text{if } x > 0, \\
0 & \text{if } x = 0, \\
-1 & \text{if } x < 0, 
\end{cases}
\end{align*}
\]

and

\[
I(\alpha_k) = \int_{a(k)}^{b(k)} \tanh(\eta T \Lambda(\alpha_k)) d\alpha_k, \quad \text{where } \eta T \gg 1, \text{ and } 0 < \delta \leq 1 \text{ are tuning parameters.}
\]

The values $a(k)$ and $b(k)$ are chosen such that $a(k) < \alpha_k^+ < b(k)$ and $\Lambda(a(k)) \Lambda(b(k)) < 0$. The expression for $\phi_2$ given in [16], is pre-determined and does not follow any line search rules. It is a non iterative way to approximate the root, $\alpha_k^+$, of equation (10) using a transform based on the hyperbolic tangent function. Ultimately, finding this root is reduced to evaluating the integral $I(\alpha_k)$ of the transformed function. The integral implicitly contains information for the location of $\alpha_k^+$ which is invariant under the transform. If the value of it is accurately computed then a major benefit of this method is that accurate approximations of $\alpha_k^+$ can be achieved regardless of the behaviour of $\Lambda’$ at $\alpha_k^+$ and its neighbourhood. Implementations of $l_1$-CG using the Sigmoid-Invariant step length can be carried out with Matlab function quad to calculate the integral $I(\alpha_k)$.

Let us denote the total number of iterations made using Sigmoid-Invariant step length by $N_{SI}$. The step length in general achieved exact line search for almost all $N_{SI}$ for various initial conditions. As $\lambda \rightarrow \lambda_{\text{max}}$, $N_{SI}$ did increase but was smaller and much more consistent compared to $N_{NR}$ for the same initial condition.

As a result of these considerations we conclude that the Newton-Raphson step length though simple is unreliable whereas the Sigmoid-Invariant step length is reliable.

### 3. SIMULATIONS

Here we compare $l_1$-CG using Sigmoid-Invariant step length with the cyclic descent algorithm which is known to be fairly rapid [6].

A compressed sensing scenario is considered with the intention of reconstructing a length $p$ sparse signal $\beta$ from $n$ noisy observations $y_i$, $n < p$. We modify the setup in [3]. The sparse signal $\beta$ is filled with $m$ randomly placed $\pm 1$ spikes while the rest of its $p - m$ components are set to zero. The sparsity is thus given by $\mu = m/p$ and the signal to noise ratio by $\text{SNR} \approx (\mu/\sigma^2)(p/n)$, where $\sigma^2$ is the variance of white Gaussian noise in the observations. The matrix $X_{n \times p}$ is obtained by first filling it with independent samples of standard Gaussian distribution and then mean correcting and orthonormalizing the columns. Several trials were conducted for fixed $n$, $p$, $m$, $\sigma^2$ and the tuning parameters to obtain a more general result. Fig.1 is obtained with $\lambda = 0.1\lambda_{\text{max}}$, and the initial condition $\beta(0) = \delta_{p \times 1}$, a $p \times 1$ vector of fives.

![Fig. 1. $l_1$-CG vs. cyclic descent: $n = 50$, $p = 100$, $\lambda = 0.0925$, $T = \eta T = 200$, $\delta = 0.1$, $m = 4$, $\mu = 0.04$, $\text{SNR} = 5$, 500 with $\sigma^2 = 16 \times 10^{-3}$, $0.16 \times 10^{-3}$ respectively.](image-url)

From Fig.1 we see that the $l_1$-CG converges much faster than
cyclic descent. This is retained in a general compressed sensing scenario as $|\beta_{[0]}|_{i}$ diverges from zero for most $i \in [1, p]$. Of course, for higher SNR (as used in [3]) both algorithms converge more rapidly but the relative superiority of $l_1$-CG is kept. This superiority is also maintained for larger values of $n$ and $p$ shown in Fig.2 below.

![Fig. 2. $l_1$-CG vs. cyclic descent: $n = 500$, $p = 1000$, $\lambda = 0.0925$, $T = \eta I = 200$, $\delta = 0.1$, $m = 40$, $\mu = 0.04$, SNR = 5 with $\sigma^2 = 16 \times 10^{-3}$.](image)

4. CONCLUSION

In this paper a nonlinear CG method without line search ($l_1$-CG) was proposed for recovering sparse signals by minimization of the $l_1$ penalized function. The algorithm’s relative performance was described for the Newton-Raphson and Sigmoid-Invariant step lengths. It was noticed that in general for large $\lambda \leq \lambda_{max}$, the use of Sigmoid-Invariant step length proved superior in terms of total number of iterations made to reach a solution. Simulations were also provided illustrating that the performance of $l_1$-CG with Sigmoid-Invariant step length is better than the performance of the cyclic descent again in terms of total number of iterations made.

Overall, a big benefit of $l_1$-CG is that it is a very simple and stable method using only formulas for the step length rather than more complicated line search techniques. As a result, it was found to be very easy to implement.

5. REFERENCES


