Fast Optimization Methods for L1 Regularization: A Comparative Study and Two New Approaches

Mark Schmidt[1], Glenn Fung[2], Rómer Rosales[2]

[1] Department of Computer Science University of British Columbia,
[2] IKM CKS, Siemens Medical Solutions, USA

Abstract. L1 regularization is effective for feature selection, but the resulting optimization is challenging due to the non-differentiability of the 1-norm. In this paper we compare state-of-the-art optimization techniques to solve this problem across several loss functions. Furthermore, we propose two new techniques. The first is based on a smooth (differentiable) convex approximation for the L1 regularizer that does not depend on any assumptions about the loss function used. The other technique is a new strategy that addresses the non-differentiability of the L1-regularizer by casting the problem as a constrained optimization problem that is then solved using a specialized gradient projection method. Extensive comparisons show that our newly proposed approaches consistently rank among the best in terms of convergence speed and efficiency by measuring the number of function evaluations required.

1 Introduction

Parsimonious models are normally preferred over more complex ones. Sparsity, a concept commonly employed to describe model complexity, can be defined in terms of the training examples that are used to define the model (as in Support Vector Machines), or in terms of the covariates (feature selection). In addition to parsimony, feature selection can help prevent overfitting in problems with many input features relative to the amount/variability of the data; see [9, 18] for an overview.

The problem of obtaining an optimal subset of features for a linear classifier is known to be NP-hard [18], and is computationally unsolvable in most large applications. A popular strategy is to use a continuous, convex relaxation of the non-convex feature selection problem, through the use of a prior or a regularizer that encourages sparsity in the model[1].

In recent years, there has been an increasing interest in the L1 regularizer, since it has the beneficial effects of regularizing model coefficients (as in L2 regularization), but yields sparse models that are more easily interpreted

[1] *e.g.,* the number of nonzero components of the normal to a hyperplane classifier is equivalent to the number of features it needs to employ.

[17]. Logarithmic sample complexity bounds[2] allow L1-regularized models to learn effectively even under an exponential number of irrelevant features (relative to training samples) [13], giving better performance than ridge (L2) penalties in these scenarios. Furthermore, L1-regularization has appealing asymptotic sample-consistency in terms of variable selection [19].

For this paper, we will consider problems with the general form:

$$\min_x f(x) \equiv L(x) + \lambda ||x||_1. \tag{1}$$

Here, $L(x)$ is a loss function, and the goal is to minimize this loss function with the L1-penalty, yielding a regularized sparse solution. Efficient algorithms have been proposed for the special cases where $L(x)$ has a specific functional form, such as a Gaussian [3] or Logistic [11] negative log-likelihood. In this paper, we focus on the more general case where $L(x)$ is simply a twice-differentiable continuous function, no specific form is assumed.

Since the objective function is non-differentiable when x contains values of 0, this precludes the use of standard unconstrained methods. This has lead to a wide variety of approaches proposed in the literature to solve problems of this form. In this paper we evaluate twelve classical and state-of-the-art L1 regularization methods over several loss functions in this general scenario (in most cases these are generalized versions of algorithms for specific loss functions proposed in the literature). In addition, we propose two new methods:

(i) The first proposed method, *SmoothL1*, uses a smooth approximation to the L1-regularizer that is continuous and differentiable, allowing us to formulate Newton (or Quasi-Newton) methods to solve the resulting optimization problems independently of the loss function used.
(ii) The second method proposed, *ProjectionL1*, addresses the differentiability by reformulating the problem as a non-negatively constrained optimization problem. We further describe the use of the *Two-Metric Projection* method put forward by [7] to solve the resulting optimization problem efficiently.

Our numerical results indicate that some strategies for addressing the non-differentiable loss are much more efficient than others, while our two simple proposed strategies are competitive with the best strategies (including more complex algorithms). We end with a discussion of the choice of algorithms in different scenarios.

2 Fast optimization methods for L1 regularization

In this section, we review various previously proposed approaches and propose two new optimization techniques that can be used for L1-regularized optimization (Table 1 at the end gives a high level overview of these approaches). Although most methods proposed in the literature have been for individual loss

[2] Number of training examples required to learn a function

functions, we focus on methods that can be extended to handle a general unconstrained differentiable loss. We concentrate on a single scalar λ value, although all techniques below are easily generalized to include a λ for each element (that may be equal to zero to avoid penalizing some elements). Except where otherwise noted, the algorithms are stabilized (to ensure global convergence) by using a back-tracking line search that finds a step length t satisfying the Armijo condition (we generate trial points using cubic interpolation of function and directional derivative values, and use a sufficient decrease parameter of 0.0001). In this section we will assume analytic second derivatives, and defer discussion of methods that avoid explicit Hessian calculation until the end.

2.1 SubGradient Strategies

We first examine optimization strategies that use sub-gradients to extricate the task of dealing with the non-differentiable gradient. At a local minimizer $\bar{x}$ of $f(x)$, we observe the following first-order optimality conditions:

$$\begin{cases} \nabla_i L(\bar{x}) + \lambda sign(\bar{x_i}) = 0, & |\bar{x_i}| > 0 \\ |\nabla_i L(\bar{x})| \leq \lambda, & \bar{x_i} = 0 \end{cases}$$

These conditions can be used to define a sub-gradient for each x_i whose negation represents the coordinate-wise direction of maximum descent:

$$\nabla_i f(x) = \begin{cases} \nabla_i L(x) + \lambda sign(x_i), & |x_i| > 0 \\ \nabla_i L(x) + \lambda, & x_i = 0, \nabla_i L(x) < -\lambda \\ \nabla_i L(x) - \lambda, & x_i = 0, \nabla_i L(x) > \lambda \\ 0, & x_i = 0, -\lambda \leq \nabla_i L(x) \leq \lambda \end{cases}$$

Using this sub-gradient, the *Gauss-Seidel* algorithm of [16] uses a working set of variables, and does an exact line search to optimize the working set variable whose sub-gradient is largest. Variables begin at $x_i = 0$ with an empty working set, and the variable with the largest sub-gradient magnitude is introduced whenever the working set satisfies the optimality conditions. This continues until no variable can be introduced. The *Grafting* procedure uses a variation that jointly solves the working set variable optimization with standard unconstrained techniques [15]. In contrast, rather than separating variables into active and working sets, and introducing the working set variable with the largest sub-gradient magnitude, the *Shooting* algorithm simply cycles through all variables, optimizing each in turn [6]. Analogously, we can also define a *Sub-Gradient Descent* strategy that attempts to minimize $f(x)$ in terms of x jointly using the above sub-gradient, and defines the working set at each iteration to be those variables not satisfying the optimality conditions[3].

[3] In our experiments, we used the line search of [16] for both the Gauss-Seidel and Shooting algorithm. For Grafting and Sub-Gradient Descent, we use Newton steps of the form $x := x - t\nabla^2 f(x)^{-1}\nabla f(x)$ for a step length t to optimize the working set.

2.2 Unconstrained Approximations

An alternative to working directly with $f(x)$ and using sub-gradients to address non-differentiability, is to replace $f(x)$ with an (often continuous and twice-differentiable) approximation $g(x)$. The problem of minimizing $g(x)$ can then be solved with unconstrained optimization techniques, such as performing Newton iterations of the form $x := x - t\nabla^2 g(x)^{-1}\nabla g(x)$ for a suitable step length t. A simple example is the *epsL1* approximation [11]:

$$g(x) = L(x) + \lambda \sum_i \sqrt{x_i^2 + \epsilon}$$

This function is differentiable and approximates $f(x)$ for small ϵ. An alternative approximation is the class of log-barrier functions:

$$g(x) = L(x) + \lambda ||x||_1 - \mu \sum_j \log c_j(x)$$

In the *Log-Barrier* method, the constraint functions $c_j(x)$ force feasibility of iterates in a constrained formulation (see Section 2.3). In the *Log(norm(x))* method, each $c_i(x)$ is set to x_i^2, preventing any variable from becoming exactly 0. For these methods, the unconstrained optimizer must implement truncation of the step lengths in order to maintain positivity of all $c_j(x)$. Although the optimization can be performed for a fixed small value of μ, in the Log-Barrier method it is standard to solve (or approximately solve) the unconstrained problem with a decreasing sequence of μ values, which avoids ill-conditioning of the Hessian preventing convergence (see [14] for additional details).

SmoothL1 Approximation Method We propose another type of smooth approximation, that takes advantage of the non-negative projection operator $(x)_+ = \max(x, 0)$. This projection function can be smoothly approximated, by the integral of a sigmoid function:[1]:

$$(x)_+ \approx p(x, \alpha) = x + \frac{1}{\alpha}\log(1 + \exp(-\alpha x)) \qquad (2)$$

$p(x, \alpha)$ is a member of the class of smoothing functions presented in [1] proposed to solve complementarity problems. This smooth approximation of the projection has been used to transform the standard L2-penalized SVM formulation into an efficiently-solved unconstrained problem [12]. We also make use of the nice properties of $p(x, \alpha)$, but aiming for the different goal of achieving sparsity in the covariates.

By combining $p(x, \alpha)$ with the identity $|x| = (x)_+ + (-x)_+$ we arrive at the following smooth approximation for the absolute value function that consists of the sum of the integral of two sigmoid functions:

$$\begin{aligned} |x| &= (x)_+ + (-x)_+ \approx p(x, \alpha) + p(-x, \alpha) \\ &= \tfrac{1}{\alpha}\left[\log(1 + \exp(-\alpha x)) + \log(1 + \exp(\alpha x))\right] \\ &\overset{\text{def}}{=} |x|_\alpha \end{aligned} \qquad (3)$$

The corresponding loss function is: $g(x) = L(x) + \lambda \sum_i |x_i|_\alpha$; we refer to it as the *SmoothL1* approximation. It can be shown that $|x|_\alpha$ converges to $|x|$ as α approaches ∞ (the proof is similar to [12]), while $|x|_\alpha$ is twice differentiable:

$$\nabla(|x|_\alpha) = (1 + \exp(-\alpha x))^{-1} - (1 + \exp(\alpha x))^{-1} \tag{4}$$

$$\nabla^2(|x|_\alpha) = 2\alpha \exp(\alpha x)/(1 + \exp(\alpha x))^2 \tag{5}$$

With a smooth approximation, an unconstrained optimization method can be applied to $g(x)$ for a large value of α as a proxy for minimizing $f(x)$. However, for large α the SmoothL1 approximation is not appropriately modeled by a quadratic for variables near 0. To account for this, we use a continuation strategy analogous to Log-Barrier methods, where we take Newton steps between increasing the parameter α (beginning from a small α where the quadratic approximation is appropriate, and terminating at a sufficiently large value of α). The advantage of this new approach over Log-Barrier approximations is that a specialized line search that truncates the step to maintain constraint feasibility is not required (allowing the potential use of more sophisticated line search criteria), and that it does not involve solving a problem with double the number of variables (associated with using a constrained formulation).

An approach related to unconstrained approximations are Expectation Maximization (EM) approaches (see [4]). These approaches use a scale mixture of normals prior on the variables ($x_i|\tau_i \sim N(0, \tau_i)$), where the variances of the individual Gaussians have an exponential prior: $p(\tau_i|\sqrt{\lambda}) = \frac{\sqrt{\lambda}}{2} \exp(\frac{-\tau_i\sqrt{\lambda}}{2})$. Under this representation, integrating over τ_i yields a Laplacian density (and thus an L1-regularizer after taking logarithms) for $p(x_i|\lambda)$. In the *EM* approach the individual τ_i are treated as missing variables, and the 'E-step' computes the expectation of τ_i. Subsequently, the 'M-Step' uses this expectation to (exactly or approximately) compute the MAP parameters with the scale mixture (L2) prior. Algorithmically, this is equivalent to using the following approximation (where x^{old} is the value from the previous iteration)[4]:

$$g(x) = L(x) + \lambda \sum_i \frac{1}{2}||x_i^{old}||_2^2 + \frac{1}{2}\frac{||x_i||_2^2}{|x_i^{old}|_1}$$

Although this approach has previously been presented as a fixed-point Iteratively Reweighted Least Squares (IRLS) update, it is straightforward to modify it in order to compute the Newton descent direction under this approximation (allowing the method to be applied to loss functions that do not yield an IRLS approximation).

2.3 Constrained Formulations

A third general approach to address the non-differentiability of the L1-regularizer is to cast the problem as a constrained optimization problem. One approach to

[4] Numerical instability of this approach arises as x_i approaches 0. Strategies to avoid this include using a pseudo-inverse [17], reformulation [4], or by defining the working set as those variables whose magnitude is above a threshold.

do this is to replace λ with a variable $t \propto 1/\lambda$ and solve the constrained problem:

$$\min_x L(x) \quad s.t. ||x||_1 \leq t \tag{6}$$

Recently, [11] presented an algorithm for L1-regularized Logistic Regression, where the Logistic Regression IRLS update is computed subject to the constraint $||x||_1 \leq t$. The solution to the constrained Weighted Least Squares problem can be efficiently calculated using the LARS algorithm [3]. This 'IRLS-LARS' algorithm (with an Armijo backtracking linesearch) proved more efficient than other approaches examined in [11] for L1-regularized Logistic Regression. This specific strategy can clearly be more generally applied to any loss function that yields an IRLS update, but it is not a general strategy since many loss functions do not yield an IRLS approximation[5].

We can extend the IRLS-LARS algorithm to a general algorithm by observing that the algorithm is an IRLS reformulation of a Sequential Quadratic Programming (SQP) update (where a unit step length is assumed). That is, IRLS-LARS minimizes a Quadratic approximation to the function, subject to a linearization of the constraints (the linearization is redundant in this case). To handle the L1 constraint in a more general setting, we split x into non-negative variables representing positive and negative components, by defining new variables $x^+ = max(0, x)$ and $x^- = -min(0, w)$ (thus, $x = x^+ - x^-$). This gives the following constrained problem (a general form of a formulation used in [17]):

$$\min_{x^+, x^-} L(x^+ - x^-) \quad s.t. \sum_i [x_i^+ + x_i^-] \leq t, \forall_i x_i^+ \geq 0, x_i^- \geq 0 \tag{7}$$

From a probabilistic perspective, a difficulty with this formulation is that the constraints become degenerate as t approaches the L1-norm of the Maximum Likelihood Estimate (an analogous problem is present for non-probabilistic losses), a value that may not be known or desirable to compute. We use the following alternative formulation that avoids this problem, makes clear the strength of the Laplacian regularizer, and yields simple bound constraints on the variables:

$$\min_{x^+, x^-} L(x^+ - x^-) + \lambda \sum_i [x_i^+ + x_i^-] \quad s.t. \forall_i x_i^+ \geq 0, x_i^- \geq 0 \tag{8}$$

A general *SQP* algorithm takes descent steps of the form $x := x - td$ for a step length t, where the descent direction d is calculated by solving the following Quadratic Program [8]:

$$\min_d (\nabla L(x^+ - x^-) + \lambda 1)^T d + \frac{1}{2} d^T \nabla^2 L(x^+ - x^-) d \tag{9}$$

$$s.t. \forall_i x_i^+ + d_i^+ \geq 0, x_i^- + d_i^- \geq 0 \tag{10}$$

[5] IRLS updates are typically applicable in cases where the loss is an affine function of the covariates.

The linear constraints allow the objective function $f(x)$ to be used directly as a measure of progress (assuming the variables are initially non-negative and the step length is never greater than one), avoiding the need to use special techniques to avoid a scenario known as the Maratos effect [14]. For strictly convex problems, the SQP iterates converge super-linearly to the optimal solution [8], explaining the low number of iterates reported by [11] for IRLS-LARS. This general SQP algorithm can be considerably less efficient than the IRLS-LARS algorithm, since a general Quadratic Program must be solved at each iteration (although warm-starting is possible), and the algorithm does not take advantage of the form of the constraints.

ProjectionL1 Method We propose to take advantage of the non-negative bound constraints by using a *Two-Metric Projection* method [7], which we now outline. Using the notation $x^* = [x^+ \; x^-]^T$, the active set of constraints for non-negative bounds $x^*_i \geq 0$ at an iterate x_* is defined as $\{i | x^*_i = 0, \nabla L(x^+ - x^-) + \lambda > 0\}$. To avoid very small steps, we replace the test $x^*_i = 0$ with $0 \leq x^*_i \leq \epsilon$, for a small ϵ. At each iteration, we optimize the variables whose bound constraint is not active (the working set) using a projected-gradient strategy. A standard projected-gradient algorithm would take descent steps on the working set of the form: $x^* := [x^* - t\nabla f(x^+ - x^-)]^+$, where t represents the step length[6] and the element-wise 'plus' function projects onto the non-negative orthant.

The gradient projection strategy is appealing since it allows rapid changes in the active set, and is especially suited to handle this type of problem due to the simplicity of the constraint projection operator. However, its convergence may be slow due to the use of the steepest decent direction. Hence, the 'Two-Metric Projection' strategy scales the descent direction by the inverse of the working set's Hessian matrix, yielding the following simple update: $x^* := [x^* - t\nabla^2 f(x^*)^{-1}\nabla f(x^*)]^+$. As in SQP, this algorithm achieves a superlinear rate of convergence [7]. However, it has a substantially reduced iteration cost compared to SQP (or IRLS-LARS).

To complete our discussion of the L1-regularized optimization methods proposed in the literature, we note that in the Basis Pursuit Denoising literature, Interior Point (primal-dual log-barrier) methods have been used (for example, [2]). These methods, closely related to Log-Barrier methods, simultaneously optimize both the primal variables x^* and a set of dual variables ν corresponding to the Lagrange multipliers. It is straightforward to adapt these methods to the general case using the bound-constrained formulation above. Assuming x^* is feasible and ν is non-negative, for a barrier parameter μ the remaining (modified) first-order optimality (KKT) conditions for the bound-constrained problem can be written as follows (where $\circ$ denotes the element-wise Hadamard product):

$$0 = r(x^*, \nu) \equiv \begin{bmatrix} \nabla L(x^+ - x^-) + \lambda 1 - \nu \\ -\nu \circ x^* - \mu 1 \end{bmatrix}.$$

[6] The line search along the projection arc requires a modified Armijo condition [7]

This equation corresponds to the gradient of the Lagrangian, and the (modified) complementary condition. We seek to solve the equation $r(x^*, \nu) = 0$ by taking Newton-Raphson steps of the form $[x^* \ \nu]^T := [x^* \ \nu]^T - t\nabla r(x^*, \nu)^{-1} r(x^*, \nu)$, where the step length t is truncated to ensure that x^* and ν are non-negative (computing $\nabla r(x^*, \nu)^{-1} r(x^*, \nu)$ requires some algebraic manipulation). Between iterates, the barrier parameter μ is updated based on an update rate (we use 10), the number of constraints m, and the duality gap $\nu^T x^*$:$\mu = 10m(\nu^T x^*)^{-1}$ (see [5] for a review of Interior Point methods).

3 Experiments

We have applied the above strategies to a variety of loss functions and data sets. Specifically, we looked at a generalized version of the **Gauss-Seidel**, **Shooting**, **Grafting**, **Sub-Gradient**, **epsL1**, **Log-Barrier**, **Log(norm(x))**, **SmoothL1**, **EM**, **SQP**, **ProjectionL1**, and **Interior Point** methods. Although the general-L1 framework make no assumptions about convexity, we have restricted our experiments to convex functions.

All methods were run until the same convergence criteria was met (*i.e.,* where appropriate, that the step length between iterates, change in function value between iterates, negative directional derivative, or optimality condition was less than 10^{-6}). We assessed the ability of the methods to optimize a loss function known only through a 'black box' function that returns the objective value and derivatives for a given parameter setting. Convergence was measured based on function evaluations; this is, the number of times the algorithm invoked the 'black box' (to make the comparisons fair, all of the implementations were designed and tuned with this in mind). The iterates were truncated to 250 such evaluations, and methods whose final loss was greater than 10^{-3} times the minimum found across the methods were assigned the maximum value of 250 evaluations to punish for low accuracy. This was only needed in a small minority of cases, since all methods typically either found a high accuracy solution, or reached the maximum number of iterations. We used a second-order (Hessian-based Newton) strategy across all methods examined.

3.1 Binary Classification

Our first experiment focused on the problem of optimizing the negative log-likelihood associated with binary Probit Regression (using y as class labels, z as the features, ϕ as the error function, and x as the paramters): $L(x) = \log(\phi(\frac{y_i x^T z_i}{\sqrt{(2)}}))$. We applied all 12 optimization strategies to 11 publicly available data sets[7] from the UCI[8] amd Statlog[9] repositories. All methods were

[7] 1: Wisconsin Breast Cancer, 2: Australian Heart, 3: Pima Diabetes, 4: Australian Credit, 5: Sonar, 6: Ionosphere, 7: German, 8: Bright, 9: Dim, 10: Adult, 11: Census

[8] http://www.ics.uci.edu/~mlearn/MLRepository.html

[9] http://www.liacc.up.pt/ML/old/statlog/

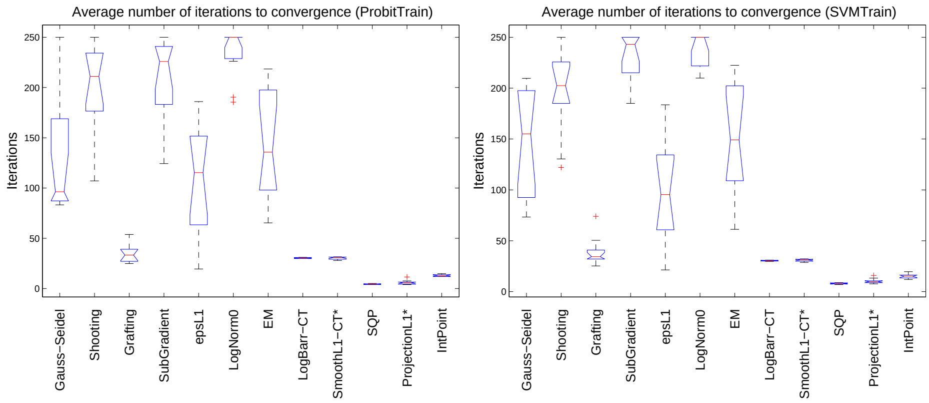

Fig. 1. Distribution of function evaluations (averaged over λ) across 11 data sets to train: (left) a Probit Regression classifier with L1-regularization and (right) a Smooth Support Vector Machine classifier with L1-regularization (*=new method)

initialized with $x = 0$ (those that do not allow this used $x = 0.01$). Using $\lambda_{max} \overset{\text{def}}{=} \max_i |\nabla L(0)|$ as the maximum value of λ for each data set[10], we evaluated each data set at λ_{max} multiplied by each of $[.1, .2, .3, .4, .5, .6, .7, .8, .9]$.

Since the methods discussed in this report apply to general differentiable loss functions, we can easily replace the binary Probit Regression loss function with other loss functions. We tested the optimizers using a differentiable loss function closely related to the hinge loss used in Support Vector Machines: $l(x) = (1 - y_i x^T z_i)^+$. In 'Smooth' Support Vector Machines, the projection ('plus') function in the hinge loss is replaced with the smooth approximation in Section 2.2, yielding a differentiable objective [12]. We repeated the Probit Regression experiment with the Smooth Support Vector Machine loss function (we set the parameter α controlling the accuracy of the loss approximation to 5). Fig. 1 plots the distribution of the mean number of iterations to convergence across the data sets for both binary classification loss functions. We also examined the binary Logistic Regression loss (not shown due to space limit), finding results similar to the Probit Regression experiment (these results are consistent with the findings reported in [11]).

3.2 Multinomial and Structured Classification

We examined optimizing two more complicated objectives than those described above: Multinomial Logistic Regression (using the Softmax function) and (2-dimensional) Conditional Random Fields (CRFs). These represent more challenging scenarios since the Hessians of these models are often indefinite. We

[10] This and higher values produce $x = 0$ as their solution, following from the first-order optimality conditions of the unconstrained problem.

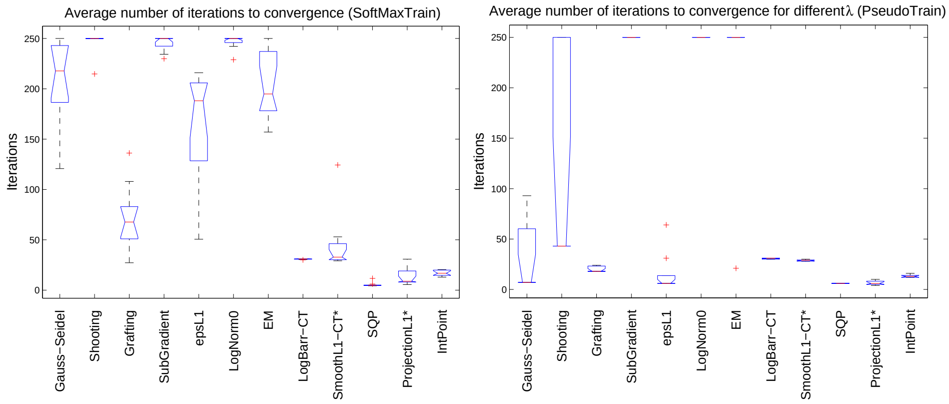

Fig. 2. Left: Distribution of function evaluations (averaged over λ) across 11 data sets to train a Multinomial Logistic Regression classifier with L1-regularization. Right: evaluations on the image patch classification data set to train an L1-regularized 2D Conditional Random Field evaluated for various λ values (*=new method).

trained Multinomial Logistic Regression classifiers on 11 data sets[11] from the UCI repository and the Statlog project. We trained the 2D Conditional Random Field on an image patch classification task [10], using the following Pseudo-likelihood (v represents edge weights that are also penalized).

$$l(x,v) = \log(1 + \exp(y_i x^T z_i + \sum_{j \in nei(i)} y_i y_j v^T z_{ij})) \tag{11}$$

A clear advantage of our approach of treating the loss as a generic function, is that it is trivial to apply the methods to these more complicated objectives (once the loss function and its partial derivatives are defined). We used the 'CRF2D' software to provide the CRF loss that we augmented with L1-regularization[12]. The summarized results of these experiments are shown in Fig. 2.

4 Discussion

Table 1 summarizes the different methods we have examined, including an aggregate convergence ranking across the experiments (ie. number of times the loss is evaluated over all training examples), and a ranking of the iteration speed of the different methods (which typically only depends on the number of variables). If we were to completely ignore iteration cost, SQP would be the fastest method. However, in terms of runtime, ProjectionL1 was typically the fastest

[11] 1:Iris, 2:Glass, 3:Wine, 4:Vowel, 5:Vehicle, 6:LED, 7:Satellite, 8:Waveform21, 9:DNA, 10:Waveform40, 11:Shuttle

[12] http://www.cs.ubc.ca/∼murphyk/Software/CRF/crf.html

Optimization Method	Approx Objective	Sub-Gradient	Explicit Constraints	Convergence Ranking	Iteration Speed Ranking
Gauss-Seidel [16]	N	Y	N	6	1
Shooting [15]	N	Y	N	8	1
Grafting [6]	N	Y	N	4	2
Sub-Gradient	N	Y	N	9	2
epsL1 [11]	Y	N	N	5	2
Log(norm(x))	Y	N	N	10	2
EM [4]	Y^*	Y^{***}	N	7	2
Log-Barrier [14]	Y^*	N	Y	3	3
SmoothL1 [ThisPaper]	Y^*	N	N	3	2
SQP [11]	N	N	Y	1	4
ProjectionL1 [ThisPaper]	N	Y^{***}	Y	1	3
Interior Point [5]	Y^{**}	N	Y	2	3

Table 1. Convergence ranking is determined by the average number of iterations to convergence across the 405 experiments (methods whose average values are within 5 are grouped, the methods that required the fewest iterations are ranked 1, the method requiring the most iterations is ranked 10). Iteration speed is based on the cost of computing the descent direction if all variables are non-zero (the fastest methods are ranked 1, the slowest ranked 4). *: method improves the approximation between iterations. **: method uses a constrained objective that is improved between iterations. ***: methods use the correct gradient, but only for the working set (other sub-gradient methods also restrict to the working set).

method across the experiments, since its iteration cost (solving a symmetric linear system) is substantially smaller than the cost of solving a quadratic program (even if LARS is used). Although the approaches that explicitly enforced constraints were generally superior to the unconstrained approaches, the SmoothL1 approach was the most effective approach that did not use a constrained approach (the constrained approaches have a higher iteration cost since they solve a linear system with double the number of variables). The 250 iteration limit may have favorably skewed the convergence of methods that reached this limit (making it difficult to draw definite conclusions on these). However, overall our experiments indicated that the proposed ProjectionL1 strategy was the most efficient in terms of runtime on the test problems, although the proposed SmoothL1 algorithm may be efficient on problems with many variables, while SQP may be more efficient on problems with very expensive function evaluations.

In some scenarios, it might not be practical to compute (or store) analytic Hessians. If we replace the analytic Hessian with a suitable (limited-memory) Hessian approximation (ie. L-BFGS), all of the above methods can be applied without modification (with the exception of the InteriorPoint method). This substantially reduces the iteration cost and memory requirements (for all but the coordinate descent strategies), at the cost of an increase in function evaluations.

In this work, we have reviewed 12 methods for solving general L1-regularized optimization problems, and provided a numerical comparison on several standard machine learning problems. Two of these methods are novel (introduced in this paper) and prove to be among the most efficient overall. Due to space constraints, we have omitted some information that we would have liked to include. Online[13], we have made available additional details/proofs on some of the methods, code (to enable reproducible research), and additional experimental results.

References

1. C. Chen and O. L. Mangasarian. A class of smoothing functions for nonlinear and mixed complementarity problems. *Comput. Optim. Appl*, 5(2):97–138, 1996.
2. S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Sci. Comput.*, 20(1):33–61, 1999.
3. B. Efron, I. Johnstone, T. Hastie, and R. Tibshirani. Least angle regression. *Ann. Stat.*, 32(2):407–499, 2004.
4. M. Figueiredo. Adapative sparseness for supervised learning. *IEEE. Trans. Pattern. Anal. Mach. Intell.*, 25(9):1150–1159, 2003.
5. R. M. Freund and S. Mizuno. Interior point methods: Current status and future directions. *Optima*, 51:1–9, 1996.
6. W. Fu. Penalized regressions: The bridge versus the LASSO. *J. Comput. Graph. Stat.*, 7(3):397–416, 1998.
7. E. Gafni and D. Bertsekas. Two-metric projection methods for constrained optimization. *SIAM J. Contr. Optim.*, 22(6):936–964, 1984.
8. U. M. Garcia Palomares and O. L. Mangasarian. Superlinearly convergent Quasi–Newton algorithms for nonlinearly constrained optimization problems. *Math. Program.*, 11:1–13, 1976.
9. I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, 2003.
10. S. Kumar and M. Hebert. Discriminative random fields: A discriminative framework for contextual interaction in classification. In *ICCV*, 2003.
11. S.-I. Lee, H. Lee, P. Abbeel, and A.Y. Ng. Efficient L1 regularized logistic regression. In *AAAI*, 2006.
12. Y.-J. Lee and O. L. Mangasarian. SSVM: A smooth support vector machine. *Comput. Optim. Appl.*, 20:5–22, 2001.
13. A. Ng. Feature selection, L1 vs. L2 regularization, and rotational invariance. In *ICML*, pages 78–85, New York, NY, USA, 2004. ACM Press.
14. J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer-Verlag, New York, 1999.
15. S. Perkins, K. Lacker, and J. Theiler. Grafting: Fast, incremental feature selection by gradient descent in function space. *J. Mach. Learn. Res.*, 3:1333–1356, 2003.
16. S. Shevade and S. Keerthi. A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, 19(17):2246–2253, 2003.
17. R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Roy. Stat. Soc. B*, 58(1):267–288.
18. J. Weston, A. Elisseeff, B. Scholkopf, and M. Tipping. Use of the zero norm with linear models and kernel methods. *J. Mach. Learn. Res.*, 3:1439–1461, 2003.
19. P. Zhao and B. Yu. On model selection consistency of LASSO. *J. Mach. Learn. Res.*, 7:2541–2567, 2007.

[13] http://www.cs.wisc.edu/∼gfung/GeneralL1