9.520: Statistical Learning Theory and Applications March 8th, 2010

Sparsity Based Regularization

Lecturer: Lorenzo Rosasco *Scribe: Ioannis Gkioulekas*

1 Introduction

In previous lectures, we saw how regularization can be used to restore the well-posedness of the empirical risk minimization (ERM) problem. We also derived algorithms that use regularization to impose smoothness assumptions on the solution space (as in the case of Tikhonov regularization) or introduce additional structure by confining the solution space to low dimensional manifolds (manifold regularization). In this lecture, we will examine the use of regularization for the achievement of an alternative objective, namely *sparsity*.

During the last ten years, there has been an increased interest in the general field of sparsity. Such interest comes not only from the Machine Learning community, but also from other scientific areas. For example, in Signal Processing sparsity is examined mainly in the context of compressive sensing [CRT06, Dono06] and the so called basis pursuit [CDS96]. In the Statistics literature, basis pursuit is known as the lasso [Tibs96]. Strong connections also exist with sparse coding [OlFi97] and independent component analysis [HyOj00].

In these notes, we discuss sparsity from a regularization point of view, and only refer to these connections as they arise from within this framework. Initially, we motivate the use of sparsity and emphasize on the problem of variable selection. Then, we present the formulation of the sparsity based regularization problem, develop tractable approximations to it, and justify them using a geometric interpretation of sparsity. Finally, after discussion of some of the properties of these approximations, we describe an algorithm for the solution of the sparsity based regularization problem.

2 Motivation

The widespread availability of very high dimensional data sets makes their efficient manipulation from Machine Learning algorithms and applications very challenging. Challenges can be both computational (lack of storage and processing resources) and theoretical (lack of algorithms with performance that scales well to such high dimensions).

In order to be able to cope with these challenges, we seek to find a parsimonious model for the data, requiring a number of parameters much smaller than its dimension. In a sense we try to achieve a *compression* of the data, by removing redundancy between the various measurements originally available and keeping only those that are the most relevant. However, we try to do so in a data driven way, by using the data itself to determine our selection strategy, thus building a *data driven representation* of the data. We hope that we will be able to efficiently manipulate this new, compressed representation and that from the compression there will be no essential loss of information, as we assume that the variables selected carry almost all the relevant information for our task.

This process serves another, dual purpose. Specifically, we seek to develop a representation that is *interpretable*. We want the data driven selection strategy to help us understand the model underlying the data. In a sense, the high dimensionality of the original data set reflects our ignorance about the particular problem we are dealing with. Due to this ignorance, we believe that the full model is highly redundant and can be explained only by a small number of the most relevant variables, which our data driven scheme helps select. From this perspective, our problem is one of *variable selection*.

We provide here an instructive example, to further illustrate the above two points. Assume that we have n patients from 2 groups, which may be for example two different diseases. For each patient, we take p measurements, corresponding to the expression of p genes, with n typically being in the order of tens and p of thousands. Then, we want to build a data driven compressed representation of this data set, and use it to achieve two distinct goals. The first goal is to learn a classification rule that will predict the occurrence of the disease in future patients. A second goal is to study the variable selection procedure defined by the above representation in order to detect which particular genes are the more relevant for this classification, thus probably also responsible for the disease. Therefore, we see that there is a dual, prediction-selection goal. Ideally, the number of the selected relevant genes will be much smaller than the original p genes.

It is important to note that the procedure we seek is not just a dimensionality reduction or feature selection procedure. Viewing our problem as a variable selection problem helps us understand the difference. Consider, for example, the principal component analysis (PCA) algorithm, which is very popular for dimensionality reduction. PCA returns directions in the feature space which, in some sense, capture most of the information (variability) of the data set. However, these directions are defined as functions of the original p measurements. Thus, the resulting projection space does not tell us which the most relevant from the original measurements are, unless it happens that all the axes of this space are aligned with some axes of the original space, something generally unlikely. Therefore, PCA is not a variable selection algorithm and variable selection is a different procedure from feature selection.

3 Strategies for variable selection

So far, we have not defined what we mean by "relevant" variables, neither have we provided any method for their selection. In this section, we explore some possibilities. Note that in the following, for brevity we will only refer to the problem of classification. However, all results presented are directly applicable to the case of regression.

An obvious procedure would be to construct all the possible subsets of the p measurements with cardinality up to some N, and use them to train a classifier. Then, the best subset is defined by a joint consideration of size and error, and the most relevant variables are the ones included in this set. Although such an algorithm would return the "optimal" solution, it is computationally intractable. Indeed, the number of all possible subsets is equal to 2^p, exponential on the number of measurements p, and the problem itself is NP-complete. As we assume that p is very large for our data sets, this procedure is not feasible.

There are three different classes of methods that can be used to approximate the above scheme,

1. filter methods: such methods rank the quality of each measurement with regards to the classification task separately. Then, they use these rankings to select a variable subset of size N, by taking the N variables with the highest rankings. This scheme usually results in sub-optimal subsets as it assumes that all variables are independent, which is not generally true.

2. wrapper methods: methods of this class use heuristics to perform the search in the space of subsets of variables. A well-known member of this class is the is the recursive feature elimination (RFE) algorithm. Due to the use of heuristics, wrapper methods again are not guaranteed to return a good result.

3. embedded methods: these methods derive their name from the fact that the selection procedure is embedded in the training phase. This is in contrast to the previous two classes, where firstly a training phase is executed for all the candidate subsets and then a selection is made based on the results (rankings) of this phase. This is the class of methods we will examine in detail in the following, and from which the notion of sparsity will arise naturally.

We will not cover filter and wrapper methods in these notes. For more details on these two classes of methods, a comprehensive review can be found in [GuEl03].

4 Embedded methods

For the rest of these notes, we restrict our analysis to the finite dimensional case. Before we proceed, we formalize some notation we have already been using to some extent. Our data consists of the measurement matrix X and the labels Y, where

$$X = \begin{pmatrix} x_1^1 & \dots & \dots & \dots & x_1^p \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^1 & \dots & \dots & \dots & x_n^p \end{pmatrix}, Y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}. \tag{1}$$

Thus, X is a $n \times p$ matrix and Y a $n \times 1$ column vector. In the above, n is the number of examples and p is the number of measurements (or variables) available for each example, finite by assumption. Therefore, the i-th row of X corresponds to the measurements for the i-th example, and y_i is the label for that example. Depending on whether we study the classification or regression case, we have that $y_i \in \{-1, 1\}$ or $y_i \in \mathbb{R}$, respectively, for $i = 1, \dots, n$. Finally, we use X^j to denote the j-th column of X, $j = 1, \dots, p$. We assume that $p \gg n$.

4.1 Sparsity

For our analysis, we will assume for the rest that we have a linear model, where the output of our learning algorithm is a linear combination of the p variables,

$$f(x) = \sum_{i=1}^{p} \beta^i x^i = \langle \beta, x \rangle . \tag{2}$$

This framework allows us to define sparsity. Specifically, a function such as the above is called *sparse* if most of the coefficients β_i are zero. Informally, this means that we discard all variables for which $\beta_i = 0$ and therefore our variable selection is the small set of variables for which $\beta_i \neq 0$.

To find f, we need to specify β. From the above formulation, we have that

$$Y = X\beta. \tag{3}$$

This is an ill-posed problem. Specifically, by selecting using ERM a function of the above form, intuitively we expect it to overfit the data if we only discard few variables, and to oversmooth if we discard too many variables. As in previous lectures, in order to restore the well-posedness of the ERM problem and allow its solution to generalize, we can introduce regularization on the sparsity of the solution. In the next subsection, we examine how this can be done.

4.2 ℓ_0 regularization

The natural approach towards regularizing the sparsity of the solution of the ERM is to use the number of non-zero coefficients as a penalty. We define the ℓ_0-norm (using the term norm here is an abuse of terminology, as $\|\cdot\|_0$ does not satisfy all of the properties of a norm)

$$\|\beta\|_0 = \#\{i = 1, \dots, p \mid \beta^i \neq 0\}. \tag{4}$$

Thus, $\|\cdot\|_0$ can be seen as a measure of complexity of each function f. Then, we can regularize our problem by minimizing

$$\min_{\beta \in \mathbb{R}^P} \left\{ \frac{1}{n} \sum_{j=1}^{n} V(y_j, \langle \beta, x_j \rangle) + \lambda \|\beta\|_0 \right\} \tag{5}$$

instead of minimizing only the empirical risk.

Minimizing (5) is not computationally tractable. This problem is as difficult as the one we considered in section 3, where we needed to select the optimal of all possible subsets of measurements. The only change in this case is that the optimality criterion now also includes the regularization term. Furthermore, $\|\cdot\|_0$ is not a convex function. In fact, as we will see in subsection 4.4, the ℓ_0 norm is in a sense the most concave of all ℓ_p norms we can use for regularization. However, the minimizer of (5) is our optimal solution and, if available, can be used as a gold standard for the evaluation of other algorithms.

Due to the above undesired properties, we need to consider approximations to ℓ_0 regularization. There are two main approaches to approximating the loss function of (5):

1. using a convex relaxation, or
2. using a greedy scheme.

The second approach is easier computationally. However, as it is a local method, a mistake at the beginning of its running may render it unable to converge to a good solution without paying a considerable price in running time (or to converge at all). On the other hand, using a convex relaxation constitutes a global method. Thus, although the corresponding minimization problem is more difficult to solve, the method always converges to a good solution, depending of course on the quality of the approximation. For these reasons, in the following we only cover the convex relaxation approach.

4.3 Convex relaxation of ℓ_0 regularization

Assuming that $V(y, f(x))$ is convex, we can get a convex relaxation of (5) by replacing the ℓ_0 norm with some approximation that is convex. A natural such approximation is the ℓ_1 norm,

$$\|\beta\|_1 = \sum_{i=1}^{p} \left|\beta^i\right|, \tag{6}$$

which is a convex function of β. Then, the approximation to ℓ_0 regularization takes the form

$$\min_{\beta \in \mathbb{R}^P} \left\{ \frac{1}{n} \sum_{j=1}^{n} V(y_j, \langle \beta, x_j \rangle) + \lambda \|\beta\|_1 \right\}. \tag{7}$$

which constitutes the ℓ_1 regularization.

For the rest of our analysis, we restrict our attention to the case where $V(y, f(x))$ is the square loss. Then, the minimization problem of (7) is called *basis pursuit* in the Signal Processing literature and *lasso* in the Statistics literature. The minimization problem of (7) takes the form

$$\min_{\beta \in \mathbb{R}^P} \left\{ \|Y - X\beta\|_n^2 + \lambda \|\beta\|_1 \right\}, \tag{8}$$

where

$$\|Y - X\beta\|_n^2 = \frac{1}{n} \sum_{i=1}^{n} (y_i - \langle \beta, x_i \rangle)^2 \tag{9}$$

is the square loss.

4.4 Geometric interpretation of sparsity based regularization

A question that naturally arises from the above considerations is why lasso is a good approximation to ℓ_0 regularization. After all, we could have used Tikhonov regularization instead, in which case our minimization problem would become

$$\min_{\beta \in \mathbb{R}^P} \left\{ \|Y - X\beta\|_n^2 + \lambda \|\beta\|_2^2 \right\}, \tag{10}$$

where

$$\|\beta\|_2^2 = \sum_{i=1}^{p} \left|\beta^i\right|^2. \tag{11}$$

In this case, we would again obtain a solution of the form

$$f_s(x) = \sum_{i=1}^{p} \beta^i x^i = \langle \beta_s, x \rangle. \tag{12}$$

We will use a geometric argument to show that most of the coefficients of f_s (or at least no less than those of the solution we obtain from lasso) are nonzero. We consider the case when $p = 2$ for visualization purposes, although our conclusions also hold for higher dimensions. For the various p-norms, we write the corresponding minimization problem in a constrained minimization form instead of the lagrangian form we have used in (8) and (10),

$$\min_{\beta \in \mathbb{R}^P} \left\{ \|\beta\|_p^p \right\}, \tag{13}$$

$$\text{subject to } \|Y - X\beta\|_n^2 \leq R^2. \tag{14}$$

We refer to figure 1. From the expressions of the ℓ_1 and ℓ_2 norms, it is easy to see that the corresponding ℓ_1 and ℓ_2 balls have the shape drawn at the figure. Furthermore, we can see that the ℓ_0 "ball" corresponds to the segments of the β_1 and β_2 axes between 0 and the points of intersection with the ℓ_1 and ℓ_2 balls. Finally, all balls ℓ_p with $0 < p < 1$ will be curves between ℓ_1 and ℓ_0 with positive curvature. Therefore, we verify that that ℓ_1 is the closest approximation to ℓ_0 that is also convex, and that ℓ_0 is the most concave of all p-norms as stated in subsection 4.2.

Constraint (14) corresponds to a beam of lines parallel to the line defined by the solution of the linear system $Y = X\beta$ and at distance $-R \leq d \leq R$ from it, as shown in the figure (and assuming that the linear system does not have a unique solution). Then, from (13), to solve the minimization problem graphically we need to find the ℓ_p ball of minimum radius that intersects with at least one of the lines of the beam. We see immediately that, for the ℓ_1 ball, this point of intersection coincides with one of the points where the ball intersects with the β_1 or β_2 axes. At these points, we have $\beta_2 = 0$ or $\beta_1 = 0$, respectively. To the contrary, for the ℓ_2 ball, the point of intersection is not on either of the two axes, unless the solution of the linear system $Y = X\beta$ is the vertical line $\beta_1 = \textit{constant}$ or the horizontal line $\beta_2 = \textit{constant}$. Therefore the minimizer of the constrained optimization problem (13), (14) for the case $p = 1$ always has one of its two coordinates equal to 0. On the other hand, for the case $p = 2$, none of the two coordinates of the solution is zero, except for the two special cases we mentioned. We note that these two cases correspond to the PCA algorithm returning directions aligned with the axes of the original measurement space, as mentioned in section 2.

We can see that the same analysis holds for all other balls of norms ℓ_p with $p > 1$. We conclude that the convex approximation of ℓ_0 regularization using the ℓ_1 norm produces solutions β that are at least as sparse as those produced by approximations using any other norm ℓ_p with $p > 1$, and at most times much sparser.

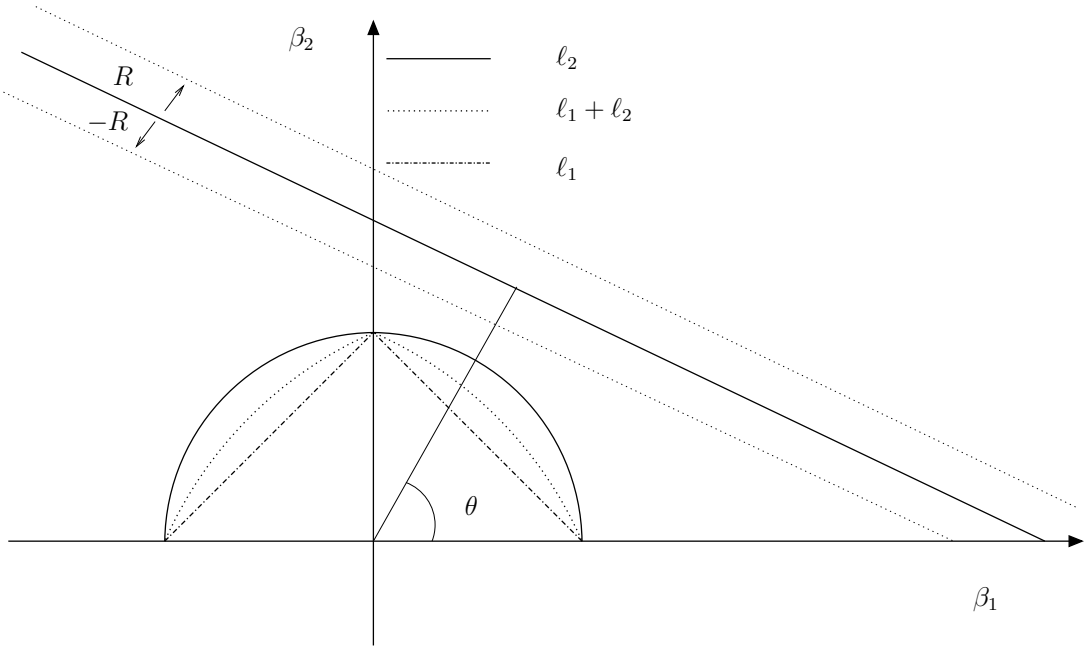

Figure 1: Geometric interpretation of sparsity based regularization for various ℓ_p norms.

4.5 Considerations about ℓ_1 regularization

Generally, there are reasonably good conditions under which ℓ_0 and ℓ_1 regularization are equivalent, which can be seen to an extent from the geometric analysis we performed above. However, such equivalence is not trivial. This is expected, considering that the original problem of ℓ_0 regularization is NP-complete, as we mentioned in section 4.2.

It is also worth making an observation about the regularization parameter λ. In the cases of both ℓ_0 and ℓ_1 regularization, λ controls the sparsity of the resulting function. Thus, λ can be used to control the trade-off between overfitting and sparsity, in a way similar to the case of Tikhonov regularization where λ can control the trade-off between overfitting and oversmoothing.

A theoretical result from the domain of compressive sensing provides a bound on the distance of the solution β^λ obtained from ℓ_1 regularization with regularization parameter λ from the true β^*. Specifically, if $Y = X\beta^* + \xi$, where X and Y are as defined above, $\xi \sim \mathcal{N}(0, \sigma^2 I)$ is a noise term, and the true β^* has at most s nonzero coefficients with $s \leq \frac{n}{2}$, then

$$\left\|\beta^\lambda - \beta^*\right\| \leq Cs\sigma^2 \log p. \tag{15}$$

The constant C in the above bound depends on the matrix X.

We can also draw an interesting parallel between ℓ_1 regularization and support vector machines (SVM). By replacing in (8) the square loss with the hinge loss and the ℓ_1 penalty with the ℓ_2 penalty, we can see that the SVM algorithm also enforces sparsity, however on the examples and not on the variables of the problem. SVM can be seen, then, as an example selection problem, with the solution being a weighted combination of the examples.

However, there is a significant difference between ℓ_1 regularization and the SVM algorithm. Unlike the latter, in ℓ_1 regularization we do not get a minimization functional that depends on the data only through inner products. As a result, we cannot apply the "kernel trick", and thus we

cannot use ℓ_1 regularization as derived above to perform nonlinear variable selection. It is possible that we can still use ℓ_1 regularization to learn sparse linear combinations of nonlinear mappings of the original variables. However, we no longer enjoy the benefits of the theoretical framework of reproducing kernel Hilbert spaces, as is the case in the SVM algorithm. Furthermore, we are constrained to use only finite dimensional dictionaries of nonlinear mappings of the original variables. As an additional disadvantage of this approach, the resulting function will no longer be interpretable with regards to the original variables, and thus will not provide us with insights on the problem, unlike the "linear" ℓ_1 regularization. As the interpretability of the results is a significant aspect of sparsity based regularization, this is a serious disadvantage. In general, nonlinear variable selection is an open problem and constitutes an interesting research direction.

Another noteworthy behavior of ℓ_1 regularization arises when, in the data set, we have variables that are strongly correlated. In this case, from each group of correlated variables, the algorithm selects only one of them. This can create problems in the interpretability of the result as, between two variables of similar behavior and relevance, one is selected and another is rejected. On the other hand, ℓ_1 regularization in this case acts in a way that reduces redundancy, and therefore produces a result that is better from a compression point of view.

Finally, we note that, in the case of the square loss, we have transformed the original non-convex minimization problem of (5) to the convex minimization problem of (8). This is a significant improvement, however our optimization problem remains nonlinear and therefore a closed-form solution cannot be obtained. An even more serious problem is that, although the functional in the minimization problem is convex, it is not strictly convex. As a result, it does not have a unique solution. The existence of multiple solutions with same prediction but different selection properties makes their interpretation problematic.

5 Solving the ℓ_1 regularization problem

For the solution of the nonlinear, (not strictly) convex minimization problem of (8), many optimization approaches can be used. Examples of "off-the-shelf" optimization algorithms that can be used include interior point methods, homotopy methods and coordinate descent.

In this section, we use convex analysis tools to develop a powerful, iterative optimization algorithm, that is well suited to our specific problem. We firstly describe the iteration of the algorithm. Specifically, initially we set $\beta_0^\lambda = 0$. Then, the iteration step of the algorithm continues as

$$\beta_t^\lambda = S_\lambda \left[b_{t-1}^\lambda + \tau X^T \left(Y - X b_{t-1}^\lambda \right) \right] \tag{16}$$

where we observe that the term $X^T \left(Y - X b_{t-1}^\lambda \right)$ is a gradient descent term. In the above iteration, τ is a normalization constant used to ensure that $\tau \|X\| \leq 1$, and S_λ is defined component-wise as follows

$$S_\lambda(\beta^i) = \begin{cases} \beta^i + \lambda/2 & \text{if } \beta^i < -\lambda/2 \\ 0 & \text{if } |\beta^i| \leq \lambda/2 \\ \beta^i - \lambda/2 & \text{if } \beta^i < \lambda/2 \end{cases} \tag{17}$$

S_λ is plotted in figure 2. We can see that S_λ acts as an adaptive threshold, setting very small values to 0 and slightly decreasing all other.

The algorithm can be stopped when either a maximum number of iterations t_{max} or the required precision have been reached. The complexity of the algorithm is $O\left(tp^2\right)$ for each value λ for which it is run. The above algorithm is easy to implement, but can be very expensive computationally. To speedup the algorithm, we can use an adaptive step-size or continuation methods. Moreover, when we need to run the algorithm for multiple values of λ, we make the following two intuitive observations:

1. close values of λ are expected to have nearby solutions, and

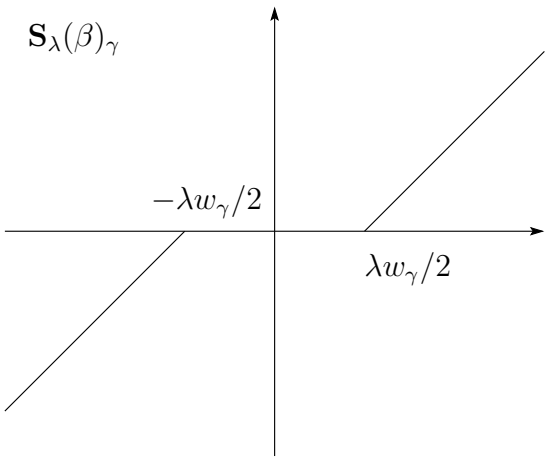

Figure 2: Thresholding function S_λ for iterative thresholding algorithm.

2. solutions obtained for larger λ are more sparse and hence "closer" to the initialization $\beta_0^\lambda = 0$.

Therefore, when the algorithm must be run for multiple values of λ, the two observations above suggest that the following heuristic methods be followed to speed-up its execution:

1. arrange the λ values from the larger to the smaller, and run the algorithm for the various values in this order, and

2. for each λ, initialize the iteration of the algorithm from the solution of the immediately larger value of λ instead of 0 (except of course for the largest λ).

References

[CRT06] Emmanuel J. Candès, Justin Romberg, and Terence Tao, Robust Uncertainty Principles: Exact Signal Reconstruction from Highly Incomplete Frequency Information, IEEE Transactions on Information Theory, vol. 52, no. 2, pp. 489–509, 2006.

[Dono06] David L. Donoho, Compressed Sensing, IEEE Transactions on Information Theory, vol. 52, no. 4, pp. 1289–1306, 2006.

[CDS96] Scott S. Chen, David L. Donoho, and Michael A. Saunders, Atomic decomposition by basis pursuit, SIAM Journal on Scientific Computing, vol. 20, issue 1, pp. 33–61, 1998.

[Tibs96] Robert Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society, Series B (Methodological), vol. 58, no. 1, pp. 267–288, 1996.

[OlFi97] Bruno A. Olshausen and David J. Field, Sparse coding with an overcomplete basis set: A strategy employed by V1?, Vision Research, vol. 37, issue 23, pp. 3311–3325, 1997.

[HyOj00] Aaro Hyvärinen and Erkki Oja, Independent component analysis: algorithms and applications, Neural Networks, vol. 13, issue 4, pp. 411–430, 2000.

[GuEl03] Isabelle Guyon and André Elisseeff, An introduction to variable and feature selection, Journal of Machine Learning Research, vol. 3, pp. 1157–1182, 2003.