A SPLIT BREGMAN METHOD FOR NON-NEGATIVE SPARSITY PENALIZED LEAST SQUARES WITH APPLICATIONS TO HYPERSPECTRAL DEMIXING

Arthur Szlam [*]
The Courant Institute
NYU
New York, NY

Zhaohui Guo [†]
Department of Mathematics
UCLA
Los Angeles, CA

Stanley Osher[‡]
Department of Mathematics
UCLA
Los Angeles, CA

ABSTRACT

We will describe an alternating direction (aka split Bregman) method for solving problems of the form $\min_u ||Au - f||^2 + \eta||u||_1$ such that $u \geq 0$, where A is an $m \times n$ matrix, and η is a nonnegative parameter. The algorithm works especially well for solving large numbers of small to medium overdetermined problems (i.e. $m > n$) with a fixed A. We will demonstrate applications in the analysis of hyperspectral images.

***Index Terms*—** Nonnegative least squares, hyperspectral demixing, over-determined linear systems, split Bregman

1. INTRODUCTION

In this note we will describe an alternating direction (aka split Bregman) method modeled on the approach in [1] for solving problems of the form

$$\begin{array}{l} \min_u ||Au - f||^2 + \eta||u||_1; \\ u \geq 0, \end{array} \tag{1}$$

where A is an $m \times n$ matrix, and η is a nonnegative parameter. The algorithm works especially well for solving large numbers of small to medium overdetermined problems (i.e. $m > n$) with a fixed A. This is different from the usual compressed sensing or sparse coding situation, where $n > m$. The overdetermined problem occurs naturally in some statistical and signal processing applications; for example in the analysis of hyperspectral images.

In this case, each pixel in the image has m bands, corresponding to the intensities of m EM frequencies recorded by a sensor. Assuming that all pixels in a given image are a nonnegative combination of a small collection of n materials with known spectra, and that a mixture of materials with given proportions has a spectrum close to the combination of the known spectra with the given proportions; we are led to (1).

[*]Research supported by DMS-0811203
[†]Research supported by a grant from the department of defense.
[‡]Research supported by a grant from the department of defense.

2. THE ALGORITHM

We replace 1 with the equivalent problem

$$\begin{array}{l} \min_{u,d} ||Au - f||^2 + \eta \sum u; \\ u = P(d), \end{array} \tag{2}$$

where the operator P is defined componentwise by

$$P(d) := \begin{cases} d & d > 0 \\ 0 & \text{otherwise,} \end{cases}$$

Next, we replace the constrained problem 2 with the sequence of unconstrained problems

$$\begin{array}{l} d^{k+1}, u^{k+1} = \min_{d,u} \lambda||Au - f||^2 + \lambda\eta \sum u \\ \qquad\qquad + \ ||u - P(d) - b^k||^2 \\ b^{k+1} = b^k + P(d^{k+1}) - u^{k+1}, \end{array} \tag{3}$$

where λ is a parameter chosen by the user.

This is a classical augmented Lagrangian method [2, 3]. In order to further simplify the subproblems, we split the first subproblem in two, obtaining

$$\begin{array}{l} d^{k+1} = \min_d \lambda||Au^k - f||^2 + \lambda\eta \sum u + ||u^k - P(d) - b^k||^2 \\ u^{k+1} = \min_u \lambda||Au - f||^2 + \lambda\eta \sum u + ||u - P(d^{k+1}) - b^k||^2 \\ b^{k+1} = b^k + P(u^{k+1}) - u^{k+1}. \end{array}$$

Now each subproblem can be solved exactly, giving rise to the iterations

$$\begin{array}{l} d^{k+1} = P(u^k - b^k), \\ u^{k+1} = (\lambda A^T A + I)^{-1}(P(d^{k+1}) + b^k + \lambda A^T f - \lambda\eta \mathbf{1}), \\ b^{k+1} = b^k + P(d^{k+1}) - u^{k+1}. \end{array}$$

This splitting trick also has a long history, see [4] for a nice discussion. We note the similarity of the method presented here and the primal method in [5]; however, there, the choice of splitting is different and the subproblems are not easy to solve exactly.

The method here is most similar to the method presented in [6]. The method there arose from Bregman iterations on

$$\min_u |\mu u|_1 + 1/2||Au - f||^2,$$

where $\mu > 0$ if $u > 0$, and $\mu = \infty$ if $u < 0$; the method here uses Bregman iterations on

$$\min_u ||Au - f||^2 + I(u) + |u|_1$$

where $I(u) = \infty$ if $u < 0$ and $I(u) = 0$ if $u \geq 0$. The algorithmic differences appear in the treatment of the sparsity term, and more importantly for the computation time, the Bregman iterations here do not update the m-vectors f_k, using $A^T f$ for the whole computation. If m is significantly larger than n, this results in a large speedup.

If A is fixed for many f and n is not too large, we can precompute $(\lambda A^T A + I)^{-1}$ and save run time. Also note that using an SVD of A, it is not necessary to compute the full n^2 inverse in the underdetermined case $m < n$. That is, if

$$A = U\Sigma V,$$

where Σ is a $m \times m$ diagonal matrix, U is a $m \times m$ orthogonal matrix, and V is a $m \times n$ matrix with orthogonal rows; and v is a vector, then

$$(\lambda A^T A + I)^{-1} v = V(\lambda \Sigma^2 + I_m)^{-1} V^T v + (v - V^T V v).$$

Up to a constant, this multiplication takes the same amount of time as multiplication by A.

3. EXPERIMENTS

In the next two sections we will test the algorithm on artificial data and real data from hyperspectral images.

3.1. Artificial data

We test the method using randomly generated A and f. More specifically: for $\eta = 0, 1, 10, 100$ we generate 100 512×256 Gaussian normal matrices A and 512×1 Gaussian normal vectors f. We fix $\lambda = 1/500$, and run the algorithm for each instantiation of A and f. In figure 1, we plot the distance between between d^k and d^{200}, for $\eta > 0$, and d^k and the output of the Matlab function lsqnonneg for $\eta = 0$. Note that to precision, d^{200} satisfies the KKT conditions, and because the energy and constraints are convex and differentiable, is a minimum.

The method experimentally converges quite rapidly to the minimizer, on average getting single precision accuracy in less than 40 iterations.

The time to compute $(\lambda A^T A + I)^{-1}$ was on average .0087 seconds; but in applications such as hyperspectral imaging this would only need be done once for all the signals to be processed. The average time to compute $A^T f$ was .0014 seconds, and the average time for an iteration was $3.5 \cdot 10^{-5}$ seconds. With $(\lambda A^T A + I)^{-1}$ precomputed, the average run time to single precision was .003 seconds For comparison, Matlab's command lsqnonneg[1] took on average .38 seconds, more than a hundred times slower. All experiments were carried out on a two core 3.0 Ghz pc running Matlab 7.7.0.

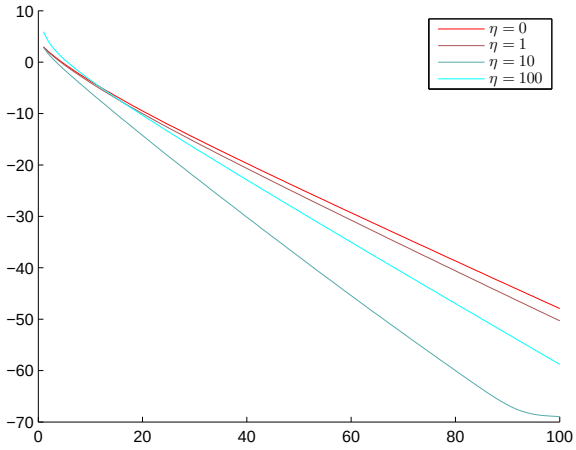


Fig. 1. Distance to the minimizer. The y axis is log scale, the x axis is in iterations. Each iteration took on average 3.5e-5 seconds on a two core 3.0 Ghz pc running Matlab 7.7.0; thus the entire x axis plus the time taken to compute $A^T f$ is .005 seconds. The time taken to compute $(\lambda A^T A + I)^{-1}$ was .014 seconds. For comparison, Matlab's command lsqnonneg took .38 seconds.

3.2. Demixing hyperspectral images

Each pixel of a hyperspectral image is a m dimensional vector corresponding to the m EM frequencies recorded by the sensor. It is often the case that different materials have different characteristic frequency profiles. Here, we will work with images from satelites, and the materials of interest will be asphalt, grass, metal, water, etc. Given a list of n prespecified frequency profiles (called endmembers) corresponding to the materials of interest, we would like to label each of the pixels in the image with its matching material. However, because of the limited resolution of the image, it is often the case that a single pixel contains more than one material, and we need to "demix" the endmembers in the pixels. While a precise demixing could be quite difficult, an approximation can be obtained using the linear mixture model [7, 8, 9], where we assume

$$f \sim Au,$$

where A is the mixing matrix consisting of n m-dimensional endmembers written as columns. In practice, m is often less than or nearly less than n. Because we assume that no pixel

[1]Note that between Matlab release 7.6 and 7.7 lsqnnoneg was significantly improved. We use the newer version.

contains a negative amount of any of the endmembers, we stipulate that $u \geq 0$.

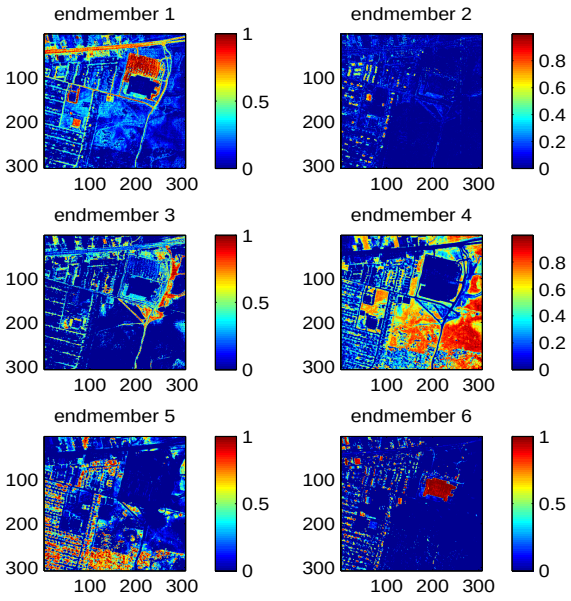
With this assumption, given the endmembers, we can simply run the algorithm described above to get the proportions of endmembers contained in each pixel. Note that because the matrix A is fixed for all the pixels in an image (and in fact, often fixed over many images), as above, we only need compute the $m \times m$ matrix $(A^T A + I)^{-1}$ once and then store it. We will work on the "urban" image available at `http://www.agc.army.mil/Hypercube/`. This image is comprised of 307×307 pixels each with 210 spectral bands. However, several of the bands are corrupted; we remove these, obtaining a 187×307^2 matrix f. We choose 6 endmembers A by hand, as shown in figure 2. In figure 3.2, we see the output of the demixing. Each of the six images corresponds to one of the six endmembers; the brighter the pixel, the more of that endmember makes up the pixel.

The computation times for the $307 \times 307 \times 187$ image are shown in figure 3, which plots iterations of the proposed algorithm against the relative error of the iteration of the proposed algorithm and the output of Matlab's function lsqnnoneg. Each iteration took .026 seconds; the computation of $A^T f$ took .16 seconds, and the computation of $(A^T A + I)^{-1}$ took .003 seconds. The parameter λ was set at $200/||A^T A||_2$. All tests were run on a two core 2.16 Ghz pc running Matlab 7.7.0; for comparison, using Matlab's built in function lsqnnoneg took 27 seconds; thus, to get results to single precision using the proposed algorithm took roughly 1/5 the time of the Matlab function.

We also show results on the Smith island dataset, which we obtained from the U.S. Naval Research Laboratory. An RGB version of the image and location of the chosen endmembers and the the demixing results are shown in figure 3.2. This larger dataset has 946×679 pixels in 126 bands. We remove 14 noisy bands and about 20% of the pixels which are zero in every band, obtaining 495354 pixels each with 112 bands. On this dataset each iteration of the proposed algorithm took about .14 seconds; the computation of $A^T f$ took .41 seconds, and the computation of $(A^T A + I)^{-1}$ took .001 seconds. Again, the parameter λ was set at $200/||A^T A||_2$. By comparision, lsqnonneg took about 128 seconds, so to obtain results to single precision using our method took 1/4 the time of the Matlab function.

The reader may have noticed that the convergence rate was slowest on the Smith Island dataset, and fastest on the artificial data. We have observed in our experiments that the convergence rate of the algorithm seems to depend on the condition number of A, which was roughly 85 for Smith Island, 45 for the Urban dataset, and 5 for the artificial data. We suspect that some sort of preconditioning trick will lead to an even faster algorithm.

(a) The 'Urban' image; the highlighted locations have spectra as in figure 2.



(b) The demixing of the Urban image using the endmembers whose locations are shown in the subfigure above. Each image corresponds to an endmember. The color value in image j at a given pixel x denotes the proportion of endmember j at the location x.

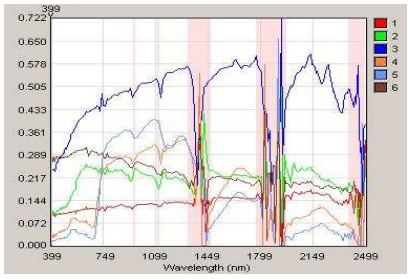


Fig. 2. The 'Urban' endmembers; the pink bands were removed during preprocessing.

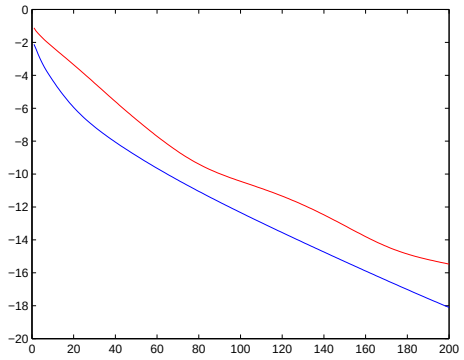


Fig. 3. The x axis is iterations, and the y axis is relative error (in a logarithmic scale) between iteration x of the proposed algorithm and the output of Matlab's function lsqnnoneg on the pixels of the Urban dataset in blue, and the Smith Island dataset in red. Each iteration took .026 seconds on the Urban dataset, and .14 seconds on the Smith Island Dataset

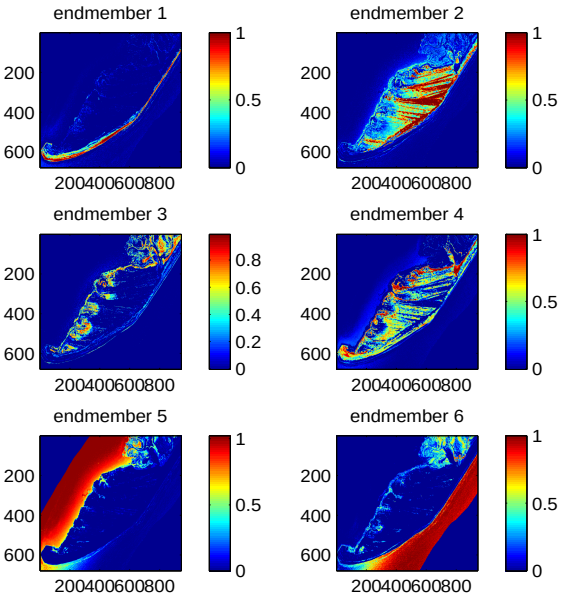
4. CONCLUSIONS

We have presented a novel method for the rapid solution of problem (1) which is especially fast in the overdetermined case, and when the matrix A is fixed for many f. This is precisely the situation arising from the problem of demixing hyperspectral images under the assumption of a linear mixture model. We gave experiments demonstrating the efficiency and accuracy of the method on artificial data and on data from real hyperspectral images.

5. ACKNOWLEDGMENTS

The authors would like to thank Charles Bachmann and the U.S. Naval Research Laboratory for the use of the "Smith Island" data set.

(a) RGB view of the Smith Island dataset.



(b) The demixed Smith Island dataset, according to the endmembers whose locations are shown in the subfigure above. Each image corresponds to an endmember. The color value in image j at a given pixel x denotes the proportion of endmember j at the location x.

6. REFERENCES

[1] T. Goldstein and S. Osher, "The split Bregman method for l1 regularized problems," *SIAM. J. Imaging Sci*, vol. 2, pp. 323–343, 2009.

[2] M. R. Hestenes, "Multiplier and gradient methods," *J. Optimiz. Theory App.*, vol. 4, pp. 303–320, 1969.

[3] M. J. D. Powell, "A method for nonlinear constraints in minimization problems," *Optimization*.

[4] E. Esser, "Applications of lagrangian-based alternating direction methods and connections to split bregman," Tech. Rep. TR09-31, UCLA CAM, 2009.

[5] J. Yang and Y. Zhang, "Alternating direction algorithm for l_1 problems in compressive sensing," Tech. Rep. TR09-37, Rice CAAM, 2009.

[6] Z. Guo, T. Wittman, and S. Osher, "L1 unmixing and its application to hyperspectral image enhancement," in *Proc. SPIE Conference on Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XV*, 2009.

[7] B. Hapke, "Bidirection reflectance spectrocopy. i. theory,," *J. Geophys. Res.*, vol. 86, pp. 3039–3054, 1981.

[8] P. Johnson, M. Smith, S. Taylor-George, and J. Adams, "A semiempirical method for analysis of the reflectance spectra of binary mineral mixtures," *J. Geophys. Res.*, vol. 88, pp. 3557–3561, 1983.

[9] N. Keshave and J.F. Mustard, "unmixing," *Signal Processing Magazine*, vol. 19, pp. 44–57, 2002.