SESOP: Sequential Subspace Optimization Method

Guy Narkiss* Michael Zibulevsky†

Electrical Engineering Dept., Technion – Israel Institute of Technology, Haifa 32000, Israel

Abstract

We present the Sequential Subspace Optimization (SESOP) method for large-scale smooth unconstrained problems. At each iteration we search for a minimum of the objective function over a subspace spanned by the current gradient and by directions of few previous steps. We also include into this subspace the direction from the starting point to the current point, and a weighted sum of all previous gradients, following [Nemirovski-1982]. This safeguard measure provides an optimal worst case convergence rate of order $1/N^2$ (for convex problems), where N is the iteration count. In the case of quadratic objective, the method is equivalent to the conjugate gradients method.

We identify an important class of problems, where subspace optimization can be implemented extremely fast. This happens when the objective function is a combination of expensive linear mappings with computationally cheap non-linear functions. This is a typical situation in many applications, like tomography, signal and image denoising with Basis Pursuit, pattern recognition with Support Vector Machine, and many others. We demonstrate highly competitive numerical results using examples from the mentioned areas.

1 Introduction

We consider an unconstrained minimization of a smooth function

$$\min_{\mathbf{x}\in\mathcal{R}^n} f(\mathbf{x}). \tag{1}$$

When the number of variables is very large, say $n = 10^4 - 10^7$ and more, there is a need for optimization algorithms, for which storage requirement and computational cost per iteration grow not more than linearly in n. An early algorithm of this type is the conjugate gradient (CG) method [2]. It is known that CG worst case convergence rate for quadratic problems is $O(k^{-2})$ (in terms of objective function), where k is the iteration count. This rate of convergence is independent of the problem size and is optimal, *i.e.* it coincides with the complexity of convex smooth unconstrained optimization. The extensions of CG to nonlinear functions by Fletcher-Reeves and Polak-Ribière are no longer worst-case optimal. Nemirovski [6] suggested an optimal method for convex smooth unconstrained optimization. This method sequentially minimizes the objective function over subspaces spanned by the three following vectors:

1. The sum of all previous steps, *i.e.* $\mathbf{d}_k^1 = \mathbf{x}_k - \mathbf{x}_0$.
2. A weighted sum of all previous gradients $\mathbf{d}_k^2 = \sum_{i=0}^{k-1} w_i \mathbf{g}(\mathbf{x}_i)$ with pre-specified weights w_i.
3. The current gradient $\mathbf{g}(\mathbf{x}_k)$.

While Nemirovski's method achieves the optimal complexity in worst case problems, it often behaves poorer than conventional algorithms like non-linear CG or Truncated Newton (TN) [2] in non-worst case problems. In the current work we present a method, which is equivalent to CG in the quadratic case, and often outperforms CG and TN in non-quadratic case, while preserving worst-case optimality.

Our crucial observation is that for many important problems subspace optimization can be implemented extremely fast. This happens, for example, when the objective function is a combination of expensive linear mappings with computationally cheap non-linear functions. It is a typical situation in many applications, like tomography, signal and image processing with Basis Pursuit, pattern recognition with Support Vector Machine, and many others. Another example is constrained optimization, where barrier or penalty aggregate may have this property in linear programming, semidefinite programming. Motivated by this observation we tend to increase the dimensionality of the search

*guyn@siglab.technion.ac.il

†mzib@ee.technion.ac.il

subspaces and use quite accurate subspace optimization. The first additional vector we include is the last step $\mathbf{p}_k = \mathbf{x}_k - \mathbf{x}_{k-1}$. There is a deep reason to do so: Iteration of quadratic CG can be defined as an optimization in the subspace of the current gradient $\mathbf{g}(\mathbf{x}_k)$ and the last step $\mathbf{p}_k$. Preserving this property is a natural way to extend CG to the non-quadratic case. Note that Fletcher-Reeves and Polak-Ribière nonlinear CG method lack this property, which could be very helpful: every iteration is guaranteed to be at least as good as steepest descent. On the other hand, by the *expanding manifold* property, quadratic CG achieves minimum over the subspace of the current gradient and all previous steps and gradients. We can approximate this property including several previous steps and gradients into the optimization subspace.

Let us summarize: Using only 2-d subspace optimizations in directions $\mathbf{g}(\mathbf{x}_k)$ and $\mathbf{p}_k$, we get a method, which coincides with CG, when the problem becomes quadratic. This property is favorable in the proximity of the solution, where the problem has a good quadratic approximation. Globally (in our experience) this method behaves better and is more stable then Polak-Ribière CG. Using two additional Nemirovski directions: $\mathbf{d}_k^1 = \mathbf{x}_k - \mathbf{x}_0$ and $\mathbf{d}_k^2 = \sum_{i=0}^{k-1} w_i \mathbf{g}(\mathbf{x}_i)$ with appropriate weights w_i, we guarantee the worst-case optimality of the method. Including more previous steps and gradients into the optimization subspace helps to further reduce the number of iterations, while moderately increasing the iteration cost.

We also introduce pre-conditioning into this scheme, using pre-multiplication of gradients by an approximate inverse Hessian (in our experiments we have used diagonal approximation). This measure quite often significantly accelerates convergence.

2 SESOP algorithm

Construction of subspace structure In order to define the subspace structure, denote the following sets of directions:

1. **Current gradient**: $\mathbf{g}(\mathbf{x}_k)$ - the gradient at the k'th point $\mathbf{x}_k$.
2. **Nemirovski directions**:

$$\begin{aligned} \mathbf{d}_k^{(1)} &= \mathbf{x}_k - \mathbf{x}_0 \\ \mathbf{d}_k^{(2)} &= \sum_{i=0}^{k} w_i \mathbf{g}(\mathbf{x}_i), \end{aligned} \tag{2}$$

where w_k is defined by

$$w_k = \begin{cases} 1 & \text{for } k = 0 \\ \frac{1}{2} + \sqrt{\frac{1}{4} + w_{k-1}^2} & \text{for } k > 0. \end{cases} \tag{3}$$

3. **Previous directions**:

$$\mathbf{p}_{k-i} = \mathbf{x}_{k-i} - \mathbf{x}_{k-i-1}, \quad i = 0, \ldots, s_1. \tag{4}$$

4. **Previous gradients**:

$$\mathbf{g}_{k-i}, \quad i = 1, \ldots, s_2. \tag{5}$$

The mandatory direction 1 and any subset of directions 2 - 4 can be used to define the subspace structure.

Algorithm summary Let $\mathbf{D}$ be a matrix of the chosen M (column) directions described above, and α a column vector of M coefficients. On every iteration we find a new direction $\mathbf{D}\alpha$ in the subspace spanned by the columns of $\mathbf{D}$. The algorithm is summarized as follows:

1. Initialize $\mathbf{x}_k = \mathbf{x}_0$, $\mathbf{D} = \mathbf{D}_0 = \mathbf{g}(\mathbf{x}_0)$.
2. Normalize the columns of $\mathbf{D}$.
3. Find
$$\alpha^* = \underset{\alpha}{\operatorname{argmin}} f\big(\mathbf{x}_k + \mathbf{D}\alpha\big). \tag{6}$$
4. Update current iterate:
$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{D}\alpha^*. \tag{7}$$
5. Update matrix $\mathbf{D}$ according to the chosen subspace directions.
6. Repeat steps 2 - 5 until convergence.

Reduced computations for subspace minimization Consider a function of the form

$$f(\mathbf{x}) = \varphi(\mathbf{A}\mathbf{x}) + \psi(\mathbf{x}). \tag{8}$$

Such functions are very common in many applications. The multiplications $\mathbf{A}\mathbf{x}$ and $\mathbf{A}^T\mathbf{y}$ are usually the most computationally expensive operations for calculating the function f and its gradient. Methods based on subspace optimization often iterate the multi-dimensional minimizer incrementally in the form

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \sum_{i=1}^{M} \alpha_i \mathbf{r}_i, \tag{9}$$

where the coefficients α_i, the directions $\mathbf{r}_i$ and the number of directions M are determined according to the specific optimization scheme. Such a framework allows us to save a large part of the matrix-vector multiplications originally needed for calculation of the objective function value (8). The term $\mathbf{A}\mathbf{x}_{k+1}$ can be broken into

$$\begin{aligned}\mathbf{A}\mathbf{x}_{k+1} &= \mathbf{A}\mathbf{x}_k + \mathbf{A}\sum_{i=1}^{M}\alpha_i\mathbf{r}_i \\ &= \mathbf{A}\mathbf{x}_k + \sum_{i=1}^{M}\alpha_i\mathbf{A}\mathbf{r}_i \\ &= \mathbf{v}_0 + \sum_{i=1}^{M}\mathbf{v}_i\alpha_i,\end{aligned} \tag{10}$$

where $\mathbf{v}_i = \mathbf{A}\mathbf{r}_i$. For each new direction $\mathbf{r}_i$ we need to calculate and save one vector ($\mathbf{v}_i$). For line search operation along a single direction, or subspace minimization along several directions, there is no need to perform any matrix-vector multiplication, since the function and its gradient with respect to α are gained using the precalculated set of vectors $\mathbf{v}_i$.

3 Computational Experiments

We compared the performance of several algorithms with several large-scale optimization problems: Computational Tomography (CT), Basis Pursuit (BP), and Support Vector Machine (SVM). Details appear in [4], [5]. We compared CPU runtime and the number of matrix-vector multiplication between all methods.

Computerized Tomography (CT) We solved the two dimensional straight-ray transmission tomography problem, for sparse images of sizes 128^2, 256^2 pixels. Our objective is minimizing following penalized least squares function

$$\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2 + \mu\|\mathbf{x}\|_1, \tag{11}$$

where $\mathbf{A}$ represents the Radon transform, $\mathbf{x}$ is the unknown original image, $\mathbf{y}$ is the observed noisy projection data and μ is a regularization parameter. In order to use smooth optimization, we use smooth approximation of the l_1-norm. Figure 1 presents the inaccuracy in objective function as a function of iteration number. Iteration numbers and runtime are summarized in Table 1. SESOP method outperforms the CG method by 25% less iterations.

Image size	Method	Convergence iter	Convergence time	Good results iter	Good results time
128^2	SESOP	138	28.4	36	8.21
	CG	294	118.6	48	13
	TN	2632	355	2053	280
256^2	SESOP	182	166	41	39.4
	CG	377	627	54	63.2
	TN	5050	3208	2729	1728

Table 1: Sparse tomography: Iterations and CPU runtime [sec] to convergence ($\|\nabla f\| \leq 10^{-4}$), and to 'good results' (PSNR reached 0.01dB from the final PSNR).

Basis Pursuit (BP) We bring an example of image de-noising in BP framework [1] using contourlets dictionary (see for example [3]). We solve the following problem

$$\min_{\alpha} \frac{1}{2\sigma_n^2}\|\mathbf{\Phi}\alpha-\mathbf{y}\|_2^2 + \sum_i \lambda_i|\alpha_i|. \tag{12}$$

where $\mathbf{y} = \mathbf{x}+\mathbf{n}$ is the observed sum of the original picture $\mathbf{x}$ (parsed column-wise) with Gaussian noise $\mathbf{n}$ with variance σ_n^2. We assume $\mathbf{x} = \mathbf{\Phi}\alpha$ where $\mathbf{\Phi}$ is a "synthesis" operator (equivalent to a matrix of basis functions in its columns), and sparsity of the original picture's coefficients. The experiment was conducted on the popular image 'peppers' (see [4]). For picture size of 256^2 pixels, the number of coefficients is $87,296$. Inaccuracy in objective function is shown in Figure 2. Iteration numbers and runtime are summarized in Table 2. Subspace method converged faster than CG, although for identical number of iterations for 'Good results' the CG was faster (pictures of sizes 256^2, 512^2).

3.1 Other Numerical Experiments with SESOP

In [5] we bring more numerical results with SESOP in the area of pattern recognition using Support Vector Machines. Just to summarize briefly: we have solved six problems with $10^3 - 10^6$ variables. SESOP was consistently faster than Nesterov method, CG and TN, on average outperforming TN ten times and CG about two times.

4 Conclusions

SESOP is an efficient tool for large-scale unconstrained optimization. The main advantages of SESOP are op-

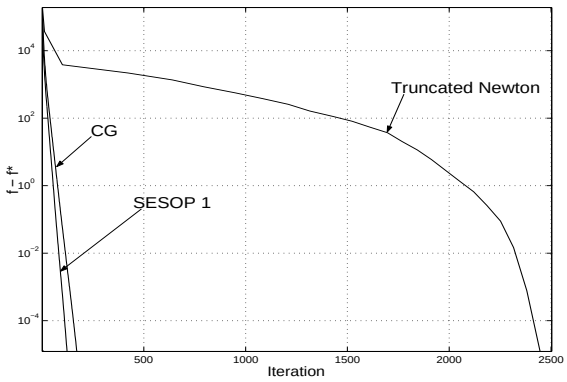

Figure 1: Sparse tomography: Inaccuracy in objective function [log scale] with iterations

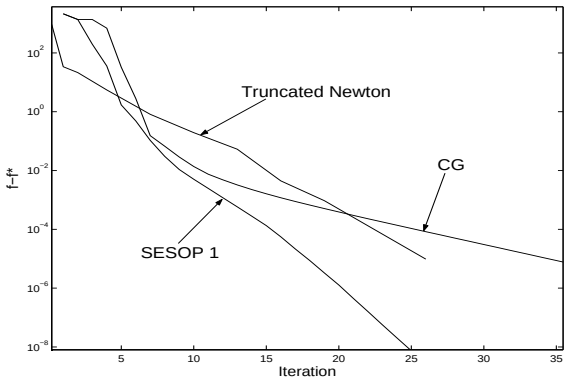

Figure 2: Basis pursuit: Inaccuracy in objective function [log scale] with iterations

Image size	Method	Convergence iter	Convergence time	Good results iter	Good results time
128^2	SESOP	25	5.54	5	1.06
	CG	55	8.22	7	1.28
	TN	56	7.88	12	5.52
256^2	SESOP2	83	74.36	8	8.44
	CG	335	211	8	7.18
	TN	306	126.99	3	3.68
512^2	SESOP2	75	382.23	8	51.7
	CG	268	984.15	10	45.13
	TN	156	432.18	9	184.15

Table 2: Basis Pursuit de-noising example: Iterations and CPU runtime [sec] to convergence ($\|\nabla f\| \leq 10^{-4}$), and to 'good results' (PSNR reached 0.01dB from the final PSNR).

timal worst-case complexity for smooth convex unconstrained problems, low memory requirements and low computation load per iteration. Unconstrained optimization is a building block for many constrained optimization techniques, which makes SESOP a promising candidate for embedding into many existing solvers.

Acknowledgment. We are grateful to Arkadi Nemirovski for his most useful advice and support. Our thanks to Boaz Matalon for the Contourlet code and explanations. This research has been supported in parts by the "Dvorah" fund of the Technion and by the HASSIP Research Network Program HPRN-CT-2002-00285, sponsored by the European Commission. The research was carried out in the Ollendorff Minerva Center, funded through the BMBF.

References

[1] S. S. CHEN, D. L. DONOHO, AND M. A. SAUNDERS, *Atomic decomposition by basis pursuit*, SIAM Journal on Scientific Computing, 20 (1999), pp. 33–61.

[2] P. E. GILL, W. MURRAY, AND M. H. WRIGHT, *Practical Optimization*, Academic Press, 1981.

[3] B. MATALON, M. ZIBULEVSKY, AND M. ELAD, *Improved denoising of images using modeling of a redundant contourlet transform*, in SPIE, Optics and Photonics, Wavelets XI, to be published, 2005.

[4] G. NARKISS AND M. ZIBULEVSKY, *Sequential subspace optimization method for large-scale unconstrained optimization*, technical report CCIT No. 559, Technion - The Israel Institute of Technology, faculty of Electrical Engineering, Haifa, Israel, 2005.

[5] ——, *Support vector machine via sequential subspace optimization*, technical report CCIT No. 557, Technion - The Israel Institute of Technology, faculty of Electrical Engineering, Haifa, Israel, 2005.

[6] A. NEMIROVSKI, *Orth-method for smooth convex optimization (in russian).*, Izvestia AN SSSR, Ser. Tekhnicheskaya Kibernetika (the journal is translated to English as Engineering Cybernetics. Soviet J. Computer & Systems Sci.), 2 (1982).