A MULTILEVEL APPROACH FOR l_1 REGULARIZED CONVEX OPTIMIZATION WITH APPLICATION TO COVARIANCE SELECTION

Abstract. We present a multilevel framework for solving l-1 regularized convex optimization problems. Such l-1 regularization is utilized to find sparse minimizers of convex functions, and is mostly known for its use in the LASSO problem, where the l-1 norm is applied to regularize a quadratic function. Our multilevel framework was initially suggested for the LASSO problem, and proved to be highly efficient. In this work we generalize this framework, and use it for solving the Covariance Selection problem, where a sparse inverse covariance matrix is estimated from a only few samples of a multivariate normal distribution. Numerical experiments demonstrate the potential of this approach, for both medium and large scale problems.

1. Introduction

[5] Our problem,

$$\min_{\mathbf{x}\in\mathbb{R}^n} F(\mathbf{x}) = \min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) + \lambda\|\mathbf{x}\|_1, \tag{1}$$

with $f(\mathbf{x})$ being a strictly convex problem, and λ a scalar parameter that balances between sparsity and adherence to $f(\mathbf{x})$. Generally, a larger parameter λ yields a sparser minimizer $\mathbf{x}^*$, but a higher value for $f(\mathbf{x}^*)$. Although this problem is an unconstrained convex optimization problem, traditional optimization methods, such as gradient descent or quasi-Newton methods, tend to be slow due to the discontinuity of the gradient, which arises from using the l_1 norm. Therefore, various computational optimization methods were developed for the task. The most common methods are the so-called "iterative shrinkage" or "iterative soft thresholding (IST)" methods that are often used together with some accelerations. In this work we adopt the common practice of seeking any one of those solutions and refer to it as "the minimizer" of (1), denoted by $\mathbf{x}^*$.

This paper introduces a straightforward multilevel method for l_1 penalized inverse covariance selection problems like (1), based on the main concept of classical algebraic multigrid methods [2]; that is, we accelerate the convergence of simple iterative methods for (1) using a nested hierarchy of smaller versions of the problem. Multigrid methods are commonly applied to linear systems arising from discretization of partial differential equations as well as other ill-conditioned systems. In many cases algebraic multigrid methods enable us to treat such problems effectively regardless of their condition number. This is done by projecting the original problem to a lower-dimensional subspace that contains the error components that are not treated effectively by standard iterative methods, such as Jacobi and Gauss-Seidel. Then, these error components are corrected by solving a lower-dimensional

Date: April 24, 2014.

Department of Computer Science, The Technion—Israel Institute of Technology, Haifa 32000, Israel.

problem. This correction together with the standard iterative methods serve as two complementary processes which combine to yield an effective solver. The idea of introducing multigrid-like methods for (1) has yet to be explored and it has great potential. In this work we follow the idea of "multiplicative correction" multigrid methods, which exploit a hierarchy of approximate operators that evolve with the solution process, eventually becoming exact—see [1, 4] and references therein.

In classical multigrid methods, the aim is to define a multilevel solver with optimal *asymptotic* convergence behavior, because the asymptotic convergence rates of simpler iterative solvers tend to be slow for problems of interest. However, the present problem is different as the main challenge is finding the non-zero elements of the minimizer $\mathbf{x}^*$ and its sign-pattern. Therefore, our algorithm is different from the classical multigrid approach. At each iteration (called a "multilevel V-cycle") it reduces the dimension of the problem and creates a multilevel hierarchy of smaller and smaller problems. We take advantage of the typical sparsity of $\mathbf{x}$ and reduce the dimension of the problem (1) by ignoring ostensibly irrelevant variables. That is, each low-level problem is defined by (1), restricted to a specially chosen subset variables, resulting in a nested hierarchy of problems. It then performs sub-space correcting shrinkage sweeps over each of the low dimensional problems in turn, that aim to activate the variables that comprise the support of a true minimizer. Under suitable conditions, our algorithm converges to the global minimizer of (1)—we do not compromise solution quality in return for improved performance.

2. Multilevel Iterated shrinkage

A simple iterated shrinkage method for solving (1) is of the form

$$\mathbf{z} = \mathcal{S}_{\lambda/c}\left(\mathbf{x}^k - \frac{1}{c}\nabla f(\mathbf{x}^k)\right), \tag{2}$$

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha(\mathbf{z} - \mathbf{x}^k), \tag{3}$$

where $\mathbf{x}^k$ is the approximate solution at the k-th iteration, $\alpha > 0$ is a line-search scalar, $c > 0$ is method dependent, and

$$\mathcal{S}_q(t) = \operatorname{sign}(t) \cdot \max(0, |t| - q) \tag{4}$$

is the "soft shrinkage" function, so dubbed because the size of the argument t is reduced by q (or set to zero if $q > |t|$).

We next describe our new multilevel approach for solving (1). At each iteration, called a "V-cycle", we define a hierarchy of reduced problems, referred to as *low-level problems*. Each low-level problem is defined by (1), restricted to a specially chosen subset of variables, and in each V-cycle we traverse the entire hierarchy of levels. We iteratively repeat these V-cycles, reducing the functional of (1) at each one, until some convergence criterion is satisfied. A precise description is given in the following sections in a two-level framework, with the extension to the multi-level framework obtained by recursion. In this description, all elements that are related to the low-level problem are denoted by a subscript c.

2.1. Definition of the low-level problem.

In this subsection we define the reduced problem given its designated subset of variables, $\mathcal{C} \subset \{1, ..., m\}$, while the choice of $\mathcal{C}$ will be discussed later. Given $\mathcal{C}$, we define a so-called prolongation matrix $P \in \mathbb{R}^{m\times|\mathcal{C}|}$, that transfers a low-level vector $\mathbf{x}_c \in \mathbb{R}^{|\mathcal{C}|}$ into an upper-level vector $\mathbf{x} \in \mathbb{R}^m$ by the relation $\mathbf{x} = P\mathbf{x}_c$. We choose P to be a zero-filling operator, which zeros the elements of $\mathbf{x}$ that do not belong to $\mathcal{C}$, while retaining the values of $\mathbf{x}_c$ in the elements that do belong to $\mathcal{C}$.

Next, we restrict (1) onto the atoms in $\mathcal{C}$, or more generally, onto the range of P. That is, we substitute $P\mathbf{x}_c$ for $\mathbf{x}$ in the objective (1), and get the new problem:

$$\min_{\mathbf{x}_c \in \mathbb{R}^{|\mathcal{C}|}} F_c(\mathbf{x}_c) \equiv \min_{\mathbf{x}_c \in \mathbb{R}^{|\mathcal{C}|}} F(P\mathbf{x}_c) = \min_{\mathbf{x}_c \in \mathbb{R}^{|\mathcal{C}|}} f(P\mathbf{x}_c) + \lambda \|P\mathbf{x}_c\|_1, \tag{5}$$

which has only $|\mathcal{C}|$ degrees of freedom. Since our P is zero-filling, we have that $\|P\mathbf{x}_c\|_1 = \|\mathbf{x}_c\|_1$ holds for all $\mathbf{x}_c$, and therefore we can write

$$\min_{\mathbf{x} \in \mathbb{R}^{|\mathcal{C}|}} F_c(\mathbf{x}_c) = \min_{\mathbf{x}_c \in \mathbb{R}^{|\mathcal{C}|}} f_c(\mathbf{x}_c) + \lambda \|\mathbf{x}_c\|_1, \tag{6}$$

where $f_c(\mathbf{x}_c) = f(P\mathbf{x}_c)$, is the reduced sub-problem of the upper-level $f(\mathbf{x})$, restricted only to the variables in $\mathcal{C}$, such that the rest of the variables remain zeros. Note that if $\mathcal{C}$ contain the support of the true minimizer of (1), and (6) is solved exactly, then $P\mathbf{x}_c$ is in fact a solution of (1). Furthermore, because this problem is similar to (1), we can recursively extend this two-level framework to multi levels.

2.2. **Choosing the low-level variables.** Our low-level definition above suggests that we need to select a subset of low-level variables, $\mathcal{C}$, that is as likely as possible to contain the support of the true minimizer. Therefore, for choosing $\mathcal{C}$ we use the approximate solution at the k-th iteration, $\mathbf{x}^k$, which is the best one currently available. Let

$$\text{supp}(\mathbf{x}) = \{i : x_i \neq 0\},$$

denote the support of any vector $\mathbf{x}$. Then evidently, if $\text{supp}(\mathbf{x}^k) \subseteq \mathcal{C}$, then $\mathbf{x}^k$ is in the range of P. Indeed, $\mathbf{x}^k = P\mathbf{x}_c$ where $\mathbf{x}_c$ is the vector $\mathbf{x}^k$ restricted to the indices in $\mathcal{C}$. Therefore, we start by requiring $\text{supp}(\mathbf{x}^k) \subseteq \mathcal{C}$, so that by (5)-(6) we have that $F(\mathbf{x}^k) = F_c(\mathbf{x}_c)$ holds. This implies that the prolongation matrix P changes during the iterations, depending on $\mathbf{x}^k$.

Next, we decide on the additional atoms in $\mathcal{C}$, besides those in $\text{supp}(\mathbf{x}^k)$, aiming to limit its size to n_c, which is some part of all relevant variables. If $|\text{supp}(\mathbf{x}^k)| \geq n_c$, then we choose $\mathcal{C} = \text{supp}(\mathbf{x}^k)$. Otherwise (the common case) we add $n_c - |\text{supp}(\mathbf{x}^k)|$ atoms that are currently not in $\text{supp}(\mathbf{x}^k)$, and yet have a relatively good chance of being in the support of the true solution $\mathbf{x}^*$. These correspond to variables i with a relatively large value of $|(\nabla f(\mathbf{x}^k))_i|$, since including them in the support reduces the first term in the functional of (1) more significantly per given increase in the second term.

2.3. **Definition of the multi-level V-cycle.** For solving (1), we repeat

$$\mathbf{x}^{k+1} = \textbf{V-cycle}(f(\mathbf{x}), \mathbf{x}^k, \nu) \tag{7}$$

iteratively, until some convergence criterion is satisfied. The multilevel V-cycle() procedure, along with its parameters, is defined in Algorithm 3. The algorithm creates a reduced version of the problem (1) as described above, and then treats it recursively, yielding a hierarchy of smaller and smaller problems. The recursion is terminated (Step 3a) when one of the following happens. If $|\text{supp}(\mathbf{x})| \geq n_c$— then we cannot reduce the problem further. In this case we choose $\mathcal{C} = \text{supp}(\mathbf{x})$, and solve the problem (6) directly. The second base case is when the problem becomes sufficiently small and can be solved easily. In practice, we choose a minimal number of allowable columns n_{min}, and if $|\mathcal{C}| < 2n_{min}$ then we process (6) directly rather than continuing recursively.

The algorithm uses iterated shrinkage methods as so-called “relaxations”—the usual name for the iterations employed within multilevel algorithms—carrying out ν such relaxations at each level ($\nu = 1$ in our tests). All shrinkage methods of the form (2), as well as most other shrinkage methods, can be incorporated into this multilevel approach.

```
Algorithm: $\mathbf{x} \leftarrow$ V-cycle$(f(\mathbf{x}), \mathbf{x}, \nu)$
%Iterative Shrinkage method: Relax$(f(\mathbf{x}), \mathbf{x})$.
%Number of relaxations at each level: $\nu$.
%Minimal number of columns allowed: $n_{min}$.
(1) Choose the low-level variables $\mathcal{C}$ and define the prolongation $P$.
(2) Define the low-level problem $f_c(\mathbf{x}_c)$
    approximation $\mathbf{x}_c = P^T\mathbf{x}$.
(3) If $\mathcal{C} = \text{supp}(\mathbf{x})$ or $|\mathcal{C}| < 2n_{min}$,
        (a) Solve the lowest-level problem (6).
    Else $\mathbf{x}_c \leftarrow$ V-cycle$(f_c(\mathbf{x}_c), \mathbf{x}_c, \nu)$ % Recursive call
(4) Prolong solution: $\mathbf{x} \leftarrow P\mathbf{x}_c$ % Solution update.
(5) Apply $\nu$ relaxations: $\mathbf{x} \leftarrow$ Relax$(f(\mathbf{x}), \mathbf{x})$.
```

Algorithm 1: V-cycle for l_1 penalized LS minimization

In terms of cost, if we reduce the number of unknowns by a factor of two at each level, then the total cost of a V-cycle (excluding the treatment of the lowest level) is only about twice the cost of ν iterated shrinkage relaxations on the highest level. This means that, although we include a relatively large fraction of the atoms in the next low level, the cost of the entire V-cycle remains relatively small, possibly excluding the lowest level solution. The latter is a key component of our algorithm, and it will be discussed later.

3. Application for Covariance Selection

In modern settings statistical problem are high-dimensional, where the number of parameters is large when compared to the number of observations.

Specifically, given i.i.d samples $\{\mathbf{x}_1, ..., \mathbf{x}_k\}, \mathbf{x}_i \sim \mathcal{N}(\mu, \Sigma) \in \mathbb{R}^n$, and $k < n$.

$$P(\mathbf{x}; \mu, \Sigma) \propto \frac{1}{\sqrt{\det(\Sigma)}} \exp\left(\frac{1}{2}(\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu)\right)$$

Standard maximum likelihood estimation of parameters:

$$\hat{\mu} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i$$

$$\hat{\Sigma} = S = \frac{1}{m-1} \sum_{i=1}^{m} (\mathbf{x}_i - \hat{\mu})(\mathbf{x}_i - \hat{\mu})^T \in \mathbb{R}^{n \times n}$$

However, if $m < n$, S is not full-rank (Σ is full-rank). The task is to estimate the inverse covariance matrix Σ^{-1}.

In this section, and for the rest of the paper, we refer to the Covariance Selection problem. The standard approach to this problem is the maximum likelihood method (a method of estimating the parameters of a statistical model from a data set and given a statistical model), which requires maximization of the log-likelihood function:

$$\min_{A \succ 0} F(A) = \min_{A \succ 0} \{-\log\det(A) + \mathrm{tr}(AS) + \lambda \|A\|_1\}, \tag{8}$$

where $A \in \mathbb{R}^{n\times n}$ is a symmetric matrix of variables, the term $\|A\|_1 = \sum_{i,j} |A_{ij}|$ acts as a sparsity promoting regularization, and $S = \frac{1}{k}\sum_{i=1}^{k} \mathbf{x}_i \mathbf{x}_i^T \in \mathbb{R}^{n\times n}$ is our empirical covariance estimate. This problem correspond to the above formulation in (1) for $f(A) = -\log\det(A) + \mathrm{tr}(AS)$, which is strictly convex, and its gradient is given by

$$\nabla f(A) = -A^{-1} + S. \tag{9}$$

The recent approaches use Newton's method to solve (8), employing a quadratic approximation for the smooth function f, while leaving the l_1 regularization as is. More precisely, given an approximate solution matrix A^k, they solve

$$\min_{\Delta \in \mathbb{R}^{n\times n}} \tilde{F}(A^k + \Delta) = f(A^k) + \langle \nabla f(A^k), \Delta \rangle + \frac{1}{2}\langle \Delta, \nabla^2 f(A^k)\Delta \rangle + \lambda \|A^k + \Delta\|_1, \tag{10}$$

where $\nabla^2 f(A^k) = (A^k)^{-1} \otimes (A^k)^{-1}$ is the Hessian of f at A^k, and $\otimes$ is the Kronecker product. For the case of (8), the above approximation is given by

$$\min_{\Delta \in \mathbb{R}^{n\times n}} \tilde{F}(A^k + \Delta) = f(A^k) + \mathrm{tr}((S - (A^k)^{-1})\Delta) + \frac{1}{2}\mathrm{tr}(\Delta (A^k)^{-1} \Delta (A^k)^{-1}) + \lambda \|A^k + \Delta\|_1, \tag{11}$$

which is of the same form as the LASSO problem mentioned earlier. A coordinate descent update for the variable $A_{ij}, i < j$ that preserves symmetry ($(A^k)^{-1} = W^k$) [3] :

$$\mu = \mathcal{S}_{\frac{\lambda}{W_{ij}^2 + W_{ii}W_{jj}}}(A_{ij} + \Delta_{ij} - \frac{S_{ij} - W_{ij} + w_i^T \Delta w_j}{W_{ij}^2 + W_{ii}W_{jj}}) - A_{ij} - \Delta_{ij} \tag{12}$$

Nevertheless, we do not update each entry of A. We define *ActiveSet* to be the entries from A for the coordinate descent update. The entries are chosen based on the value of the gradient. Specifically, $A_{ij} \in$ *ActiveSet* if $A_{ij} \neq 0$ or $S_{ij} - A_{ij}^{-1} > \lambda$.

Now we can introduce the known QUIC algorithm, which stands for QUadratic Inverse Covariance:

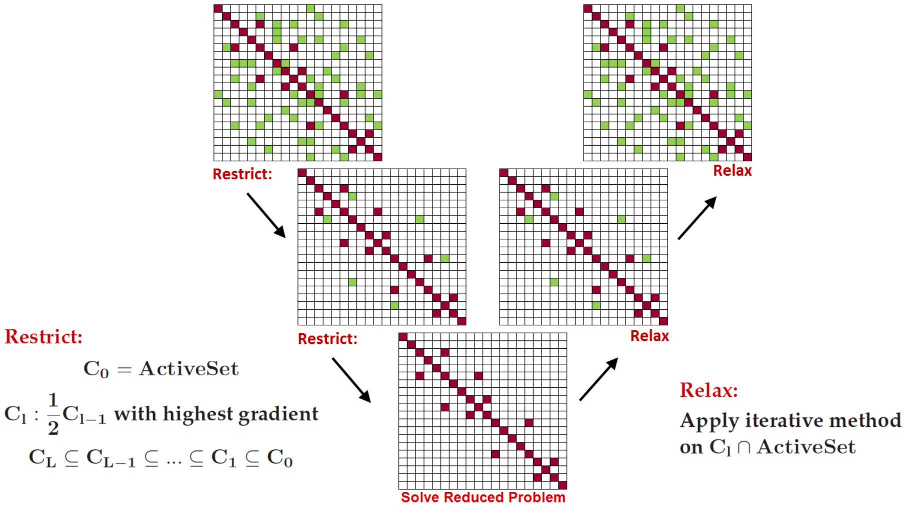
```
Algorithm: $A^{k+1} \leftarrow$ CovSelNewtonIteration($A^k$)
%Iterative Shrinkage for the LASSO problem (12): $IST(\Delta, S, (A^k)^{-1})$.;
(1) Calculate $W = (A^k)^{-1}$;
(2) Define the fixed and free ($\mathcal{F}_k$) sets.;
(3) Solve (12) restricted to ($\mathcal{F}_k$) using the method $IST(A)$.
    Denote this solution by $\Delta$;
(4) Define $A^{k+1} = A^k + \alpha\Delta$ via linesearch: find $\alpha$ s.t $A^{k+1} \succ 0$
    and $F(A^{k+1})$ is approximately minimized.
```

Algorithm 2: Covariance selection via Newton's method

We accelerate the convergence of QUIC by using a nested hierarchy of smaller versions of the problem. At each iteration, called a V-cycle, our algorithm reduces the dimension of the problem and creates a multilevel hierarchy of smaller and smaller problems, low-level problems. It then performs shrinkage sweeps over each of the low dimensional problems in turn, that aim to activate the entries which comprise the support of a true minimizer. We iteratively repeat these V-cycle until some convergence criterion is satisfied

```
Algorithm: $A^{k+1} \leftarrow$ V-cycle($A^k, \mathcal{C}, \mathcal{G}, \nu$)
%Initial $\mathcal{C}$ ActiveSet($A^k$), $\mathcal{G}$ is the values.
%Iterative Shrinkage method for covariance selection: Relax($A, \mathcal{C}$).
%Number of relaxations at each level: $\nu$.
%Minimal number of variables: $n_{min} = 4n$ or every other number $> n$).
(1) Choose $\mathcal{C}_{new}$ out of $\mathcal{C}$ according to $\mathcal{G}$. Either $|\mathcal{C}|/2$ or the support if it is
    larger than that.
(2) Define the low-level problem $f_c(A(\mathcal{C}_{new}))$.
(3) If $\mathcal{C}_{new} = \text{supp}(A)$ or $|\mathcal{C}_{new}| < 2n_{min}$,
        (a) Solve the lowest-level problem $2 \times Relax(A, \mathcal{C}_{new})$.
    Else $A \leftarrow$ V-cycle($A, \mathcal{C}_{new}, \nu$) % Recursive call
(4) Apply $\nu$ relaxations: $A \leftarrow Relax(A, \mathcal{C} \cap ActiveSet(A))$.
```

Algorithm 3: V-cycle for Covariance Selection

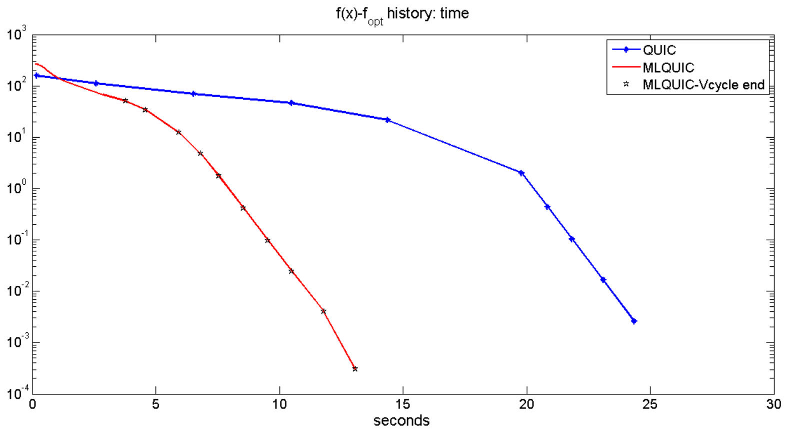

Restrict:
Relax
Restrict:
Relax
Restrict:
$C_0 = $ ActiveSet
$C_l : \frac{1}{2} C_{l-1}$ with highest gradient
$C_L \subseteq C_{L-1} \subseteq \ldots \subseteq C_1 \subseteq C_0$
Restrict:
Relax
Relax:
Apply iterative method on $C_l \cap$ ActiveSet
Solve Reduced Problem

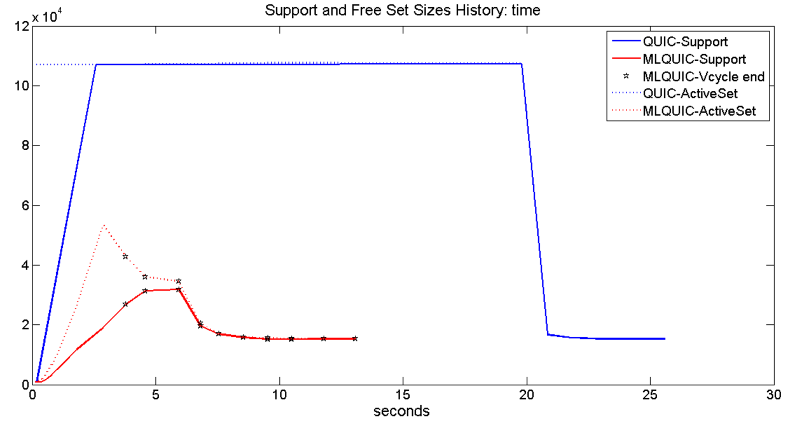
4. Numerical Results

The following graphs show the convergence to F_{opt} and the the amount of variables (*ActiveSet* and *Support* sizes) that were used while solving the problem. Each graph compares QUIC and MLQUIC. The data was taken from real experiments in biology.

4.1. **Arabidopsis.** A small flowering plant with a relatively short life cycle. A popular model organism in plant biology and genetics. Relatively small genome 135 Mbp. It was the first plant to have its genome sequenced, and is a popular tool for understanding themolecular biologyof many plant traits, including flower development and light sensing.

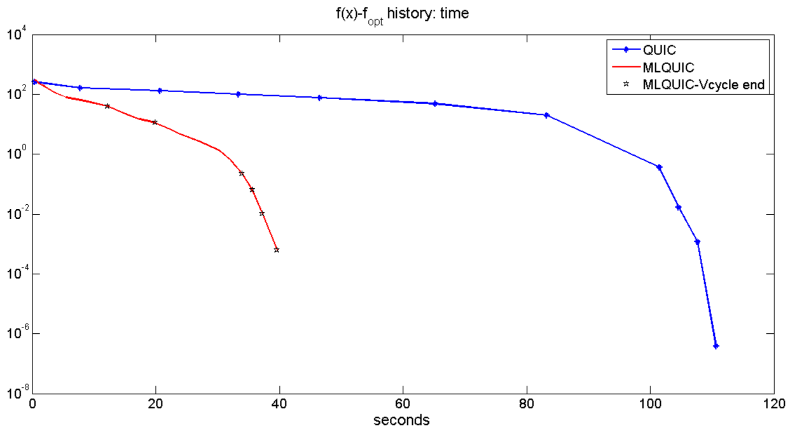
$S \in \mathbb{R}^{834 \times 834}$, number of samples: 118, $\lambda = 0.27$

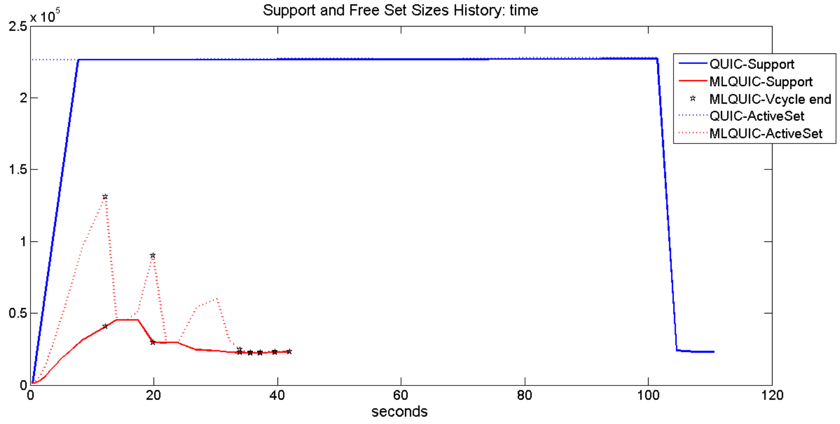




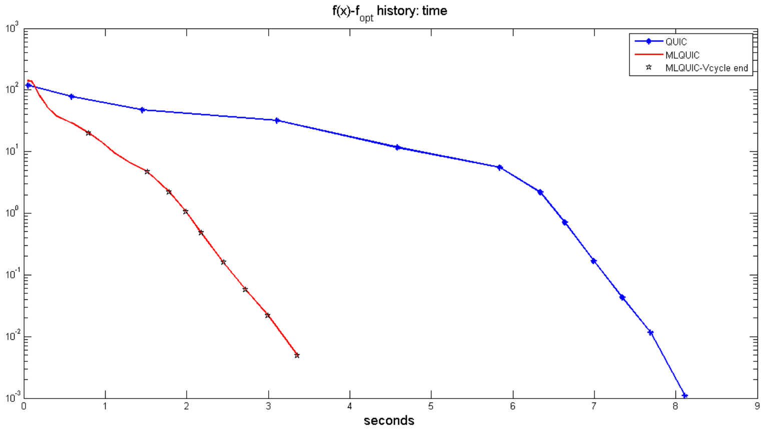
4.2. **Leukemia.** A type ofcancerin which thebloodorbone marrow characterized by an abnormal increase of immature white blood cells. Like other cancers, it results frommutationsin theDNA that disrupting the regulation of cell death, differentiation or division.

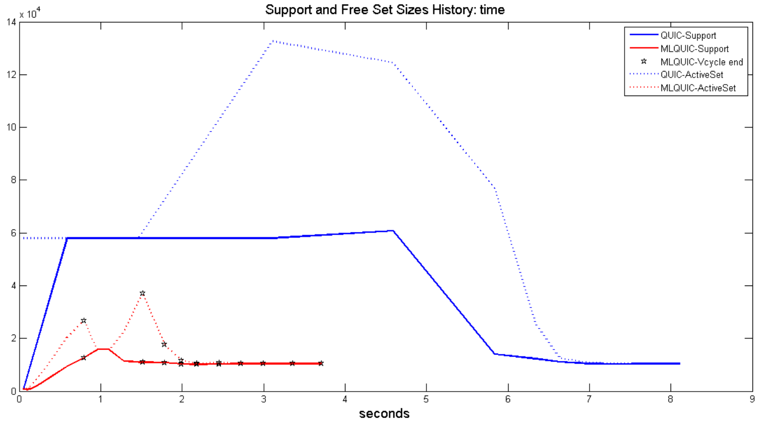
$S \in \mathbb{R}^{1255\times1255}$, number of samples: 72, $\lambda = 0.3$



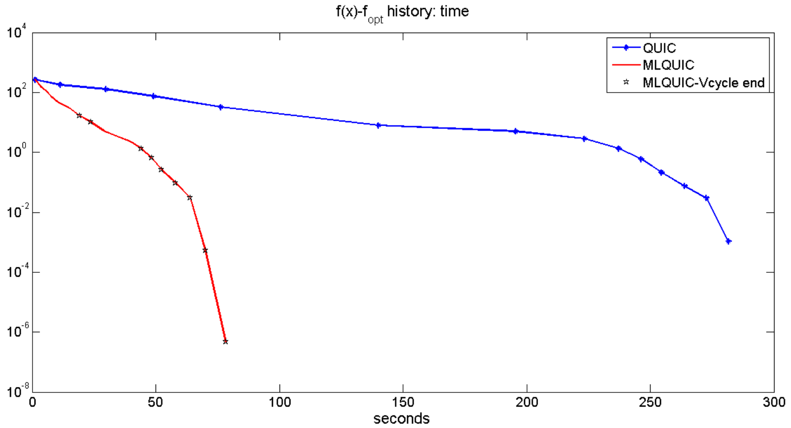


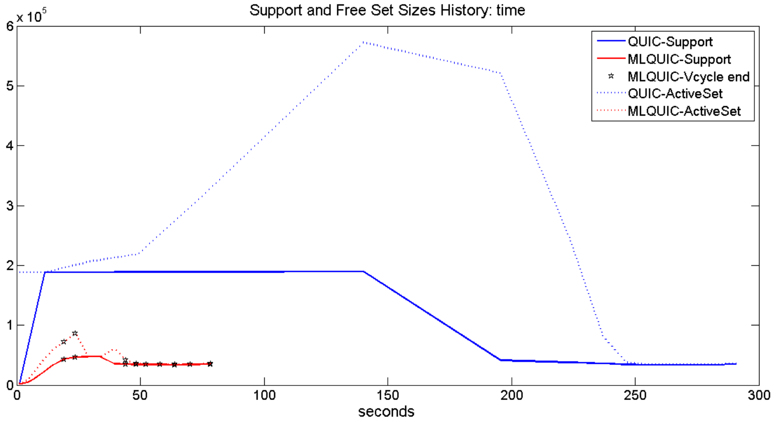
4.3. **Lymphatic System.** $S \in \mathbb{R}^{587\times 587}$, number of samples: 148, $\lambda = 0.2$





4.4. **Hereditary cancer.** $S \in \mathbb{R}^{1869\times 1869}$, $\lambda = 0.5$





4.5. **Results.** In all the graphs of convergence to F_{opt}, MLQUIC converges about 2 times faster than QUIC. In addition, MLQUIC does it with much less variables than QUIC throughout execution, while in the end both methods agree on the amount of non-zeros in Σ^{-1}.

5. Conclusion

The multilevel approach is very promising. Gradually builds the ActiveSet/support, resulting much smaller active set throughout the iterations. Applicable also for other types of sparse minimizations problems.

References

[1] A. Brandt and D. Ron, *Multigrid solvers and multilevel optimization strategies*, in Multilevel Optimization and VLSICAD, Kluwer (Boston), 2003, pp. 1–69.

[2] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A multigrid tutorial*, SIAM, Philadelphia, second ed., 2000.

[3] C.-J. Hsieh, M. A. Sustik, I. S. Dhillon, and P. D. Ravikumar, *Sparse inverse covariance matrix estimation using quadratic approximation.*, in Advances in Neural Information Processing Systems 24, 2011, pp. 2330–2338.

[4] E. Treister and I. Yavneh, *Square and stretch multigrid for stochastic matrix eigenproblems*, Numerical Linear Algebra with Application, 17 (2010), pp. 229–251.

[5] E. Treister and I. Yavneh, *A multilevel iterated-shrinkage approach to l_1 penalized least-squares minimization*, Signal Processing, IEEE Transactions on, 60 (2012), pp. 6319–6329.