Iranian Journal of Operations Research
Vol. 12, No. 1, 2021, pp. 1-19

Solving single facility goal Weber location problem using stochastic optimization methods

A. Soleimani Kourandeh[1], J. Fathali [2,*], S. Taherifard [3]

Location theory is one of the most important topics in optimization and operations research. In location problems, the goal is to find the location of one or more facilities in a way such that some criteria such as transportation costs, customer traveling distance, total service time, and cost of servicing are optimized. In this paper, we investigate the goal Weber location problem in which the location of a number of demand points on a plane is given, and the ideal is locating the facility in the distance R_i, from the i-th demand point. However, in most instances, the solution of this problem does not exist. Therefore, the minimizing sum of errors is considered. The goal Weber location problem with the ℓ_p norm is solved using the stochastic version of the LBFGS method, which is a second-order limited memory method for minimizing large-scale problems. According to the obtained numerical results, this algorithm achieves a lower optimal value in less time with comparing to other common and popular stochastic optimization algorithms. Note that although the investigated problem is not strongly convex, the numerical results show that the SLBFGS algorithm performs very well even for this type of problem.

***Keywords**: Goal Weber location problem, Quasi Newton algorithms, LBFGS methods, Stochastic optimization methods.*

Manuscript was received on 07/17/2021, revised on 09/14/2021 and accepted for publication on 11/26/2021.

1. Introduction

Today, optimization is widely used in the various fields. One of the practical areas of optimization is location problem. The first and most basic steps in planning to provide a service or products for applicants is to determine the best place to provide the service or products. Depending on the situation, a variety of location models have been proposed by researchers in this regard. Location research has been very extensive and has had a wide range of practical applications in various fields. Since the classic Weber problem [٤٣] was formulated in 1909 to determine the location of a warehouse, location theory has been an active part of research for the last decades.

The location problem varies in objective functions, distances, number and size of facilities to be established, and several other factors. Hongzhong et al. [١٥] introduced eight factors that are effective in classifying facility location models. These eight factors are: geographical characteristics, facility characteristics, objectives, solution method, demand patterns, supply chain types, time horizon and input parameters. Therefore, depending on the type of goal and using each

* Corresponding Author.
[1] Faculty of Mathematical Sciences, Shahrood University of Technology, Shahrood, Iran,
Email: aria.soleimani@gmail.com
[2] Faculty of Mathematical Sciences, Shahrood University of Technology, Shahrood, Iran,
Email: fathali@shahroodut.ac.ir
[3] Department of Mathematical Sciences, Sharif University of Technology, Tehran, Iran,
Email: sara_taherifard@alum.sharif.edu

of these different indicators, we will achieve different models of location problem. Continuous location problems are a specific type of location problems where we seek to locate one or more facilities on the plane. One of the most well-known and important continuous location problems is the Fermat-Weber single facility location problem. In this case, there are some given points on the plane and the goal is to find a new point on the plane so that the sum of total distances of the demand points to the new point is minimized [۲۱].

In 1937, Weiszfield [٤٤] proposed an iterative method to the Fermat-Weber problem. Then, in 1958, Mihl [۲۳] developed the Weiszfield method for multi-facilities location problems with Euclidean norm. In 1964, Francis [۱۲] investigated the problem of multi-facilities with rectangular norm using the Wieszfield method. In 1971, Wesolowsky and Love [٤٥] considered the location problem on a plane with a rectangular distance between the demand points and rectangular areas. Maurice and Verdini [۲٦] in 1973, and Morris [۲۷] in 1981, discussed the Wieszfield algorithm for location problems with ℓ_p norm. Also in 2010, Iyigun and Ben Israel [۱٦], used the Weiszfield algorithm for allocation problems. Fathali [۸] investigated the backup multi-facilities location problem on the plane in 2014 by presenting a Weiszfield-like algorithm. Tirkolaee et al. [41,42] in 2020, considered the green location-allocation-inventory problem in uncertainty system and rescue unit allocation problem, respectively.

In recent decades, many attempts have been made to create location models that take into account more characteristics of the real world. One of these characteristics that has emerged in recent optimization theories is the concept of "goal location". Hence, Fathali et al. [11], for the first time, raised a specific problem of Weber's goal location. In this case, they considered an ideal distance for each demand point and considered the location of the facility in such a way that its distance from the demand points is equal to the corresponding ideal distance. Since in reality there is seldom a place for the facility where the distance to the demand points is exactly equal to the ideal distance, they sought to minimize total weighted squares error in this model. They proposed the big square-small square geometric method to solve the problem with the Euclidean norm. Then Jamalian and Fathali [1۷] proposed a linear programming model for the problem with the aim of minimizing the total weighted absolute error. Fathali and Jamalian [۹] studied the problem of minimizing the sum of squares error and named it Goal Square Weber Location Problem (GSWLP). They used Particle Swarm Optimization (PSO) algorithm to solve the above problem with Euclidean norm. Recently, Soleimani et al. [۳۹, ٤۰] solved two models of the goal location problem under the symmetric and asymmetric loss functions using nonlinear optimization methods. The fuzzy version of goal location problem with asymmetric loss function was developed by Nazari et al. [۲۸].

In this paper, we consider a large-scale goal location problem with ℓ_p norm. Solving large-scale problems using deterministic optimization methods is unreasonable due to the slow convergence rate. Since calculating the gradient and inverse Hessian or the approximation of Hessian inverse costs a high computational cost, so stochastic optimization methods have a special roll in large-scale problems. One of the second-order stochastic optimization methods is the stochastic version of the LBFGS (Limited memory BFGS) method, which in practice gives good results for large-scale problems. Therefore, we use this method to solve the problem of goal location and present the numerical results obtained from it.

In what fallows of this paper, the goal Weber location problem is defined in Section 2. A brief review of stochastic optimization methods is given in Section 3. Solving the goal Weber location problem using stochastic optimization methods is described in Section 4. Finally, computational results are reported in Section 5.

2. Problem definition

Suppose $A_1,\dots,A_n$ are the location of demand points in the plane. Let for $i=1,\dots,n$, $A_i=(a_i,b_i)$ and we have an ideal distances R_i, and a positive weight w_i, corresponding to point A_i. The goal is to find the location of a new facility with coordinate $X=(x,y)$ on the plane so that the weighted distance between X and the demand point A_i is exactly equal to R_i. But since such point may not actually be found, we seek to estimate the location of X so that the distance from this point to the demand point A_i is as close as possible to R_i.

Selecting this estimated location will result in an error (loss) relative to the goal point (facility). In this case, we are looking to minimize the error caused by selecting this location on the plane. Therefore, using the appropriate loss function (error function) is of particular importance. Therefore, according to the problem, the appropriate loss function should be selected. Thus, the goal Weber location problem with ℓ_p norm is modeled as follows:

$$\min f(X)=\frac{1}{n}\sum_{i=1}^{n} w_i.E(d(X,A_i)-R_i), \qquad (1)$$

where $E(d(X,A_i)-R_i)$ is the loss function, and $d(X,A_i)$ is the distance between two points X and A_i with ℓ_p norm.

As mentioned in [11] the goal Weber location problem can be applied for finding the location of a company in the vicinities of some cities with respect to the establishing and transportation cost. This problem also has some applications in finding the location of desirable and undesirable facilities. In these cases, because of undesirability of the facilities, they shouldn't be closer than a specified distance to the facility centers. On the other hand, if the facilities be so far from the facility centers, cost of providing security, human forces, transportation installation, and other costs will increase.

In this paper, we consider the goal Weber location problem under the least squares loss function and ℓ_p norm as follows.

$$\min f(X)=\frac{1}{n}\sum_{i=1}^{n} w_i.(d(X,A_i)-R_i)^2+U(X), \qquad (2)$$

where

$$d(X,A_i)=\left(((x-a_i)^2+\varepsilon)^{\frac{p}{2}}+((y-b_i)^2+\varepsilon)^{\frac{p}{2}}\right)^{\frac{1}{p}}, \qquad (3)$$

and $U:\mathbb{R}^d\to\mathbb{R}$ is the regularizer term. Regularization is a technique used for tuning the function by adding an additional penalty term in the error function. The additional term controls the excessively fluctuating function such that the solutions don't take extreme values. In addition, regularization prevents overfitting. So we added the regularizer term to the loss function. We will assume that the regularizer is an ℓ_2 regularizer, i. e., $U(X)=\frac{\beta}{2}\parallel X\parallel^2$ where $\beta>0$ is regularizer parameter.

In large scale problems, n is a large number, so calculating the Hessian (or Hessian approximation) and gradient for these problems has a high computational cost that can be reduced by using stochastic optimization methods. Now, by assuming,

$$f_i=w_i.(d(X,A_i)-R_i)^2, \qquad (4)$$

for the subset $S\subseteq\{1,\dots,n\}$, the function f_S is defined as follows

$$f_S(X)=\frac{1}{|S|}\sum_{i\in S} f_i(X)+U(X). \qquad (5)$$

We first assume that the regularization term has been divided equally and included in f_i. That is, the problem (2) is assumed to be $\min f(X)=\frac{1}{n}\sum_{i=1}^{n} f_i(X)$.

3. Stochastic optimization techniques

In this section we will summarize some basic ideas for optimization techniques for large-scale problems. These techniques are described in two main ways: first-order methods and second-order methods.

3.1 First-order methods

The first-order methods are popular in optimization algorithms because these methods achieve the desired result at a time proportional to the problem dimension. In fact, first-order methods are gradient-based (i.e., first-order information of function) method.

Stochastic Gradient Descent (SGD): The gradient descent method is a first-order iterative method that attempts to reduce the value of a function in each iteration by moving in the opposite direction of the gradient as follow,

$$X_t = X_{t-1} - \frac{\eta}{n}\sum_{i=1}^{n} \nabla f_i(X_{t-1}).$$

This method converges linearly to the optimal point. Although, traditional gradient-based methods may be effective for solving small-scale (small n) optimization problems, we incur high computational cost for large-scale problems because we have to calculate n gradients in each iteration (batch gradient, or full gradient method), so we use a random type of gradient-based algorithms. This method was proposed by Robbins and Monro [٣٣] in 1951 as the stochastic gradient method. In this method, we select i in each iteration randomly from $\{1,2,\dots,n\}$ and update the problem parameter as follows

$$X_t = X_{t-1} - \eta \nabla f_i(X_{t-1}),$$

where η is called the step size. The advantage of this method is that in each iteration, only one gradient ∇f_i is calculated. Thus, its computational cost is $\frac{1}{n}$ of the computational cost of the standard gradient descend method. But using one gradient as the unbiased estimator for the true (full) gradient, causes reduction in the convergence rate. The SGD method also converges sublinearly even for strongly convex functions and we have

$$f(X_t) - f(X^*) = O(1/t).$$

A significant advancement in terms of the running time of first order methods was achieved recently by a clever merging of stochastic gradient descent with its full version to provide variance reduction.

3.1.1 Variance reduction methods

Recently, an important achievement has been obtained in the first order methods, which improves the running time of the algorithm by reducing the variance. These methods are linearly converged for strongly convex function that improve sublinear rate of SGD. This is achieved by increasing computation cost or increasing storage. These algorithms include Stochastic Average Gradient (SAG) [٣٤], [٦] and Stochastic Variance Reduced Gradient (SVRG) [١٨], [٣٨].

Stochastic variance reduced gradient (SVRG): This method operates in cycles. Each cycle begins with a batch (full) gradient at X_t, i. e. $\nabla f(X_t) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(X_t)$. Then, for the inner loop, we first set $\tilde{X}_1 = X_t$, and the internal iterations are updated by $\tilde{X}_{k+1} = \tilde{X}_k - \eta \tilde{g}_k$, where

$$\tilde{g}_k = \nabla f_{i_k}(\tilde{X}_k) - \left(\nabla f_{i_k}(X_t) - \nabla f(X_t)\right), \qquad (6)$$

and i_k is chosen randomly from $\{1,\dots,n\}$. We can interpret (6) as follows. It is obvious that the expected value of $\nabla f_{i_k}(X_t)$ is equal to $\nabla f(X_t)$ where random variable $i_k \in \{1,\dots,n\}$ is chosen randomly. Thus, $\nabla f_{i_k}(X_t) - \nabla f(X_t)$ is the bias in the gradient estimate $\nabla f_{i_k}(X_t)$. Therefore, the algorithm randomly chooses $i_k \in \{1,\dots,n\}$ and calculates the stochastic gradient $\nabla f_{i_k}(\tilde{X}_k)$ in every iteration, and corrects it according to the corresponding bias. Note that $\tilde{X}_k$ is the current point in inner loop. With these explanations we can say that $\tilde{g}_k$ is an unbiased estimator of $\nabla f(\tilde{X}_k)$. Note that if we chose $\tilde{g}_k = \nabla f_{i_k}(\tilde{X}_k)$ as SGD, it has a larger variance than this method. Therefore, one

iteration of SVRG is much more expensive than one of SGD, and in fact is comparable to a full gradient iteration. This method is given in Algorithm SVRG.

Algorithm SVRG [۳]
input: $X_1, f(X) = \frac{1}{n}\sum_{i=1}^{n} f_i(X), T, m$
For $t = 1$ to T **do**
1. Compute the full gradient $\nabla f(X_t)$.
2. Let $\tilde{X}_1 = X_t$
3. For $k = 1$ to m do
   3.1. Choose i_k randomly from $\{1, \ldots, n\}$.
   3.2. Set $\tilde{g}_k = \nabla f_{i_k}(\tilde{X}_k) - (\nabla f_{i_k}(X_t) - \nabla f(X_t))$.
   3.3. Set $\tilde{X}_{k+1} = \tilde{X}_k - \eta \tilde{g}_k$.
   3.4. End for.
4. Option (a): $X_{t+1} = \tilde{X}_{m+1}$.
5. Option (b): $X_{t+1} = \frac{1}{m}\sum_{k=1}^{m} \tilde{X}_{k+1}$.
6. Option (c): Choose k randomly from $\{1, \ldots, m\}$ and set $X_{t+1} = \tilde{X}_{k+1}$.
   End for.
   End of algorithm.

Stochastic Average Gradient (SAG): In this method, by reducing the estimator variance, we can achieve a linear convergence rate similar to the full gradient method while maintaining the computational cost of iteration such as the SGD method. This method combines the low cost of the SGD iteration and the linear convergence rate similar to the full gradient descend method. The form of iterations is

$$X_t = X_{t-1} - \frac{\eta}{n}\sum_{i=1}^{n} y_i^t.$$

In fact, the algorithm chooses i_t-th data randomly at each iteration and set

$$y_i^t = \begin{cases} f_i'(X_t) & \text{if } i = i_t, \\ y_i^{t-1} & o.w. \end{cases}$$

Thus, at each iteration, only one gradient with respect to i_t-th data are calculated, and instead of the other gradients, we take the same values of the previous iteration. It means, like the full gradient method, the step incorporates a gradient with respect to each data. However, the same as the SGD method, each iteration only computes the gradient with respect to a single data and the cost of the iterations is independent of n. Clearly, this method requires more storage because it must store the gradients of the previous iteration.

3.1.2 Other Popular Methods

There are other optimization techniques that have made some useful gains. In this subsection, we introduce these methods which include gradient methods with momentum, accelerated gradient methods, and coordinate descent methods.

Gradient Methods with Momentum: Each step of this method is chosen as a combination of the steepest descent direction and the most recent iterate displacement. Initially, we consider a point X_0, and the sequences of scalers $\{\alpha_t\}$ and $\{\beta_t\}$ which are predetermined or dynamical. Then the form of iteration is

$$X_1 = X_0, \quad X_{t+1} = X_t - \alpha_t \nabla f(X_t) + \beta_t (X_t - X_{t-1}). \quad (7)$$

The right-hand is called the momentum term, which preserves the movement of algorithm recursively along previous search directions. For large scale problems, a stochastic gradient is used instead of the full gradient in (7). Using the idea of Nesterov's momentum [٢٩], Katyusha algorithm [٢] introduced the concept of negative momentum which is a variance-reduction based method. In fact, Katyusha accelerate SVRG algorithm for the strongly convex problem by the mini-batch setting.

Accelerated Gradient Methods: The formula of this method is similar to (7) and it's idea of acceleration proposed by Nesterov [٣٠]. Each iteration has the form

$$\widetilde{w}_t = X_t + \beta_t(X_t - X_{t-1}),$$
$$X_{t+1} = \widetilde{w}_t - \alpha_t \nabla f(\widetilde{w}_t),$$

which one can easily obtain:

$$X_{t+1} = X_t - \alpha_t \nabla f(X_t + \beta_t(X_t - X_{t-1})) + \beta_t(X_t - X_{t-1}). \quad (8)$$

In (7), first one takes the steepest descent step and then applies the momentum term, whereas in (8), one follows the momentum term first, then applies a steepest descent step (with the gradient evaluated at $\widetilde{w}_k$, not at X_k). For large scale problems, a stochastic gradient is used instead of the full gradient in (8). This technique has been applied to a large class of algorithms. One of these algorithms is Catalyst [٢٠] which use this strategy to accelerate gradient method.

Coordinate Descent Methods: Another method, among the oldest in optimization method, is Coordinate descent (CD). These methods work by considering steps along coordinate directions: one tries minimizing the objective function with respect to a single variable while all others are kept fixed, then other variables are updated similarly in an iterative procedure. The form of iteration is

$$X_{t+1} = X_t - \alpha_t \nabla_{i_t} f(X_t) e_{i_t}, \quad where \quad \nabla_{i_t} f(X_t) := \frac{\partial f}{\partial w^{i_t}}(X_t). \quad (9)$$

Note that X_{i_t} indicates the i_t-th component of w, and e_{i_t} indicates the i_t-th coordinate vector for some $i_t \in \{1, \ldots, d\}$. For large-scale problems, a stochastic version of the CD method can be discussed, which is called Stochastic Dual Coordinate Ascent (SDCA) [٣٦]. For this purpose, it is not typical that one replaces $\nabla_{i_t} f_(X_t)$ in (9) with stochastic version, because we can usually calculate a d-dimensional stochastic gradient to apply an SG method. So, one can nevertheless apply stochastic setting for CD method by maximizing its dual problem and using positive gradient steps instead of negative one. (Thus, however, using the maximizing dual problem and using positive gradient steps rather than negatives, stochastic approximation can be applied to the CD method.) SDCA has linear convergent rate with constant dependent on the parameter dimension d. Furthermore, for Accelerated Stochastic Dual Coordinate Ascent method you can see [٣٧] and [٩].

3.2 Second-order methods

Second-order methods such as the Newton method are popular because of their quadratic convergence rate, but computing the full Hessian matrix and its inversion is costly ($O(nd^2)$ and $O(d^3)$, respectively). Recently, NewSamp's algorithm [7] proposed ideas for solving these two problems. It uses the sub-sampling technique to approximate Hessian and uses low-rank projection techniques to reduce the cost of inverting Hessian.

3.2.1 Hessian-Free Newton Methods

As we know, to get the direction of Newton method, s_t, we need to solve equation $H_t s_t = -g_t$. Solving this equation requires computation and storage that will cost us. Thus, instead of solving the Newton equation exactly, inexact methods such as the Newton Conjugate Gradient (NCG) method can be used to solve this equation and to obtain a superlinear convergence rate.

In the NCG method, we do not need to access to the Hessian matrix itself precisely, but rather the Hessian-vector production is sufficient, which is why Hessian-free is mentioned. one can see [۲۲] and [4] for further studies.

Also in 2017, LiSSA (Linear (time) Stochastic Second-Order Algorithm) [۱] used a special technique for calculating the Hessian inverse, and obtained an unbiased estimator for the Hessian inverse, and achieved linear convergence rate.

3.2.2 Quasi-Newton methods

As mentioned earlier, the Hessian matrix computation is costly, other methods used to reduce this computational cost are called quasi-Newton methods stemming from the BFGS algorithm (Broyden, 1970; Fletcher, 1970; Goldfarb, 1970; Shanno, 1970) and its limited memory variant (LBFGS) [۳۱]. In this method, the approximations to Hessian are calculated using gradient changes. In fact, it has been proven that the limited memory version is effective for problems with millions of variables.

For large-scale optimization problems, stochastic techniques are used and stochastic Newton methods are proposed in different forms. In 2007, for the first time, a simple stochastic model was introduced for the BFGS algorithm by sampling technique for computing gradient differences [۳۵]. In addition, Mokhtari [۲٤] has provided a regularized stochastic BFGS algorithm. Byrd et al. [۵] also computed estimators for average curvature information using the subsampled Hessian-vector products technique. But these algorithms all have sub-linear convergence rates. Finally, in 2016, an important achievement was provided by Moritz et al. [۲۵] who achieved linear convergence rates with ideas of variance reduction (similar to the SVRG method) and mini-batch technique.

4. Stochastic version of LBFGS algorithm

In this section, we use a second-order stochastic method to minimize the goal location problem error. This method gives acceptable solutions in less time in cases where the set of demand points has a large size. For this purpose, we introduce a stochastic version of the LBFGS method and minimize the location problem error using this method.

Solving large-scale problems using deterministic optimization methods is unreasonable due to the slow convergence rate, because calculating the gradient and inverse Hessian or the approximation of Hessian inverse has a high computational cost, so stochastic optimization methods have a special roll in large-scale problems. One of the second-order stochastic optimization methods is the stochastic version of the LBFGS method, which in practice gives good results for large-scale problems.

Although most stochastic algorithms have high processing and execution speeds, the variance obtained from gradient estimator reduces the convergence rate near the optimal solution. For example, in the SGD method, even if we initialized at the optimum, we may achieve the objective function with a worse value. For this reason, we need diminishing step sizes to guaranty convergence. One way to solve this problem is to use methods that reduce the variance of the gradient estimator and increase the speed of the algorithm.

We now describe the stochastic version of the LBFGS method and use a technique similar to the SVRG method to reduce the variance of the gradient estimator. This algorithm performs well in large-scale problems and has a high convergence rate for strongly convex functions due to the use of the idea of variance reduction technique.

4.1 SLBFGS algorithm

In this section, we describe the stochastic version of the LBFGS algorithm and explain its important points.

Our updates will use stochastic estimates of the gradient ∇f_S as well as stochastic approximations to the inverse Hessian $\nabla^2 f_T$. In order to decouple the estimation of the gradient from the estimation of the Hessian, we use distinct subsets $S, T \subseteq \{1, \ldots, n\}$ and let $|S| = b$, $|T| = b_H$.

Similar to the SVRG method, in order to reduce the variance of the gradient estimator, the full gradient is calculated once in each outer loop and the iterations are updated according to the following rule

$$X_{k+1} = X_k - \eta_k H_k v_k,$$

where H_k represents the approximate Hessian inverse and v_k represents the stochastic gradient estimator with the reduced variance in the k-th iteration.

The iterations of this algorithm are shown in Algorithm 4.1. This algorithm has several parameters, which we will describe in the following. The parameter η indicates the step size of the algorithm. The positive integer m, indicates the number of iterations of the inner loop, represents the number of calculations of the full gradient. As mentioned earlier, using the full gradient reduces the variance of the stochastic gradient estimator. In addition, every L iterations, the approximation for inverse Hessian is updated. The vector s_r stores the average directions obtained during the $2L$ recent iteration of the algorithm. The vector y_r is obtained by multiplying the vector s_r in the estimator of Hessian matrix. Note that one of the differences between the SLBFGS and LBFGS algorithms is how the y_r vector is defined. In the LBFGS algorithm, the vector y_r is defined as the difference of the gradients, but in the stochastic version, the mentioned definition performs better. Now we use the vectors y_r and s_r to calculate the approximations for the inverse Hessian. For this purpose, we put $\rho_j = 1/s_j^T y_j$, $M' = \min\{M, r\}$ and recursively, define approximations for the inverse of Hessian as follows

$$H_r^{(j)} = (I - \rho_j s_j y_j^T)^T H_r^{(j-1)} (I - \rho_j s_j y_j^T) + \rho_j s_j s_j^T, \quad j \in \{r - M' + 1, \ldots, r\}. \qquad (10)$$

Initial with $H_r^{(r-M')} = \frac{s_r^T y_r}{\|y_r\|^2}$ and set $H_r = H_r^{(r)}$. Note that the above rule preserves positive definiteness ($\rho_j > 0$). Before describing the convergence theorem, we define the λ-strongly convex and Λ-smooth function.

Definition 1 The convex function $f: \mathbb{R}^d \to \mathbb{R}$ is λ-strongly convex and Λ-smooth when

$$\forall X, Z, \quad \nabla f(X)^T (Z - X) + \frac{\Lambda}{2} \| Z - X \|^2 \geq f(Z) - f(X) \geq \nabla f(X)^T (Z - X) + \frac{\lambda}{2} \| Z - X \|^2.$$

Algorithm SLBFGS [۲۰]
Input: X_0, m, L, η
Let $r = 0$ and $H_0 = I$
For $k = 0, \ldots$
1. Compute the full gradient $\mu_k = \nabla f(X_k)$.
2. Let $x_0 = X_k$
3. For $t = 0$ to $m - 1$
   - 3.1 Choose randomly $S_{k,t} \subseteq \{1, \ldots, n\}$
   - 3.2 Compute the stochastic gradient $\nabla f_{S_{k,t}}(x_t)$
   - 3.3 Set $v_t = \nabla f_{S_{k,t}}(x_t) - \nabla f_{S_{k,t}}(X_k) + \mu_k$
   - 3.4 Let $x_{t+1} = x_t - \eta H_r v_t$
   - 3.5 If $t \equiv 0 \bmod L$ then
     - 3.5.1 Increment $r = r + 1$

3.5.2 Set $u_r = \frac{1}{L}\sum_{j=t-L}^{t-1} x_j$

3.5.3 Choose randomly $T_r \subseteq \{1,\dots,n\}$ to define the stochastic approximation $\nabla^2 f_{T_r}(u_r)$

3.5.4 Compute $s_r = u_r - u_{r-1}$

3.5.5 Compute $y_r = \nabla^2 f_{T_r}(u_r)s_r$

3.5.6 Define H_r as (10).

End if

End loop

4. Let $X_{k+1} = x_i$ where $i \in \{0,\dots,m-1\}$ is randomly choosed.

End loop

Lemma 1 ([۳۹]) The objective function of (2) is nonconvex.

Lemma 2 ([۲۰]) Suppose that f is a function λ-strongly convex and Λ-smooth, and every f_i is convex and twice continuously differentiable for each $i \in \{1,\dots,n\}$. Then, the constants $0 < \gamma \leq \Gamma$ exist such that

$$\gamma I \leq H_r \leq \Gamma I \qquad \forall r \geq 1.$$

In [۲۰], it is proved that γ and Γ have the following values,

$$\gamma = \frac{1}{(d+M)\Lambda}, \qquad \Gamma = \frac{((d+M)\Lambda)^{d+M-1}}{\lambda^{d+M}}.$$

Lemma 3 ([۲۰]) Let X^* be the unique minimizer of f. Let $\mu_k = \nabla f(X_k)$ and $v_t = \nabla f_S(x_t) - \nabla f_S(X_k) + \mu_k$ be the variance-reduced stochastic gradient, then

$$\mathbb{E}[\| v_t \|^2] \leq 4\Lambda(f(x_t) - f(X^*) + f(X_k) - f(X^*)).$$

Theorem 1 ([۲۰]) Suppose that f is a function λ-strongly convex and Λ-smooth, and every f_i is convex and twice continuously differentiable for each $i \in \{1,\dots,n\}$. Let X^* be the unique minimizer of f. Then for all $k \geq 0$,

$$\mathbb{E}[f(X_k) - f(X^*)] \leq \alpha^k \mathbb{E}[f(X_0) - f(X^*)]$$

where the convergence rate α is given by

$$\alpha = \frac{1/(2m\eta) + \eta\Gamma^2\Lambda^2}{\gamma\lambda - \eta\Gamma^2\Lambda^2} < 1.$$

Also, assuming that $\eta < \gamma\lambda/(2\Gamma^2\Lambda^2)$ and m is selected large enough to satisfy

$$\gamma\lambda > \frac{1}{2m\eta} + 2\eta\Gamma^2\Lambda^2.$$

This method achieves desirable convergence properties with the help of stochastic techniques for large scale problems. In addition, due to the use of variance reduction techniques, it has a high speed and achieves a lower optimal value in less time than other popular methods of stochastic optimization. Another advantage of this method is its limited memory feature, which leads to limited memory usage. The convergence of this method has been proven for strongly convex functions, but according to numerical results, it has a good performance even for the nonconvex function, which has not yet been proved for nonconvex functions, and this is an issue for further studies of this method.

5. Computational results

In this section, the proposed algorithms were tested on some test problems with 30 to 1000 points that are generated randomly in MATLAB. In these problems the demand point's coordinates, weights and radiuses, all are positive numbers and generated randomly in the intervals [1,100], [1,3] and [1,10], respectively. The numerical results are obtained by solving the goal location problem (2) with ℓ_p norm using SLBFGS algorithm. The results of this method are compared with those obtained by SGD and SVRG methods. All algorithms are implemented using Matlab-R2017b and a laptop with Intel core i7-4510U processor and 8 GB RAM.

For all three algorithms, the regularizer parameter is assumed to be 0.1. For the SVRG and SLBFGS methods a fixed step size of 0.0001 is used and for the SGD method a diminishing step size of $1/\| \nabla f(X_k) \|$ is used.

Note that although the problem of goal location with the convex ℓ_p norm is not strongly convex, the numerical results presented in this section show that the SLBFGS algorithm performs very well even for this type of problem.

Figures 1, 3 and 5 compare the values of the objective function of the goal location problem with the ℓ_2 norm for the problems with 30, 100 and 1000 demand points, respectively, using the SVRG, SGD, and SLBFGS algorithms versus the iterations. As one can see in these figures, the SGD algorithm has obtained the values of the objective function in an oscillating manner and has not performed well, but the SLBFGS and SVRG algorithms have performed well and have reduced the values of the objective function very well. In addition, SLBFGS achieves a lower loss function value than SVRG in fewer iterations.

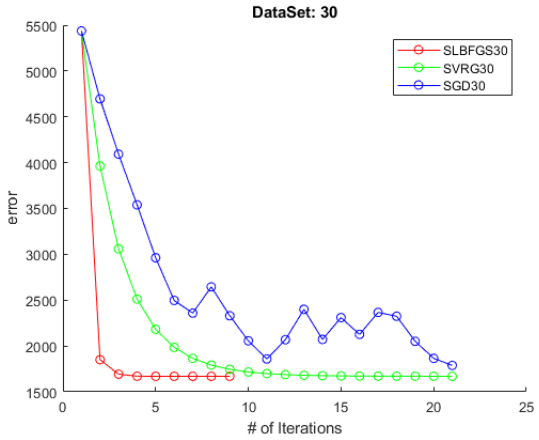

Figure 1: Comparison of the values of the objective function of the goal location problem using SVRG, SGD and SLBFGS algorithms versus the iterations for a problem with 30 demand points.

Figures 2, 4 and 6 show the variance values of the gradient estimator for the SLBFGS and SVRG methods versus the iterations for a problem with 30, 100 and 1000 demand points, respectively. According to these figures, for the instances with 30 and 100 points, the variance of the gradient estimator decreases at a high rate in both methods, which indicates the excellent performance of both methods in reducing the variance.

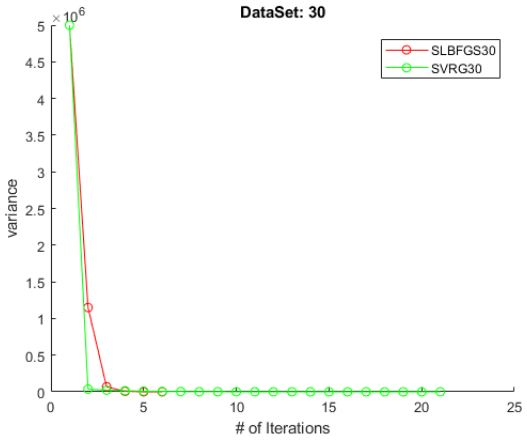

Figure 2: Comparison of variance values of gradient estimator of goal location problem using SVRG and SLBFGS algorithms versus the iterations for a problem with 30 demand points.

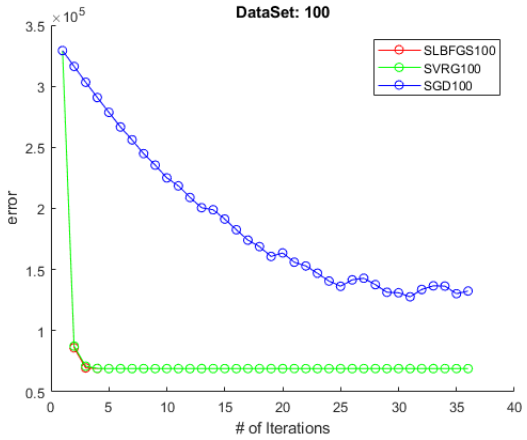

Figure 3: Comparison of the values of the objective function of the goal location problem using SVRG, SGD and SLBFGS algorithms versus the iterations for a problem with 100 demand points.

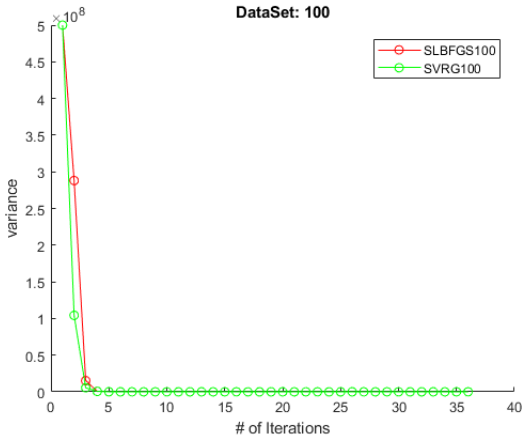

Figure 4: Comparison of variance values of gradient estimator of goal location problem using SVRG and SLBFGS algorithms versus the iterations for a problem with 100 demand points.

For the instance with 1000 points, the reduction of variance can be seen well, with the difference that in the fourth iteration, due to the randomness of the method and undesirable sampling, the variance of the gradient estimator increased but continued to decrease again. The variance obtained from the SLBFGS method has a very significant decrease in fewer iterations, which has led to a decrease in the value of the objective function at a high rate, which confirms the results in Figure 5.

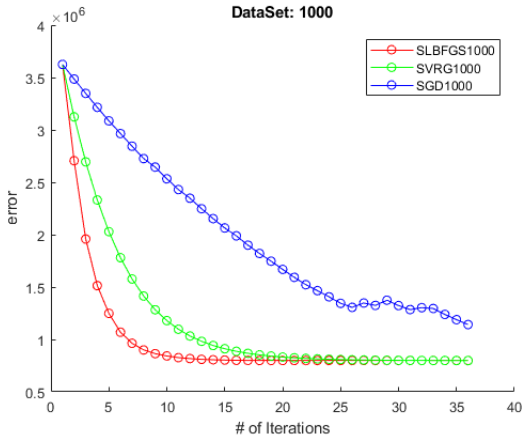

Figure 5: Comparison of the values of the objective function of the goal location problem using SVRG, SGD and SLBFGS algorithms versus the iterations for a problem with 1000 demand points.

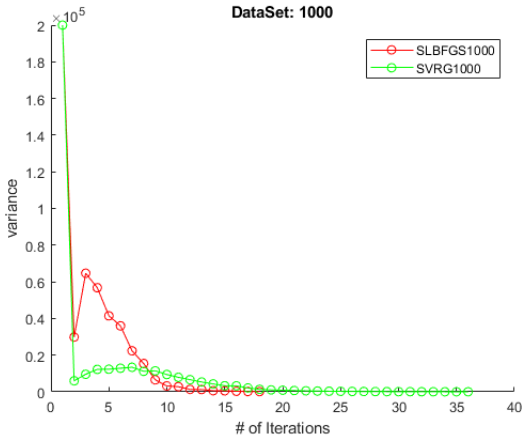

Figure 6: Comparison of variance values of gradient estimator of goal location problem using SVRG and SLBFGS algorithms versus the iterations for a problem with 1000 demand points.

Table 1 shows the optimal values, optimal solutions and running time of SVRG, SGD and SLBFGS algorithms for different values of p.

Table 1: Optimal values, optimal solutions and running time of SVRG, SGD and SLBFGS algorithms for different values of p.

p	n	Algorithm	optimal value	optimal solution	running time
	3	SGD	1854.9	(7.2772, 6.1040)	0.0307
	0	SVRG	1669.5	(8.2336, 7.6501)	0.0700
		SLBFGS	1669.2	(8.2771, 7.6902)	0.7150

	100	SGD	1.3188×10^5	(15.7438,16.1686)	0.0467
		SVRG	0.6900×10^5	(30.4584,29.2005)	0.3793
		SLBFGS	0.6900×10^5	(30.4642,29.2077)	0.2877
	1000	SGD	1.1183×10^6	(23.2480,19.1347)	0.2576
		SVRG	0.7983×10^6	(30.2357,30.3713)	2.4657
		SLBFGS	0.7992×10^6	(29.9692,30.0746)	1.4265
.5	30	SGD	2935.0	(6.9970, 3.8888)	0.1591
		SVRG	2035.3	(8.2336, 7.6501)	0.1149
		SLBFGS	2035.9	(8.2772, 7.6902)	1.1868
.5	100	SGD	1.5111×10^5	(19.1842, 14.04128)	0.1244
		SVRG	0.8457×10^5	(29.8929, 29.3884)	1.3766
		SLBFGS	0.8458×10^5	(30.0533, 29.5221)	1.6145
.5	1000	SGD	1.3037×10^6	(20.6655, 22.4782)	1.0118
		SVRG	0.9837×10^6	(30.1970, 30.55781)	6.4371
		SLBFGS	0.9835×10^6	(30.3239, 30.7762)	3.4832

In optimization methods, random sample size plays an important role in reducing the variance of the estimator and the performance of the algorithm. Therefore, we have implemented SVRG, SGD and SLBFGS algorithms with different random sample sizes. We represent the random sample size, which is the number of random Hessian and gradients to calculate the search direction, with $m = |S|$.

Figures 7 to 12 show the results obtained by the algorithms for different values of m. These results indicate that increasing the random sample size reduces the variance of the gradient estimator and improves the performance of the algorithms.

Table 2 presents the optimal values, the optimal solutions, the running time and the variance of the gradient estimator of SVRG, SGD and SLBFGS algorithms for different values of m.

Table 2: The optimal values, the optimal solutions, the running time and the variance of the gradient estimator of SVRG, SGD and SLBFGS algorithms for different values of m.

m	n	Algorithm	optimal values	optimal solutions	running time	variance
	30	SGD	3.1533×10^3	(4.4316,4.0064)	0.0095	-
		SVRG	1.6709×10^3	(8.2300,7.5365)	0.1720	0.5765
		SLBFGS	1.6703×10^3	(8.2612,7.5637)	0.4108	5.8282×10^3
	100	SGD	1.2717×10^5	(18.2118,14.7525)	0.0522	-
		SVRG	0.6900×10^5	(30.4584,29.2005)	0.3072	5.6211×10^{-22}
		SLBFGS	0.6900×10^5	(30.4642,29.2077)	0.0661	2.8786
	1000	SGD	1.3202×10^6	(20.2169,16.5848)	0.2543	-
		SVRG	0.7983×10^6	(30.2357,30.3713)	1.5258	24.1414
		SLBFGS	0.7984×10^6	(30.1902,30.3190)	1.4853	100.4597

0	30	SGD	1.8520×10^3	(6.8308, 6.5223)	0.0066	-
		SVRG	1.6709×10^3	(8.2298, 7.5364)	0.1637	0.5664
		SLBFGS	1.6707×10^3	(8.2341, 7.5429)	0.3625	6.4919
0	100	SGD	0.8835×10^5	(24.2827,20.1608)	0.0599	-
		SVRG	0.6900×10^5	(30.4584,29.2005)	0.2345	5.0009×10^{-22}
		SLBFGS	0.6900×10^5	(30.4584,29.2005)	0.0473	0.003145
2	1000	SGD	0.9340×10^6	(23.9485,24.6625)	0.2624	-
		SVRG	0.7983×10^6	(30.2325,30.3679)	1.4199	22.9194
		SLBFGS	0.7901×10^6	(30.2402,30.3200)	15.4827	94.7234

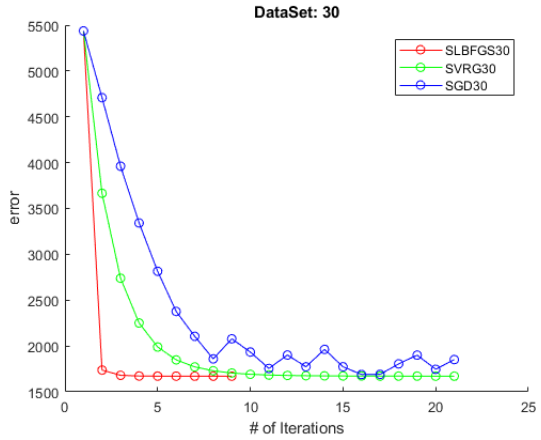

Figure 7: Comparison of the values of the objective function of the goal location problem using SVRG, SGD and SLBFGS algorithms versus the iterations for a problem with 30 demand points and $m = 10$.

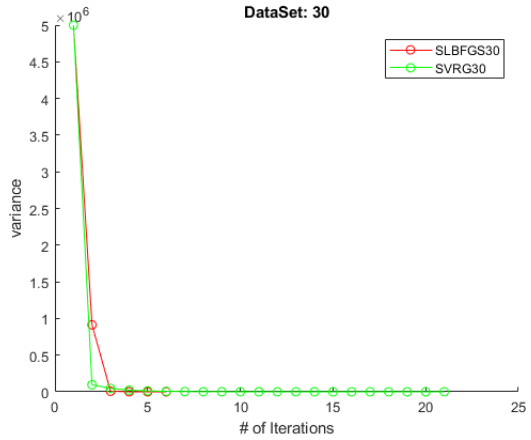

Figure 8: Comparison of variance values of gradient estimator of goal location problem using SVRG and SLBFGS algorithms versus the iterations for a problem with 30 demand points and $m = 10$.

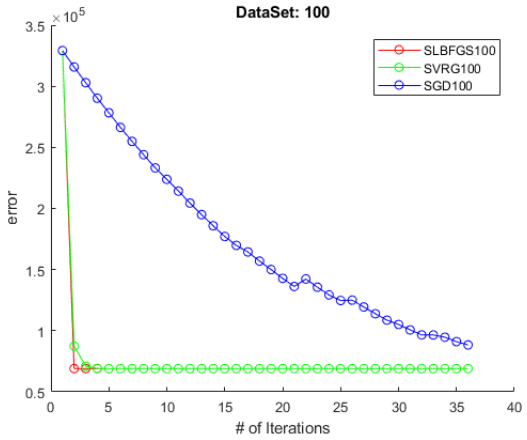

Figure 9: Comparison of the values of the objective function of the goal location problem using SVRG, SGD and SLBFGS algorithms versus the iterations for a problem with 100 demand points and $m = 10$.

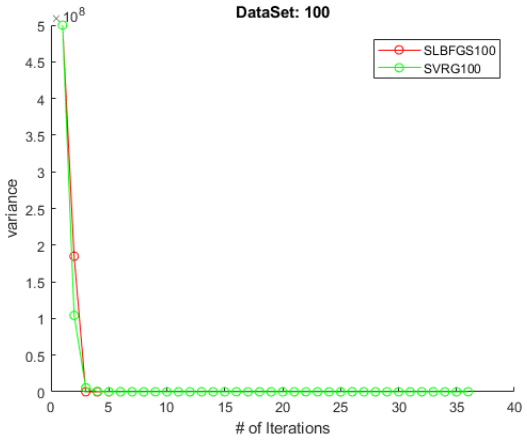

Figure 10: Comparison of variance values of gradient estimator of goal location problem using SVRG and SLBFGS algorithms versus the iterations for a problem with 100 demand points and $m = 10$.

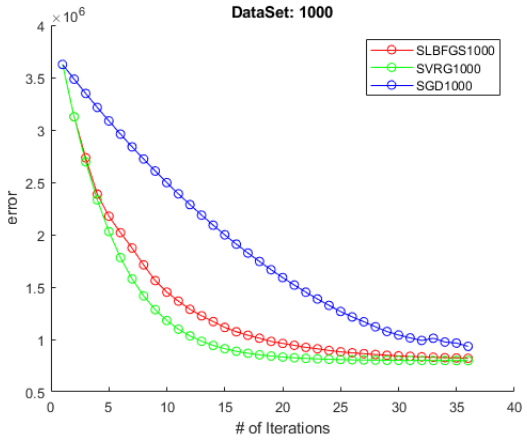

Figure 11: Comparison of the values of the objective function of the goal location problem using SVRG, SGD and SLBFGS algorithms versus the iterations for a problem with 1000 demand points and $m = 32$.

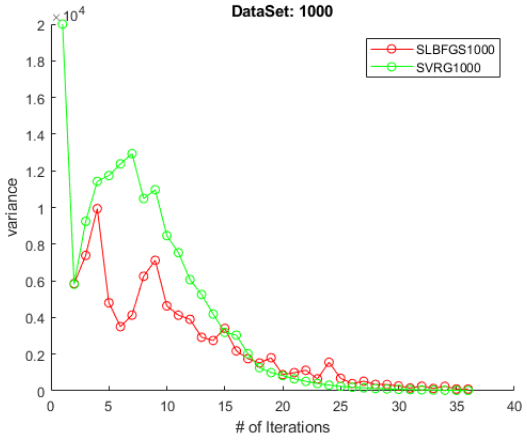

Figure 12: Comparison of variance values of gradient estimator of goal location problem using SVRG and SLBFGS algorithms versus the iterations for a problem with 1000 demand points and $m = 32$.

6. Summary and suggestions for the future works

In this paper, we solve large-scale goal location problems using stochastic optimization methods and present the numerical results. The results of the implementation of various nonlinear optimization methods show that the stochastic version of LBFGS methods have better convergence properties than the other popular methods. Stochastic version of LBFGS methods also performed well for large-scale problems and achieved fewer objective functions at lower execution times. The results indicate the outperforming this method with SGD and SVRG methods, specially for large instances. Note that although the investigated problem is not strongly convex, the numerical results show that the SLBFGS algorithm performs very well even for this type of problem that is an important and special achievement.

As the suggestion for the future works, we can solve the goal multivariate location problems using nonlinear optimization methods. Also, for large-scale problems, a stochastic approach can be considered and a variety of stochastic techniques can be used. In addition, different loss functions with

other properties can be considered and minimized. In addition, the field of stochastic optimization has a wide range of techniques, one of which is called the dynamic sample size technique. In this technique, the random sample size is dynamically changed using the variance of estimator in each iteration to help further reduce the variance and increase the speed of the algorithm. Another goal that we will address in future studies is to solve constrained goal location problems that are considered constraints on the problem. In addition, the goal location model can be considered on the plane, in which divided with a line to half spaces with varying norms (see e.g. [10]).

References

[1] Agarwal N., Bullins B., and Hazan E., (2017), Second-order stochastic optimization for machine learning in linear time, J. Mach. Learn. Res., 18, pp 4148–4187.
[2] Allen-Zhu Z., (2018) Katyusha: The first direct acceleration of stochastic gradient methods. Journal of Machine Learning Research 18, pp 1-51.
[3] Bottou L., Curtis F. E., Nocedal J., (2018), Optimization methods for large-scale machine learning, SIAM Rev., 60, pp 223-311.
[4] Byrd R. H., Chin G. M., Neveitt W., and Nocedal J., (2011), On the use of stochastic Hessian information in optimization methods for machine learning. SIAM Journal on Optimization, 21(3), pp 977-995.
[5] Byrd R. H., Hansen S. L., Nocedal J., Singer Y., (2016), A stochastic quasi-Newton method for large-scale optimization, SIAM Journal on Optimization, 26(2), pp 1008-1031.
[6] Defazio A., Bach F. R., Lacoste-Julien S., (2014) Saga: A fast incremental gradient method with support for non-strongly convex composite objectives, In Advances in Neural Information Processing Systems, pp 1646-1654.
[7] Erdogdu M. A., Montanari A., (2015), Convergence rates of sub-sampled Newton methods, In Advances in Neural Information Processing Systems, pp 3034-3042.
[8] Fathali J., (2014), Backup multifacility location problem with Lp-norm, OPSEARCH, 52, pp 382-391.
[9] Fathali J., Jamalian A., (2017), Efficient methods for goal square Weber location problem, Iranian Journal of Numerical Analysis and Optimization, 7, pp 65-82.
[10] Fathali J., Zaferanieh M., (2009), Location problems in regions with Lp and block norms, Iranian Journal of Operations Research, 2 (2), pp 72-87.
[11] Fathali J., Zaferanieh M., Nezakati A., (2009), A BSSS algorithm for the location problem with minimum square error, Advances in Operations Research, pages 1-10.
[12] Francis R. L., (1964), On the location of multiple new facilities with respect to existing facilities, The Journal of Industrial Engineering, 15, pp 10-107.
[13] Gorbunov E., Hanzely F., Richtarik P. A., (2020), Unified Theory of SGD: Variance Reduction, Sampling, Quantization and Coordinate Descent, Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, PMLR 108, pp 680-690.
[14] Gower R. M., Loizou N., Qian X., Sailanbayev A., Shulgin E., Richtárik P., (2019), SGD: General analysis and improved rates, arXiv preprint arXiv:1901.09401.
[15] Honggzhong, J., Fernando, O., Maged, D., (2007), "A modeling framework for facility location of medical services for large-scale emergencies", IIE Transactions, 39, pp 41-55, DOI: 10.1080/07408170500539113.
[16] Iyigun C., Ben-Israel A., (2010), A generalized Weiszfeld method for the multifacility location problem, Oper. Res. Lett., 38, pp 207–214.
[17] Jamalian A., Fathali J., (2009), Linear programming for the location problem with minimum absolute error, World Applied Sciences Journal, 7, pp 1423-1427.
[18] Johnson R., Zhang T., (2013), Accelerating stochastic gradient descent using predictive variance reduction, In Advances in Neural Information Processing Systems, pp 315-323.

[19] Lin Q., Lu Z., and Lin X., (2014), An accelerated proximal coordinate gradient method. In Advances in Neural Information Processing Systems, pp 3059-3067.
[20] Lin H., Mairal J., and Harchaoui Z., (2015), A universal catalyst for first-order optimization. In Advances in Neural Information Processing Systems, pp 3384-3392.
[21] Love, R.F., Morris, J.G., Wesolowsky, G.O., (1988), Facilities Location: Models and Methods, North-Holland, New York.
[22] Martens H., (2010), Deep learning via Hessian-free optimization. In International Conference on Machine Learning, pp 735-742.
[23] Miehle W., (1958), Link-length minimization in networks, Oper. Res., 6, pp 232–243.
[24] Mokhtari A., Ribeiro A., (2014), RES: regularized stochastic BFGS algorithm, IEEE Transactions on Signal Processing, 62(23), pp 6089-6104.
[25] Moritz P., Nishihara R., Jordan M., (2016), A linearly-convergent stochastic L-BFGS algorithm, In Artificial Intelligence and Statistics, pp 249-258.
[26] Morris J.G., (1981), Convergence of the Weiszfeld algorithm for Weber problems using a generalized distance function, Oper. Res., 29, pp 37–48.
[27] Morris J.G., Verdini W.A., (1979), A simple iterative scheme for solving minisum facility location problems involving lp distances, Oper. Res., 27, pp 1180–1188.
[28] Nazari M., Fathali J., Taghi-Nezhad N.A., (2020), Fuzzy single facility goal location problems with asymmetric penalty function, Modern Researches in Decision Making, 5, pp 30-58.
[29] Nesterov Y., (1983), A method of solving a convex programming problem with convergence rate O(1/k2). In Doklady AN SSSR (translated as Soviet Mathematics Doklady), volume 269, pp 543-547.
[30] Nesterov Y., (2004), Introductory Lectures on Convex Optimization: A Basic Course. Springer.
[31] Nocedal J., Wright S., (2006), Numerical Optimization, Springer Science \& Business Media.
[32] Poon C., Ltang J., Schoenlieb C., (2018), Local convergence properties of SAGA/Prox-SVRG and acceleration, Proceedings of the 35th Int. Conf. on Mach. Learn., 80, 4124–4132.
[33] Robbins H., Monro S., (1951), A stochastic approximation method, The Annals of Mathematical Statistics, pp 400-407.
[34] Roux N. L., Schmidt M., Bach F. R., (2012), A stochastic gradient method with an exponential convergence rate for finite training sets, In Advances in Neural Information Processing Systems, pp 2663-2671.
[35] Schraudolph N. N., Yu J., G¨unter S., (2007), A stochastic quasi-Newton method for online convex optimization, In Artificial Intelligence and Statistics, pp 436-443.
[36] Shalev-Shwartz S., Zhang T., (2013), Stochastic dual coordinate ascent methods for regularized loss minimization. Journal of Machine Learning Research, 14(Feb), pp 567-599.
[37] Shalev-Shwartz S., Zhang T., (2016), Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. Mathematical Programming, 155(1-2), pp 105-145.
[38] Shamir O., Srebro N., Zhang T., (2014), Communication-efficient distributed optimization using an approximate Newton-type method, In International Conference on Machine Learning, pp 1000-1008.
[39] Soleimani A., Fathali J., Nazari M., (2019), Single facility goal location problems with Lp norm, Modern Researches in Decision Making, 3, pp 125–152.
[40] Soleimani A., Fathali J., Nezakati A., Nazari M., (2019), Single facility goal location problems with symmetric and asymmetric penalty functions, Control and Optimization in Applied Mathematics (COAM), 4, pp 1–14.

[41] Tirkolaee E. B., Aydın N. S., Ranjbar-Bourani M., Weber G. W. (2020), A robust bi-objective mathematical model for disaster rescue units allocation and scheduling with learning effect, Computers and Industrial Engineering, 149, 106790.
[42] Tirkolaee E. B., Mahdavi I., Esfahani M. M. S., Weber G. W. (2020), A robust green location-allocation-inventory problem to design an urban waste management system under uncertainty, Waste Management, 102, pp 340-350.
[43] Weber A., (1929), Uber den Standort der Industrient", Tubingen, (1909), English Trans.: Theory of Location of Industries, C.J., Friedrich, ed., and Trans., Chicago University Press, Chicago, Illinois.
[44] Weiszfeld E., (1937), Sur le point par lequel la somme des distances de n points donns est minimum, Tohoku Math., 43, pp 355–386.
[45] Wesolowsky G. O., Love R. F., (1971), Location of facilities with rectangular distance among point and area destinations, Naval Research Logistics Quarterly, 1, pp 83-90.