A new projection technique for developing a Liu-Storey method to solve nonlinear systems of monotone equations

M M Mahdi¹ and M A K Shiker¹,²

¹Department of Mathematics, College of Education for Pure Sciences, University of Babylon, Babil-Iraq.
E-mail: ¹mohmath44@gmail.com, ²mmtmmlhh@yahoo.com
²Corresponding Author.

Abstract. The projection technique is a very important method and efficient for solving unconstraint optimization and nonlinear equations. In this study, we developed a Liu-Storey (LS) algorithm for solving monotone equations of nonlinear systems. The new algorithm satisfies the sufficient descent condition and it’s a suitable method of large scale equations for its limited memory. We established a global convergence of suggest method under the mild conditions. Numerical results proved that the new algorithm works well and promising.

Keywords: Projection Algorithm, Monotone Equations, Nonlinear Systems and Conjugate Gradient Method.

1. Introduction

As we know, the projection approach is a very simple iterative method to find a solution vector \( \mathbf{x}^* \) of nonlinear systems:

\[
F(x) = 0, \quad x \in \mathbb{R}^n,
\]

s.t. \( F: \mathbb{R}^n \rightarrow \mathbb{R}^n \) is continuous mapping and monotonicity condition hold, i.e. \( \langle F(x) - F(y), x - y \rangle \geq 0, \forall x, y \in \mathbb{R}^n \). This problem arises in various applications in applied mathematics, power engineering, economics and chemical systems. For example, the variational inequality [1], the problems in proximal algorithm with Bregman distances [2], the problems of economic equilibrium [3] can be reformulated as (1.1). For solving systems of equation, there are many numerical methods including the Newton method [4], derivative-free method [5] and projection technique based gradient direction [6]. Many from that approaches are iterative process begin with \( x_k \), the next iterate is \( x_{k+1} = x_k + \alpha_k d_k \), \( k \in \mathbb{N} \), where \( d_k \) is called search direction while \( \alpha_k \) denote step length. We mixed the conjugate gradient method with projection algorithm to be a suitable to deal with large-scale equations. In 1964 Goldstein [7] introduced the first projection technique for convex problems in Hilbert spaces. It was then Solodov and Svaiters [8] extended Goldstein method and constructed a hyperplane \( \mathcal{H}_k \) that strictly separates \( x_k \) from the solution set of (1.1) i.e.

\[
\mathcal{H}_k = \{ x \in \mathbb{R}^n | F(z_k)^T(x - z_k) = 0 \}
\]

s.t. \( z_k = x_k + \alpha_k d_k \) is created by employing a line search condition with the direction \( d_k \) s.t. \( F(z_k)^T(x_k - z_k) > 0 \). With the monotonicity of \( F \), we own that for each \( x^* \) s.t. \( F(x^*) = 0 \),

\[
F(z_k)^T(x^* - z_k) \leq 0.
\]

Yet, by Solodov and Svaiters [8] the following approximation \( x_{k+1} \) is constructed by projecting \( x_k \) onto \( \mathcal{H}_k \) i.e.

\[
x_{k+1} = P_{\mathcal{H}_k}[x_k - \delta F(z_k)],
\]

[1] Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.
Published under licence by IOP Publishing Ltd
\[ \mathcal{S}_k = \frac{F(z_k)^T(x_k - z_k)}{\|F(z_k)\|^2}, \]

Where \( \Omega \subseteq \mathbb{R}^n \), be a closed convex.

Recently, the Liu-Storey (LS) conjugate gradient formula \([9]\) is

\[ d_{k+1} = \begin{cases} -F_{k+1} + \frac{\alpha_k F_k d_k}{\|F_k d_k\|^2} & k \geq 1 \\ -F_{k+1} & k = 0 \end{cases} \]

Where \( y_k = F_k - F_{k-1} \). Depending on the last method, we will suggest the developed (LS) form for solving nonlinear systems \((1.1)\) such that

\[ d_{k+1} = \begin{cases} \lambda_k F_k + \frac{F_k^T w_k d_k - d_k^T F_k w_k}{\max\{\|d_k\|, \|w_k\|, \|d_k w_k\|, -d_k^T F_k\}} & k \geq 1 \\ -F_k & k = 0 \end{cases} \] \hspace{1cm} (1.3)

where \( y_k = F_k - F_{k-1} + r \times s_k \), \( s_k = z_k - x_k \), \( r > 0 \), \( \lambda, \theta > 0 \) is fixed. It is simply to offer that our formula can be reduced to a criterion Liu-Storey technique if applied exact line search. The new line search of the proposed algorithm:

\[ -F(x_k + \alpha_k d_k)^T d_k \geq \frac{\tau}{2} \gamma \alpha_k \|d_k\|^2, \quad \text{s.t.} \quad \alpha_k = \rho \Omega \] \hspace{1cm} (1.4)

where \( \rho, \tau, \Omega \) and \( \gamma \in (0, \infty) \). We will adopt the projection based technique to present a developed (LS) gradient method with projection approach to solve monotone equations of nonlinear systems.

In our work, we discuss a developed (LS) projection algorithm for nonlinear systems \((1.1)\). In the next section, we introduce the new algorithm with some assumptions and analysis it's the global convergence. Finally, some numerical experiments and conclusion are presented in the last section.

2. Projection Based Method

Now, projection gradient technique is another efficient algorithm to solve large scale unconstrained minimization problem:

\[ \min_{x \in \mathbb{R}^n} f(x), \] \hspace{1cm} (2.1)

where \( f : \mathbb{R}^n \to \mathbb{R} \) is smooth nonlinear function because of its simplicity and low storage requirements. The steps of a new algorithm are stated as follows:

2.1. New projection algorithm (MOH2):

1. Given initial point \( x_0 \in \mathbb{R}^n \), and parameters \( s, \theta > 0 \) and \( \rho, \tau, \epsilon, \gamma, \Omega \in (0, 1) \).
2. If \( \|F(x_k)\| \leq \varepsilon \) break, Otherwise find \( d_k \) by \((2.5)\).
3. Determine \( \alpha_k = \rho \Omega \), such that

\[ -F(x_k + \alpha_k d_k)^T d_k \geq \frac{\tau}{2} \gamma \alpha_k \|d_k\|^2 \]

4. Let \( z_k = x_k + \alpha_k d_k \).
5. If \( \|F(z_k)\| \leq \varepsilon \) break and set \( x_{k+1} = z_k \). Otherwise compute \( x_{k+1} \) by \((6)\).
6. Let \( k = k + 1 \), and return to stage (2).

3. Global Convergence Test

In this part, we investigate the global convergence of the offer approach and we need some necessary assumptions:

- B1. The solution set of nonlinear equations \((1.1)\) is nonempty.
- B2. The mapping \( F(x) \) is monotone and Lipschitz continuous of \( \mathbb{R}^n \), i.e., \( \exists L > 0 \), s.t.

\[ \|F(x) - F(y)\| \leq L \|x - y\|, \quad \text{for all} \quad x, y \in \mathbb{R}^n. \] \hspace{1cm} (3.1)

In the next lemma we display that the new algorithm (MOH2) has a sufficient descent condition.
3.1. **Lemma:** For each \( k \geq 0 \), we have
\[
F(x_k)^T d_k = -\lambda_k \| F_k \|^2
\]  
(3.2)
And
\[
\| d_k \| \leq \left( \lambda + \frac{2}{\theta} \right) \| F_k \|
\]  
(3.3)

**Proof:** When \( k = 0 \), then (3.2) and (3.3) holds, since \( d_0 = -F(x_0) \), from the definition of \( d_k \) in (1.3), we have
\[
d_{k+1}^T F_k(x_k) = -\lambda \| F(x_{k+1}) \|^2 + \left[ \frac{F_k^T w_k d_k - d_k^T F_{k+1} w_k}{\max \{ \theta \| d_k \| \| w_k \|, \| d_k^T w_k \| - d_k^T F_k \}} \right] F_{k+1} = -\lambda \| F(x_{k+1}) \|^2.
\]
Thus (3.2) hold for all \( k \geq 1 \), and
\[
\| d_{k+1} \| = \left\| -d_k^T w_k \theta \| w_k \| \right\| \leq \lambda \| F(x_{k+1}) \| + \left[ \| w_k \| \| d_k \| \| F_{k+1} \| + \| w_k \| \| d_k \| \| F_{k+1} \| \| d_k \| \right] \max \{ \theta \| d_k \| \| w_k \|, \| d_k^T w_k \| - d_k^T F_k \}
\]
\[
\| d_{k+1} \| \leq \left( \lambda + \frac{2}{\theta} \right) \| F(x_{k+1}) \|.
\]
And the inequality follows form
\[
\max \{ \theta \| d_k \| \| w_k \|, \| d_k^T w_k \| - d_k^T F_k \} \geq \theta \| d_k \| \| w_k \|.
\]
Then (3.3) is hold.

Now, we derive some properties of algorithm (2.1) and show that the line search is well defined.

3.2. **Lemma:** Let assumptions (B1, B2) satisfied, impels that algorithm (2.1) will produce an approximation \( x_k = x_k + \alpha_k d_k \) in a limited number of backtracking procedure.

**Proof:** Assume \( \| F(x_k) \| \rightarrow 0 \) not satisfy, or algorithm (2.1) breaks. Then \( \exists \epsilon > 0 \) satisfying \( \| F(x_k) \| > \epsilon \) for all \( k \geq 0 \).

We use a contradiction to establish this lemma, assume that the property (1.4) not satisfy for several iteration indexes \( k_\ast \).

Set \( \alpha_k = \rho \geq 0 \). It can be concluded
\[
- F(x_k + \alpha_k, d_k, x_k) d_k, x_k < \frac{1}{2} \gamma \alpha_k \| (x_k) \|^2.
\]
By assumptions B1,B2 and (3.2) in Lemma (3.1), we have
\[
\| F(x_k) \|^2 = - F(x_k)^T d_k = [ F(x_k^k + \alpha_k, d_k - F(x_k)^T d_k - F(x_k^k + \alpha_k, d_k^k) ]^T d_k
\]
\[
= L \alpha_k \| (d_k) \|^2 + \lambda \frac{\gamma}{2} \| (d_k) \|^2 + \lambda \frac{\gamma}{2} \| (d_k) \|^2
\]
\[
\| F(x_k) \|^2 = \alpha_k \| (d_k) \|^2 \left[ L + \frac{\gamma}{2} \right]
\]
by assumption B2, we conclude that \( \exists M > n \) s.t.
\[
\| F(x_k) \|^2 \leq M.
\]
Thus, we have
\[
\alpha_k > \left[ \frac{\| F(x_k) \|^2}{L + \frac{\gamma}{2} \| (d_k) \|^2} \right] > \frac{\epsilon^2}{\left[ L + \frac{\gamma}{2} \right] \left[ \lambda M + \frac{\gamma}{2} \right] ^2} > 0.
\]
This impels contradicts with definition of \( \alpha_k \). So the line search (3.4) can hold a nonnegative step length \( \alpha_k \) in a limited number of backtracking steps its well defined.

Now, the next lemma like to Lemma (3.1) in Solodov and Svaiter [8], so we omit the proof.

3.3. **Lemma:** Suppose that assumptions (B1, B2) satisfied and the sequence \( \{ x_k, z_k \} \) is produced by algorithm (2.1), for all \( x^\ast \) is s solution of (1.1) s.t. \( F(x^\ast) = 0 \), then
And, the sequences \( \{x_k, z_k\} \) are bounded, and
\[
\begin{align*}
\lim_{k \to \infty} \|x_k - z_k\| &= 0 \\
\lim_{k \to \infty} \|x_{k+1} - x_k\| &= 0.
\end{align*}
\]

3.4. Theorem: Suppose that assumptions (B1, B2) satisfied and the sequence \( \{x_k\} \) be produced by algorithm (2.1), then
\[
\lim_{k \to \infty} \inf \|F_k\| = 0.
\]

Proof: Assume that (3.8) is not hold. Set a fixed \( \epsilon > 0 \) s.t. \( \|F_k\| \geq \epsilon \) this with (3.2) implies that
\[
\|d_k\| \geq \lambda \|F_k\| \geq \lambda \epsilon, \quad \forall k \geq 0.
\]

From relations (3.5), (3.6) and (3.7), we obtain
\[
\|d_{k+1}\| \leq \left(1 + \frac{2}{\theta}\right)\|F_{k+1}\| \leq \left(1 + \frac{2}{\theta}\right)\lambda \epsilon, \forall k \geq 0.
\]

This mean a sequence \( \{d_k\} \) is bounded. Then \( \exists N_1 \) infinite index and an a limit point \( d \) holds
\[
\lim_{k \to \infty} \|d_k\| = d, \quad k \in N_1.
\]

Via the boundedness of \( \{x_k\} \) in lemma (3.3), we conclude that \( \exists \) there exists a limit point \( \bar{x} \) and there is set \( N_2 \subset N_1 \) be infinite index holds
\[
\lim_{k \to \infty} x_k = \bar{x}, \quad k \in N_2.
\]

From lemma (3.2) and lemma (3.3), w.e have
\[
\alpha_k \|d_k\| \to 0, \quad k \to \infty.
\]

this together with (3.10), we get
\[
\lim_{k \to \infty} a_k = 0.
\]

From (1.4) we have
\[
-F(x_k + a_k d_k) d_k \leq \frac{\tau}{2} \gamma \alpha_k \|d_k\|^2.
\]

Where \( \alpha_k = \frac{d_k}{\rho} \)

Therefore, taking the limit as \( k \to \infty \) in both sides of (3.11) for all \( k \in N_2 \) generates \( F(\bar{x})^T \bar{d} > 0 \).

In the other term, by making the limit as \( k \to \infty \) in both terms of (3.2) for all \( k \in N_2 \), such that \( F(\bar{x})^T \bar{d} \leq 0 \). Which generates a contradiction. \( \square \)

4. Numerical Experiments

In the present section, we shown the our results of numerical experiments to analyze the performance of (MOH2) and compare it with the three famous methods, a scaled derivative-free projection method (UU)[10], a modified Liu-Story conjugate projection method (LS)[11] and a projection based technique (DFBP)[12].

In the suggested algorithm, we used the parameters: \( \rho = 0.7, \sigma = 0.1, \tau = 0.4, \theta = 2, \beta = 0.5 \) and \( \epsilon = 10^{-4} \). The parameters in the LS, PDFB and UU come from [10], [11] and [12] respectively. We take on the similar finish condition for each the forth algorithms i.e. we break them when the upper number of approximation override 500000 or the inequality \( \|F(x_k)\| \leq 10^{-4} \) is satisfied. All algorithms written in MATLAB program R2014a and turn on a PC (win8) CPU 2.30 GHz and 4 GB RAM where all these method applied in the same computer.

We solved 7 constraint test problem see Awwal et al. [3] by using 8 initial different starting point similar to the problems in [13, 14, 15], such that
\[
\begin{align*}
x_0 &= (10,10,...,10)^T, \quad x_1 = (-10,-10,...,-10)^T, \quad x_2 = (1,1,...,1)^T, \quad x_3 = (-1,-1,...,-1)^T \\
x_4 &= (1,\frac{1}{2},...,\frac{1}{n})^T, \quad x_5 = (0,1,0,1,...,1)^T, \quad x_6 = (\frac{1}{n},\frac{1}{n},...,\frac{1}{n})^T, \quad x_7 = (1,\frac{1}{n},...,\frac{1}{n},0)^T.
\end{align*}
\]

The preliminary numerical experiments are reported in tables (4.1) and (4.2) for number of iteration (Ni), number of function evaluations (Nf), CPU time (CPU), probability (Prob) and dimension (Dim).
It can be observed from the tables that our proposed MOH2 method wins higher percentage of the numerical experiments. Numerical results listed in tables (4.1) and (4.2) show that the new method is efficient for solving problem (1.1).

The present performance profile of number of iteration in figure (1), performance number of evolutions in figure (2) and performance of CPU time in figure (3). The performance of suggested method (MOH2) it is obvious from these figures much best than LS, DFBP and UU methods, whenever, the introduced methods are efficiently and promising.

### Table 4.1: Numerical Results

| P. Dim. | S.P. | New | LS | DFBP | UU |
|---------|------|-----|----|------|----|
| x₀ 20000 | 74 | 150 | 30 | 322 | 56 | 171 | 261 | 523 |
| x₁ 20000 | 74 | 150 | 30 | 322 | 56 | 171 | 261 | 523 |
| x₂ 20000 | 62 | 126 | 5  | 22  | 33 | 94  | 269 | 539 |
| x₃ 20000 | 62 | 126 | 5  | 22  | 33 | 94  | 269 | 539 |
| x₄ 20000 | 40 | 82  | 43 | 175 | 10 | 22  | 170 | 341 |
| x₅ 20000 | 52 | 106 | 3  | 12  | 7  | 16  | 228 | 457 |
| x₆ 20000 | 60 | 122 | 70 | 283 | 31 | 85  | 260 | 521 |
| x₇ 20000 | 60 | 122 | 70 | 283 | 31 | 85  | 260 | 521 |
| x₀ 50000 | 74 | 150 | 30 | 322 | 29 | 60  | 304 | 611 |
| x₁ 50000 | 72 | 146 | 33 | 368 | 135| 412 | 89  | 180 |
| x₂ 50000 | 62 | 126 | 5  | 22  | 28 | 58  | 208 | 538 |
| x₃ 50000 | 63 | 128 | 13 | 101 | 123| 372 | 92  | 187 |
| x₄ 50000 | 40 | 82  | 58 | 239 | 18  | 38  | 196 | 340 |
| x₅ 50000 | 52 | 106 | 3  | 12  | 24 | 50  | 207 | 456 |
| x₆ 50000 | 60 | 122 | 92 | 374 | 27 | 56  | 259 | 520 |
| x₇ 50000 | 60 | 122 | 92 | 374 | 27 | 56  | 259 | 520 |
| x₀ 10000 | 1122083 | 964783 | 140299 | 1199292 | 50582 | 140852 | 80563 | 161129 |
| x₁ 10000 | 1135515 | 1073878 | 152995 | 1309110 | 55873 | 156143 | 89469 | 178941 |
| x₂ 10000 | 99605 | 789574 | 118372 | 1009956 | 41605 | 115103 | 65675 | 131353 |
| x₃ 10000 | 117988 | 935984 | 136851 | 1168990 | 49004 | 135936 | 77670 | 155343 |
| x₄ 10000 | 1101372 | 806584 | 120253 | 1025846 | 42163 | 116584 | 66492 | 132987 |
| x₅ 10000 | 98007 | 780850 | 116540 | 993990 | 40850 | 112790 | 64229 | 128461 |
| x₆ 10000 | 41822 | 326437 | 50627 | 42438 | 21682 | 42438 | 28215 | 56433 |
| x₇ 10000 | 42438 | 334940 | 50627 | 42438 | 21682 | 42438 | 28215 | 56433 |
### Table 4.1: Numerical Results – continued

| P. Dim. | S.P | New | LS | DFBP | UU |
|---------|-----|-----|-----|------|----|
| 10000   | x₀  | 161 | 641 | 1218 | 21600| 386 | 1057 | 753 | 1509 |
|         | x₁  | 151 | 561 | 1255 | 21712| 385 | 1055 | 754 | 1511 |
|         | x₂  | 150 | 558 | 1220 | 21610| 387 | 1059 | 753 | 1509 |
|         | x₃  | 146 | 533 | 1218 | 21598| 385 | 1055 | 753 | 1509 |
|         | x₄  | 156 | 599 | 1218 | 21600| 387 | 1059 | 753 | 1509 |
|         | x₅  | 147 | 534 | 1218 | 21599| 385 | 1055 | 753 | 1509 |
|         | x₆  | 160 | 636 | 1218 | 21599| 388 | 1061 | 753 | 1509 |
|         | x₇  | 158 | 617 | 1218 | 21599| 390 | 1065 | 753 | 1509 |
| 5000    | x₀  | 70  | 142 | 21  | 178  | 58  | 169  | 311 | 624  |
|         | x₁  | 72  | 146 | 36  | 359  | 64  | 187  | 322 | 646  |
|         | x₂  | 64  | 130 | 6   | 30   | 40  | 115  | 283 | 568  |
|         | x₃  | 67  | 136 | 12  | 80   | 50  | 145  | 298 | 598  |
|         | x₄  | 66  | 134 | 9   | 55   | 46  | 133  | 292 | 586  |
|         | x₅  | 66  | 134 | 9   | 53   | 45  | 130  | 291 | 584  |
|         | x₆  | 65  | 132 | 8   | 45   | 43  | 124  | 288 | 578  |
|         | x₇  | 65  | 132 | 8   | 45   | 43  | 124  | 288 | 578  |
| 50000   | x₀  | 493 | 3130| 11113| 22417| 19422| 38854| 310748| 621498|
|         | x₁  | 64  | 132 | 30  | 289  | 408 | 1685 | 174 | 350  |
|         | x₂  | 487 | 3120| 11089| 22180| 19411| 38824| 310670| 621342|
|         | x₃  | 63  | 128 | 6   | 38   | 224 | 825  | 122 | 246  |
|         | x₄  | 49  | 130 | 636 | 1274 | 1119 | 2240 | 17977 | 35956  |
|         | x₅  | 453 | 3045| 10880| 21762| 19042| 38086| 304721| 609444 |
|         | x₆  | 475 | 3052| 10938| 21878| 19147| 38296| 306459| 612920 |
|         | x₇  | 475 | 3052| 10938| 21878| 19147| 38296| 306457| 612916 |

**Figure 1.** Performance of the iterations number.  
**Figure 2.** Performance of the function evaluations.
Table 4.2: Numerical results (CPU time)

| P. Dim. | S. P | CPU time |
|---------|------|----------|
|         |      | New      | LS  | DFBP  | UU     |
| 20000   | x_0  | 0.64065  | 4.78125 | 0.75000 | 3.81250 |
| 20000   | x_1  | 0.65625  | 3.31250 | 0.79687 | 2.75000 |
| 20000   | x_2  | 0.51562  | 0.07812 | 0.37500 | 2.40620 |
| 20000   | x_3  | 0.60937  | 0.12500 | 0.35937 | 1.87500 |
| 20000   | x_4  | 0.35937  | 0.87500 | 0.14062 | 1.07812 |
| 20000   | x_5  | 0.45312  | 0.09375 | 0.07812 | 1.43750 |
| 20000   | x_6  | 0.50000  | 1.50000 | 0.40625 | 1.79687 |
| 20000   | x_7  | 0.42187  | 1.26562 | 0.48437 | 1.67187 |
| 50000   | x_0  | 0.68750  | 4.48437 | 0.31250 | 4.37500 |
| 50000   | x_1  | 0.57812  | 3.46875 | 1.73437 | 0.93750 |
| 50000   | x_2  | 0.56250  | 0.06250 | 0.18750 | 2.48437 |
| 50000   | x_3  | 0.51562  | 0.46875 | 1.40600 | 0.76562 |
| 50000   | x_4  | 0.39062  | 1.17187 | 0.14062 | 1.10937 |
| 50000   | x_5  | 0.45312  | 0.0312  | 0.25000 | 1.54687 |
| 50000   | x_6  | 0.48437  | 1.60937 | 0.26562 | 1.7187 |
| 50000   | x_7  | 0.50000  | 1.43750 | 0.21875 | 1.87500 |
| 50000   | x_0  | 3081.03125 | 4109.140625 | 458.90625 | 579.59375 |
| 50000   | x_1  | 3416.42187 | 4762.203125 | 501.84375 | 649.46875 |
| 50000   | x_2  | 2751.17187 | 3138.265625 | 365.04687 | 470.57812 |
| 50000   | x_3  | 3071.59375 | 3663.25000 | 432.87500 | 556.98437 |
| 50000   | x_4  | 2537.18750 | 4082.46875 | 378.15625 | 473.28125 |
| 50000   | x_5  | 2453.28125 | 3673.75000 | 366.93750 | 461.01562 |
| 50000   | x_6  | 1028.90625 | 1352.04687 | 155.93750 | 200.20312 |
| 50000   | x_7  | 1054.93750 | 1644.07812 | 154.65625 | 201.29687 |
| 10000   | x_0  | 0.20312  | 1.51562 | 0.81250 | 0.60937 |
| 10000   | x_1  | 0.35937  | 5.18750 | 1.23437 | 0.73437 |
| 10000   | x_2  | 0.20312  | 0.95312 | 0.43750 | 0.32812 |
| 10000   | x_3  | 0.23437  | 3.10937 | 0.85937 | 0.43750 |
| 10000   | x_4  | 0.15625  | 2.90625 | 0.85937 | 0.32812 |
| 10000   | x_5  | 0.20312  | 2.65625 | 0.73437 | 0.29687 |
| 10000   | x_6  | 0.23437  | 1.35937 | 0.87500 | 0.23437 |
| 10000   | x_7  | 0.26562  | 1.90625 | 0.64060 | 0.25000 |
Table 4.2: Numerical results (CPU time) - continued

| P. Dim. | S.P | CPU time | New | LS  | DFBP | UU  |
|---------|-----|----------|-----|-----|------|-----|
| 10000   | x₀  | 0.28120  | 11.25000 | 1.45312 | 2.45312 |
| 10000   | x₁  | 0.21875  | 9.04687  | 1.06250 | 1.50000 |
| 10000   | x₂  | 0.21875  | 8.46870  | 0.85937 | 1.37500 |
| 10000   | x₃  | 0.18750  | 5.84370  | 0.79687 | 1.20312 |
| 10000   | x₄  | 0.21875  | 5.76562  | 0.78125 | 1.00010 |
| 10000   | x₅  | 0.18750  | 5.84370  | 0.73437 | 0.85937 |
| 10000   | x₆  | 0.25000  | 5.71870  | 0.76562 | 0.78125 |
| 10000   | x₇  | 0.21875  | 5.73437  | 0.73437 | 0.78125 |
| 5000    | x₀  | 1.07812  | 3.35937  | 3.18750 | 8.87500 |
| 5000    | x₁  | 1.07812  | 4.82812  | 2.35937 | 5.07812 |
| 5000    | x₂  | 0.92187  | 0.32812  | 1.48437 | 4.18750 |
| 5000    | x₃  | 1.03125  | 0.82812  | 1.64062 | 4.18750 |
| 5000    | x₄  | 0.79687  | 0.50000  | 1.60937 | 4.09375 |
| 5000    | x₅  | 0.93750  | 0.53125  | 1.59375 | 4.09375 |
| 5000    | x₆  | 0.98437  | 0.43750  | 1.39062 | 4.09375 |
| 5000    | x₇  | 0.87500  | 0.34375  | 1.32812 | 4.03125 |
| 50000   | x₀  | 8.15625  | 92.17187 | 142.21875 | 1802.67180 |
| 50000   | x₁  | 0.54687  | 1.51562  | 6.45312  | 1.12500 |
| 50000   | x₂  | 8.03125  | 84.0468  | 136.01562 | 1627.96870 |
| 50000   | x₃  | 0.50000  | 0.09375  | 2.53125  | 0.67187 |
| 50000   | x₄  | 0.43750  | 4.98437  | 7.48437  | 93.81250 |
| 50000   | x₅  | 7.64062  | 82.20312 | 131.87500 | 1588.15620 |
| 50000   | x₆  | 7.73437  | 82.43750 | 133.81250 | 1661.25000 |
| 50000   | x₇  | 7.53125  | 82.26560 | 134.34375 | 1887.56250 |

Figure 3. Performance of the CPU time
5. Conclusions
In the present paper, we introduce a developed Liu-Story (LS) projection type based gradient algorithm to solve the nonlinear systems of monotone equations. The new algorithm is a suitable method of large scale equations due to its low memory requirements. The proposed method satisfies the sufficient descent condition and the global convergence with some suitable assumptions. The numerical experiments indicate that the proposed technique is efficient and very competitive to solve nonlinear systems of monotone equations.

References

[1] Zhang L and Zhou W J 2006 Spectral gradient projection method for solving nonlinear monotone equations, J. Comput. Appl. Math., 196, p. 478–484.
[2] Iusem A N and Solodov M V 1977 Newton-type methods with generalized distances for constrained optimization, Optim., 41, 257-278.
[3] Awwal A M, Kumam P, Abubakar A B and Wakili A 2018 A projection Hestenes-Stiefel like method for monotone nonlinear equations with convex constraints, Thai Journal of Mathematics, 16 p.181-199.
[4] Shiker M A K and Sahib Z 2018 A modified technique for solving unconstrained optimization, J. Eng. Applied Sci., 13 9667-9671.
[5] Li Q N and Li D H 2011 A class of derivative-free methods for large-scale nonlinear monotone equations, IMA J. Numer. Anal. 31 1625–1635.
[6] Dreeb N K, Hashim K H, Mahdi M M, Wasi H A, Dwail H H, Shiker M A K and Hussein H A 2019 Solving a large-scale nonlinear system of monotone equations by using a projection technique, Journal of Engineering and Applied Sciences, 14 10102-10108.
[7] Goldstein A A 1964 Convex programming in Hilbert space, Amer. Math. Soc. 70, 709–710.
[8] Solodov M V and Svaiter B F 1998 A globally convergent inexact Newton method for systems of monotone equations, in: M. Fukushima L Qi (Eds.), Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods, Kluwer Academic Publishers, 355-369.
[9] Liu Y and Storey C 1991 Efficient generalized conjugate gradient algorithms, Part I: Theory, J. Optim. Theory Appl., 69 129-137.
[10] Koorapetse M, Kaelo1 P and Offen1 E R 2019 A Scaled Derivative-Free Projection Method for Solving Nonlinear Monotone Equations, Bulletin of the Iranian Mathematical Society, Vol. 45, p755-770.
[11] Hu Y and Wei Z 2014 A Modified Liu-Storey Conjugate Gradient Projection Algorithm for Nonlinear Monotone Equations, International Mathematical Forum, Vol. 9 no. 36 1767-1777.
[12] Shiker M A K and Amini K 2018 A new projection-based algorithm for solving a large scale nonlinear system of monotone equations, Croatian operational research review, corrr, 9, 63-73.
[13] Hashim K H, Dreeb, N K, Dwail, H H, Mahdi M M, Wasi, H A, Shiker, M A K and Hussein H A 2019 A new line search method to solve the nonlinear systems of monotone equations, Journal of Engineering and Applied Sciences, 14 10080-10086.
[14] Amini K, Shiker M A K and Kimiae M 2016 A line search trust-region algorithm with nonmonotone adaptive radius for a system of nonlinear equations. 4 OR- Journal of operation research, 14 (2) 133-152.
[15] Hassan Z A H and Shiker M A K 2018 Using of generalized baye’s theorem to evaluate the reliability of aircraft systems. Journal of Engineering and Applied Sciences, (Special Issue13), 10797–10801.