On Sensitivity of Iterated Methods to Initial Points in Solving Nonlinear System of Equations and Predict Methods Behavior Far Away from Solution

Rasoul Hekmati

Independent Researcher. M.SC From Sharif University of Technology, Tehran, Iran

Abstract: In this paper we present an algorithm in order to finding initial points for nonlinear system of equations which numerical methods are unable to have fast convergence when they start their iterations from those points. Identifying bad-behavior initial points is a competition for optimization software to solve problems starting from bad-behavior initial points and this will show the method’s robustness. We also use function approximation theory in order to predict method’s behavior far from solution by various examples. The results are summarized in the form of diagrams for having a general view of method’s performance.

Key words: Iterated methods, nonlinear system of equations, bad-behavior initial points, sensitivity diagrams.

INTRODUCTION

Solving system of nonlinear equations is an important part of optimization literature:

\[ F(x) = 0 \]  

Where \( F: x \rightarrow \mathbb{R}^n \) is a continuously mapping. It had investigated at numerous papers and there are a lot of methods for solving this kind of problems (Grapsa and Vrahatis, 1990), (Nocedal, 1999).

Bound-constrained nonlinear system of equations defined as:

\[ F(x) = 0, \quad x \in \Omega \]

Which \( \Omega = \{ x | l \leq x \leq u \} \), \( l \in (\mathbb{R} \cup -\infty)^n \), \( u \in (\mathbb{R} \cup \infty)^n \) and \( F \) is as defined to (1.1), arises in many practical, physical, chemical and economic problems (Bullard and Bigler, 1991), (Shacham et al., 2002), (Dirkse, 1995), (Ferris and Pang, 1997), (Floudas et al., 1999), (Hock and Schittkowski, 1981). In (Dennis et al., 1999), (Fletcher and Leyffer, 2003) the case that \( F \) is continuously differentiable have investigated and (Ulbrich, 2000) address the study of nonsmooth equations. Themethod given in (Kanzow, 2001), (Qi et al., 2004), (Kozakevich et al., 1997) is active-set-type methods.

Line search methods are studied in paper (Kanzow et al., 2005) and the papers (Dennis et al., 1999), (Ulbrich, 2000), (Fletcher and Leyffer, 2003) presents Trust region strategy. Fletcher and Leyffer propose a trust-region filter SQP method in (Fletcher and Leyffer, 2003); the method that Ulbrich proposed in (Ulbrich, 2000) is a newton-like method with projection embedded into a trust-region technique. In (Bellavia et al., 2003) Bellavia, Maconi and Morini develop an affine trust region method that uses a scaling matrix to form an elliptical trust region whose shapes depend on the scaling matrix. These methods are globally convergent and they find a solution to problem (1.1) starting from an arbitrary initial guess or they fail.

When an algorithm is presented, it should be tested on a set of functions to show that the algorithm works and it works better than other algorithms. Testing an algorithm on a large set of functions is tedious and bothersome. Algorithms that succeed from the standard starting points are unable to solve problem from points far away and sometimes they fail. (Hillstrom, 1997) was one of the first to point out the need of testing software at nonstandard starting points. He proposed using random starting points, but it produced large amounts of data which can be hard to interpret. We’ll try to emphasize problems that solving them with special initial points are a challenging work for algorithms.

We design a simple algorithm that use a trial method for solving some famous constrained and unconstrained system of equations as main body of algorithm and discern bad-behavior initial points by running the trial method frequently. By distinguishing this points for various problems we’ll have a criterion to test future software’s robustness and flawless more cheaply and easily. Then a special diagram for better analyses of results will be depicted and an approximation theory approach will be used for predicts method’s behavior far away from the solution. This approximated function has many forms like polynomial, rational, power, Gaussian,
exponential, fourier, spline, sum of sin functions, etc. the appropriate approximate will be elected by iterations data format.

The paper is organized as follows. In section 2 the algorithm of finding bad-behavior initial points is presented, in section 3 we illustrate the implementation issues and obtain numerical results. Diagrams of test functions that represent a better view of function’s sensitivity to initial points are demonstrated and finally some bad-behavior and critical points for test problems are reported.

2. Description of Method:

First of all we should choose one of the algorithms designed for solving (1.1) and an algorithm for solving (1.2) like the methods addressed in section 2. We choose STRN (scaled trust region newton method) from (Bellavia et al., 2003). This method had presented for solving bound-constrained nonlinear system of equations but by some manipulating in its codes, STRN is able to solve both problems (1.1), (1.2),to this Purpose we set $l = -inf, u = inf, D = I$.

STRN like other iterated methods uses a starting point $x_0$ and each iteration computes a search direction $p_k$ and then decides how far to move along that direction. The iteration is given by

$$x_{k+1} = x_k + \alpha_k p_k,$$

where the positive scalar $\alpha_k$ is called the step length. The success of a line search or trust region method depends on effective choices of both the direction $p_k$ and the step length $\alpha_k$, then algorithm starts from $x_0$ and continues till one of the terminating conditions occurs like this: desired solution has obtained upon successful termination, the maximum number of iterations has been reached, the maximum number of F-evaluations has been reached, the trust region radius has become too small, no improvement for the nonlinear residual could be obtained, an overflow would be generated and etc.

For having fast convergence you should estimate the initial point near the solution, now suppose that we set $x_0 = x^*$, where $x^*$ is the solutions of the system. Obviously our method will terminate in first or second iteration:

- iteration = 0 or 1
- $\|F\|_2 = 0$
- $F$ - evaluations = 0 or 1

Where F-evaluations are depended on method. Now suppose the initial points as x-axis and the number of iterations as y-axis, then we have a function that shows sensitivity of method in solving a problem to initial points, this function has a root in $x_0$. For plotting a sample of this function we have this algorithm:

Algorithm for detect some bad-behavior initial-points:

Let $x_0 = x^*$, $k \in N, M \geq 1000$ and T as a string.

For $i=1, 2, ..., k$
1. Let $x = x_0 + i$ or $x = x_0 - i$.
2. Solve (1.1) or (1.2) by a method (here [iteration]=STRN(F,x))
   If termination was successful
   \[
   T (i) = \text{iteration}.
   \]
   If maximum number of iterations or F-evaluations occurs
   \[
   T (i) = M.
   \]
   Else
   \[
   T (i) = \text{‘error’}.
   \]
3. $x_0 = x$
Plot (T).

M means the limiting number of iterations or F-evaluations; ‘error’ depends on method’s properties that we used to run the algorithm. Let $k = 500$ then this algorithm will give us 500 initial points for x-axis from interval$(x_0, x_0 + 500)$ or $(x_0 - 500, x_0)$ notice that we move along dimensions monotonously, it means that if for example $x_0 = (1,1,1)$ then $x_0 + 3 = (4,4,4)$. We are going to be interested at points which T takes M or ‘error’.

3. Numerical Results:

We choose some problems from (Mor’e et al., 1981) for system (1.1) and some problems from the library NLE (Schacham et al., 2002) accessible through the web site: WWW.polymath-software.com/library. It contains over 70 real-life problems whose dimensions vary from a single equation to 14 equations and all of the problems have solutions in the interior of feasible region. Name of problems is associated according to dimension of problem and a number, for example Threeeq2. The stopping tolerances atol=$10^{-8}$ and rtol=0,
maxit=1000 and maxnf=1000 are maximum number of iterations and function evaluations. We run the code for
\( \Delta_0 = 1 \), \( \beta_1 = 0.1 \), \( \beta_2 = 0.25 \), \( \theta = 0.99995 \), \( \delta_1 = 0.25, \delta_2 = 2 \). For more information see (Bellavia et al., 2003), (Bellavia et al., 2004).

3.1 Variations in all Dimensions:
3.1.1 Rosenbrock:
The standard starting point for this problem is \((-1.2, 1)\) and the solution is \(x^* = (1,1)\). STRN solve this problem in 12 iteration and 15 F-evaluations with \(\frac{1}{2}\|F(x)\|^2 = 0.32 \times 10^{-10}\). We run our sensitivity detector algorithm for this problem with \(k = 500\) and plot the figure1 for \(x_0 \in (1,500)\). In figure1 the point 1 in x-axis means \(x^*\) and every unit adds 1 number to its coordinates. The algorithm reports failure because of reaching to maximum number of F-evaluations for some starting points, table1 shows bad-behavior and critical points (more than 500 iterations):

<table>
<thead>
<tr>
<th>Critical points</th>
<th>Iterations</th>
<th>Bad-behavior points</th>
<th>iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>((232,232))</td>
<td>509</td>
<td>((400,400))</td>
<td>Max F-eval</td>
</tr>
<tr>
<td>((233,233))</td>
<td>521</td>
<td>((401,401))</td>
<td>Max F-eval</td>
</tr>
<tr>
<td>((234,234))</td>
<td>533</td>
<td>((402,402))</td>
<td>Max F-eval</td>
</tr>
<tr>
<td>((235,235))</td>
<td>542</td>
<td>((403,403))</td>
<td>Max F-eval</td>
</tr>
<tr>
<td>((236,236))</td>
<td>554</td>
<td>((404,404))</td>
<td>Max F-eval</td>
</tr>
<tr>
<td>((237,237))</td>
<td>559</td>
<td>((405,405))</td>
<td>Max F-eval</td>
</tr>
<tr>
<td>((238,238))</td>
<td>559</td>
<td>((406,406))</td>
<td>Max F-eval</td>
</tr>
<tr>
<td>((239,239))</td>
<td>559</td>
<td>((407,407))</td>
<td>Max F-eval</td>
</tr>
<tr>
<td>((240,240))</td>
<td>553</td>
<td>((408,408))</td>
<td>Max F-eval</td>
</tr>
<tr>
<td>((241,241))</td>
<td>555</td>
<td>...</td>
<td>Max F-eval</td>
</tr>
<tr>
<td>((242,242))</td>
<td>553</td>
<td>((429,429))</td>
<td>Max F-eval</td>
</tr>
</tbody>
</table>

Note that \(g(x^*) = 0\).

Coefficients (with 95% confidence bounds):
\(a_1 = 784.7, b_1 = 425.1, c_1 = 3.871, a_2 = 787.8, b_2 = 402, c_2 = 3.953\)

Fig. 1: Sensitivity of STRN method to initial point in \((1,500)\) (rosenbrock).
a₃ = 10.29, b₃ = 633.6, c₃ = 234.8, a₄ = 1.291, b₄ = 304.1, c₄ = 1.768.

Figure 2 shows this function:

Fig. 2: Gaussian approximation function (rosenbrock).

By approximated function it is predictable that in every interval $E$ with $\mu(E) \approx 500$ STRN will have poor performance (more than 500 iteration) just for 11 initial points by high probability. We do a similar running for $x₀ \in (-500, 1)$, table 2 shows critical points:

<table>
<thead>
<tr>
<th>Critical points</th>
<th>iterations</th>
<th>Bad-behavior</th>
<th>iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>X=(170,170) ... x=(186,186)</td>
<td>More than 500</td>
<td>X=(262,262) ... x=(312,312)</td>
<td>Max F-eval</td>
</tr>
<tr>
<td>X=(244,244) ... x=(261,261)</td>
<td>More than 500</td>
<td>X=(417,417) ... x=(499,499)</td>
<td>Max F-eval</td>
</tr>
</tbody>
</table>

Figure 3 demonstrates that.

Fig. 3: Sensitivity of STRN method to initial point in (1,-500) (rosenbrock).
The algorithm reports failure because of reaching to maximum number of F-evaluations for bad-behavior points as it is in table 2. The periodic behavior of problem that gets worse by going far away from solution is an appropriate challenging for optimization software.

3.1.2 Helical Valley:

The standard starting point for this problem is (-1,0,0) and the solution is $x^* = (1,0,0)$. STRN solve this problem in 8 iteration and 10 F-evaluations with $\frac{1}{2}\|F(x)\|^2 = 0.75 \times 10^{-7}$. Figure 4 shows its sensitivity by starting from $x^*$. The algorithm didn’t report any failure. X-axis in figure 4 is from 1 to 500, and initial points varies from $x = (1, 0, 0)$ to $x = (500,499,499)$.

By using 6th degree Polynomial approximation we have an estimate of iterations as in figure 5. Linear model Poly6:

$$g(x) = p_1x^6 + p_2x^5 + p_3x^4 + p_4x^3 + p_5x^2 + p_6x + p_7$$

Coefficients (with 95% confidence bounds):

$$p_1 = -1.496e-013, p_2 = 2.193e-010, p_3 = -1.224e-007, p_4 = 3.303e-005, p_5 = -0.004616, p_6 = 0.3648, p_7 = 7.242$$

Fig. 4: Sensivity of STRN method to intial point in (1,500) (helical valley).

Fig. 5: Polynomial approximation (helical valley).
This function shows a little amplitude and no critical point at all; its path reminds us of a trigonometric similar function and this leads us to calculate sin approximation of this data as follows:

Sin function approximation from degree 6:
\[ g(x) = a_1 \sin(b_1 x + c_1) + a_2 \sin(b_2 x + c_2) + a_3 \sin(b_3 x + c_3) + a_4 \sin(b_4 x + c_4) + a_5 \sin(b_5 x + c_5) + a_6 \sin(b_6 x + c_6) \]

Coefficients (with 95% confidence bounds):
\[ a_1 = 53.43, \quad b_1 = 0.006108, \quad c_1 = -0.3602, \quad a_2 = 28.03, \quad b_2 = 0.009033, \quad c_2 = 1.952, \quad a_3 = 2.271, \quad b_3 = 0.03186, \quad c_3 = 0.2379, \quad a_4 = 1.776, \quad b_4 = 0.05128, \quad c_4 = 1.703, \quad a_5 = 1.461, \quad b_5 = 0.08486, \quad c_5 = -2.441, \quad a_6 = 0.9124, \quad b_6 = 0.1699, \quad c_6 = 3.9 \]

Fig. 6: Sin function approximation (helical valley).

The approximated function to merit function is better now and it has acceptable accuracy except to some points. Now we draw figure 7 to analyses method behavior for x = (1, 0, 0) to (-500,499,499):

Fig. 7: Sensivity of STRN method to intial point in (-500,1) (helical valley).

By linear approximation demonstrated in figure 8 we have

Linear model Poly1:
\[ g(x) = p_1 x + p_2 \]

\[ p_1 = 0.01438, \quad p_2 = 18.4 \]
3.1.3 Teneq1b:
For this section we bring a bound- constrained nonlinear problem named threeq4b from (Schacham et al., 2002). There are some standard starting points for this problem like $x_0 = (2, 5, 80, 1, 0, 0, 0, 0, 20, 5)$ and the feasible solution for this problem is $x^* = (2.997, 3.9664, 79.9996, 0.0023, 0.0006, 0.0645, 3.5308, 26.4315, 0.0044)$. STRN solve this problem in 36 iteration and 42 F-evaluations (if we consider the average iterations obtained by running solver for 3 standard initial points as mentioned in (Schacham et al., 2002)). The algorithm reports failure because of reaching to maximum number of F-evaluations for 3 starting points demonstrated in table3:

<table>
<thead>
<tr>
<th>Starting points with maximum F-evaluations</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(17.997, 3.9664, 94.9996, 15.0023, 15.0006, 15.0013, 15.0064, 18.5308, 41.4315, 15.0044)</td>
<td></td>
</tr>
</tbody>
</table>

In numerous points method was unsuccessful because of one of this reasons: the trust region radius had become too small, no improvement for the nonlinear residual be obtained, an overflow be generated and etc. for having a list of this points see conclusion section blow. Figure9 shows its sensitivity by starting from $x^*$, 0 on x-axis means failure:

By using smoothing spline or Gaussian approximation and similar to above discussion we can estimate method’s behavior for this function, but for brevity we ignore that. This problem is hard to be solved for some points and easy for other ones and being able to solve this problem from a good-behavior starting point isn’t advantage for optimization software.

3.2 Variation in One Dimension:
In order to examine other points, we fix all the dimensions of solution and change one variable’s amount, results show some other bad-behavior points as follows.

3.2.1 Rosenbrock:
Change second variable of rosenbrock problem’s solution
The linear polynomial approximation will be perfect. Problem hasn’t a notable critical point and all the initial points like (-1,1), (-2,1), … (-500,1) have a predictable iteration.

By changing first variable of solution and fixing second variable of rosenbrock problem we have figure 11: The results are similar to the case that we change all the variables of solution together as it had shown previously in figure 3 with some differences like at the points (-255,0), (-450,0).

3.2.2 Helical Valley:
Fix second and third variables of solution and change first one in increasing amount we have figure 12 from (1,0,0) to (500,0,0):
Fig. 11: Sensitivity of STRN method to initial point by changing first dimension (rosenbrock).

Fig. 12: Sensitivity of STRN method to initial point by changing first dimension (helical valley).

It reports a constant iteration amount for (260,0,0), ..., (500,0,0) and not a bad-behavior point at all. By fixing second and third variables of solution and change first one in decreasing amount we have figure 13 for (1,0,0) to (-500,0,0).

Fig. 13: Sensitivity of STRN method to initial point by changing first dimension (helical valley).
It is similar to figure 7 and has not any critical point at all. By fixing first and second variables of solution and change third one in increasing amount we have figure 13 for $(1,0,0)$ to $(1,0,500)$.

![Sensitivity of STRN method to initial point by changing third dimension (helical valley).](image)

**Fig. 14:** Sensitivity of STRN method to initial point by changing third dimension (helical valley).

For first time in our experiments helical valley shows bad-behavior points: $(1, 0, 36), (1, 0, 41)\ldots (1, 0, 54)$, because the trust region has become too small.

4. **Conclusion:**

We had a contempt to discern bad behavior initial points and by some experiments we find this points for some problems, to brevity we ignore to bring more examples and eliminate ‘the moving in two dimensions and fix other one’ section in helical valley’s numerical results. In order to have a complete set of this kind of experiments we decide to design an internet database webpage and put the exact diagrams and critical points for various problems and important methodsthere, the mutual critical and bad-behavior points earned with numerous methods will be super-bad-behavior points, by increasing algorithm’s accuracy, the initial points for various problems that are challenges for optimization software will be obtained.

**REFERENCES**


