An Improved Unscented Parameter Estimation Algorithm for Solving Systems of Nonlinear Equations

H. Zhang[a]*, F. Topputo[b], C. Han[a]

[a]*Beihang University, Beijing, 100191, China*

[b]*Politecnico di Milano, Milan, 20156, Italy*

Abstract

An algorithm based on unscented parameter estimation (UPE) is proposed for solving systems of nonlinear equations. By converting the original problem to a new state-space problem, UPE can be used to solve implicit equations. Compared with Newton's method (NM), UPE avoids computing Jacobians and overcomes the difficulty of guessing a good initial approximation. In this work, a hybrid UPE algorithm is proposed. This combines NM and UPE to achieve a good balance between convergence domain and rate. In the first iterations UPE is used to take advantage of its large convergence ability, albeit solution improvements are achieved at slow rate. Once the solution is deemed sufficiently accurate, the algorithm switches to the NM to exploit its fast local convergence and high precision. Numerical examples are implemented to examine the reliability and efficiency of the hybrid UPE algorithm.

Keywords: Newton's method; Systems of nonlinear equations; Root finding; Iterative method; Unscented parameter estimation

1. Introduction

Solving systems of nonlinear equations is an important problem in numerical analysis and applied science. This problem can be precisely stated as to find a vector $\boldsymbol{r} = (r_1, r_2, \ldots, r_n)$ for a given nonlinear function $\boldsymbol{F}$: $\mathrm{D} \subseteq \mathbb{R}^n \to \mathbb{R}^n$ such that $\boldsymbol{F}(\boldsymbol{r}) = 0$. This solution can be obtained as a fixed point of some function $\boldsymbol{G}$: $\mathrm{D} \subseteq \mathbb{R}^n \to \mathbb{R}^n$ by means of fixed point iteration

$$\boldsymbol{x}_{k+1} = \boldsymbol{G}(\boldsymbol{x}_k),\ k = 0,1,\ldots \tag{1}$$

A common and efficient method for solving systems of nonlinear equations, $\boldsymbol{F}(\boldsymbol{x}) = 0$, is Newton's method (NM), which converges quadratically under the conditions that the function $\boldsymbol{F}(\boldsymbol{x})$ is continuously differentiable, and a good initial approximation $\boldsymbol{x}_0$ is given. The generic iteration of Newton's method is (see [1])

$$\begin{aligned} \boldsymbol{J}(\boldsymbol{x}_k)\boldsymbol{d}_k &= -\boldsymbol{F}(\boldsymbol{x}_k) \\ \boldsymbol{x}_k &= \boldsymbol{x}_k + \boldsymbol{d}_k \end{aligned} \tag{2}$$

where $\boldsymbol{J}(\boldsymbol{x}_k)$ is the *n*-by-*n* Jacobian matrix of the function $\boldsymbol{F}(\boldsymbol{x})$ evaluated at the *k*th iteration $\boldsymbol{x}_k$, i.e.,

$$\boldsymbol{J}(\boldsymbol{x}_k) = \left[\nabla F_1(\boldsymbol{x}_k)^T \ \nabla F_1(\boldsymbol{x}_k)^T \ \cdots \ \nabla F_n(\boldsymbol{x}_k)^T \right]^T. \tag{3}$$

Although NM is attractive, it may be difficult to use it in specific problems. At first, $\boldsymbol{J}(\boldsymbol{x})$ may not be analytically available. This is common in many applications, for example, when $\boldsymbol{F}(\boldsymbol{x})$ itself is not even analytic [1]. Moreover, $\boldsymbol{J}(\boldsymbol{x})$ may be singular or ill-conditioned, and so the Newton step $\boldsymbol{d}_k$ is not even defined. In addition, NM may fail to converge if the initial approximation is far from the solution due to its property of having local convergence [1].

In the literature much effort has been put to improve the performance of NM. On the one hand, modifications to avoid the calculation of $\boldsymbol{J}(\boldsymbol{x})$ or to enlarge the convergence domain have been proposed. A quasi-Newton method, which avoids the calculation of $\boldsymbol{J}(\boldsymbol{x})$ and decreases the computational cost, has been proposed in [2]. Also, trust-region techniques have been used to improve robustness and to handle cases when $\boldsymbol{J}(\boldsymbol{x})$ is singular in [3-6]. On the other hand, improvements of the computational order of convergence as well as reductions of the number of operations and function evaluations have been attained. A third-order method based on a quadrature formula has been proposed in [7]. Variants of Newton's method based on trapezoidal and midpoint rules of quadrature have been developed in [8].

* Corresponding author. Tel.: +86 10 82316536; fax: +86 10 8233 9583.
1 PhD Candidate at School of Astronautics, Contact at zhanghongli@buaa.edu.cn
2 Assistant professor at Department of Aerospace Science and Technology, contact at francesco.topputo@polimi.it
3 Professor at School of Astronautics, contact at hanchao@buaa.edu.cn

High order iterative methods have been presented in [9, 10]. Order four and five convergence is reached by means of Adomian decomposition in [11].

In this paper, we propose an algorithm based on unscented parameter estimation (UPE) for solving systems of nonlinear equations. The UPE method is based on the theory of probability, and as such it does not require any differentiation to compute the Jacobians [12]. Also, it can reach convergence when the initial approximation is not good. To improve the existing UPE method further, a hybrid UPE algorithm (H-UPE) is proposed. This combines NM and UPE. UPE is used in the first iterations to take advantage of its large convergence ability. After a few iterations, when solution is deemed sufficiently accurate, the algorithm switches to NM, so exploiting its fast local convergence and high accuracy. In the UPE part, three sampling techniques are presented, namely, 1) the symmetrical sampling, 2) the minimal skew simplex sampling, 3) the spherical simplex sampling. We show that the devised H-UPE method represents a good balance between convergence domain and rate. It is also proved that H-UPE takes less computational time than the original UPE method, and thus it can be used to solve systems of nonlinear equations robustly and efficiently.

The remainder of the paper is structured as follows. In Section 2 background notions on UPE are recalled. In Section 3, the method to solve systems of nonlinear equations through the UPE is presented. In Section 4, the developed hybrid algorithm is presented, and extensively tested in Section 5. Final remarks are given in Section 6.

2. Background

2.1 Unscented Transformation

Let $\boldsymbol{x}$ be an n-dimensional random variable with mean $\bar{\boldsymbol{x}}$ and covariance $\boldsymbol{P}_{xx}$. A second m-dimensional random variable $\boldsymbol{y}$ is related to $\boldsymbol{x}$ through the nonlinear transformation

$$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) \tag{4}$$

where $\boldsymbol{f}$ is any given nonlinear transformation. The objective is to calculate the mean $\bar{\boldsymbol{y}}$ and the covariance $\boldsymbol{P}_{yy}$ of $\boldsymbol{y}$.

The Unscented Transformation (UT) builds on the intuition that *it is easier to approximate a probability distribution than it is to approximate an arbitrary nonlinear function* [13]. The basic UT algorithm is as follows (see [14]).

1) A set of $p+1$ weighted points $S = \{W_i, \boldsymbol{\chi}_i\}$ (sigma points) are determinately chosen to make their mean $\boldsymbol{x}$ and covariance $\boldsymbol{P}_{xx}$, and the weights W_i must obey the condition

$$\sum_{i=0}^{p} W_i = 0 \tag{5}$$

2) Instantiate each sigma point through the nonlinear transformation (4) to yield the set of projected sigma points, $\Gamma = \{W_i, \boldsymbol{y}_i\}$,

$$\boldsymbol{y}_i = \boldsymbol{f}(\boldsymbol{\chi}_i) \tag{6}$$

3) The estimated mean and covariance of $\boldsymbol{y}$ can be approximated using a weighted sample mean and covariance of the posterior sigma points,

$$\begin{aligned} \bar{\boldsymbol{y}} &= \sum_{i=1}^{2n+1} W_i^{(m)} \boldsymbol{y}_i \\ \boldsymbol{P}_{yy} &= \sum_{i=1}^{2n+1} W_i^{(c)} (\boldsymbol{y}_i - \bar{\boldsymbol{y}})(\boldsymbol{y}_i - \bar{\boldsymbol{y}})^T \end{aligned} \tag{7}$$

Despite its deceptively simplicity, the UT has a number of important properties. At first, the UT works with a finite number of sigma points, and thus it naturally lends itself to being used in a “black box” scheme. Given a model with appropriately defined inputs and outputs, the UT can be used to calculate the predicted quantities as necessary for any given nonlinear transformation. Moreover, in the UT, the most expensive operations are $O(n^3)$, which are calculating the Cholesky factorization and the outer products required to compute the covariance of the projected sigma points (see [15]). Besides, the UT effectively evaluates the Jacobian precisely through its sigma point propagation, without the need to perform any

analytic differentiation. Thus the UT can be used to handle the cases that the nonlinear transformation may be discontinuous or the Jacobian may not be analytically available.

2.2 Sampling Strategy

The main issue of the UT is to choose a proper sampling strategy of sigma points, namely, specifying the number, location and weight of the sigma points. The computational cost of the UT is directly proportional to the number of sigma points, and therefore, there is a strong incentive to minimize their number. In this work, three sampling strategies are considered.

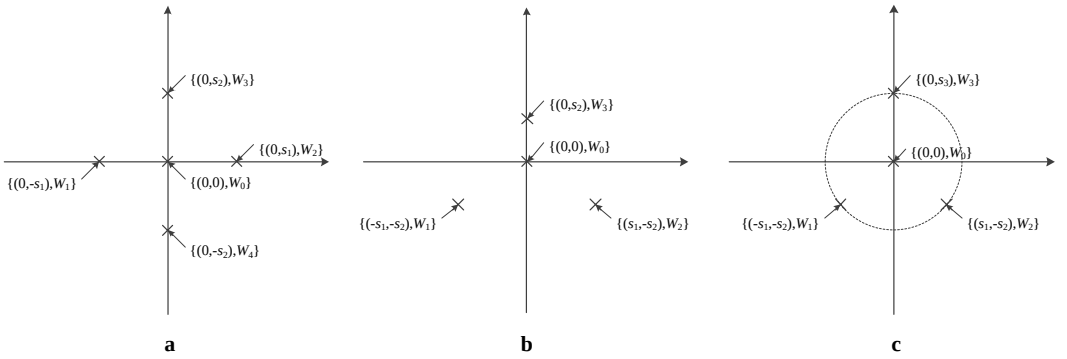

Figure 1. Sampling strategies of the UT (n=2).
(a: symmetric sampling, b: minimal skew simplex sampling, c: spherical simplex sampling)

2.2.1 Symmetric sampling

Symmetric sampling strategy uses $2n+1$ sigma points that are symmetrically-distributed about $\bar{\boldsymbol{x}}$ to match the first two moments of $\boldsymbol{x}$ (see Figure 1a). The sigma points can be determined as [15]

$$\begin{aligned}
&\boldsymbol{X}_0 = \bar{\boldsymbol{x}} \\
&\boldsymbol{X}_i = \bar{\boldsymbol{x}} + (\sqrt{(\mathrm{n}+\kappa)\boldsymbol{P}_{xx}})_i \quad i = 1,\ldots,n \\
&\boldsymbol{X}_{i+n} = \bar{\boldsymbol{x}} - (\sqrt{(\mathrm{n}+\kappa)\boldsymbol{P}_{xx}})_{i-n} \quad i = 1,\ldots,n \\
&W_i = \begin{cases} \kappa/(n+\kappa), i = 0 \\ 1/2(n+\kappa), i \neq 0 \end{cases}
\end{aligned} \tag{8}$$

where $(\sqrt{(\mathrm{n}+\kappa)\boldsymbol{P}_{xx}})_i$ is the ith row or column of the matrix square root $(\mathrm{n}+\kappa)\boldsymbol{P}_{xx}$, W_i is the weight of the ith point, and κ scales the third and higher order terms of this set (usually set to 0 or 3-n). In this sampling strategy the weights and the distances from the centre are the same for each sigma point. This means that each sigma point is of the same importance, except for the centre. The set is centrosymmetric and axisymmetric in space.

2.2.2 Minimal skew simplex sampling

The minimal skew sample points are chosen to match the first two moments of $\boldsymbol{x}$ and to minimize the third order moments (see Figure 1b). The sigma points are determined as follows [16]

1) Choose $0 \le W_0 < 1$.

2) Choose weight sequence

$$W_i = \begin{cases} (1-W_0)/2^n, i=1,2 \\ 2^{i-1}W_1, i = 3,\ldots,n+1 \end{cases} \tag{9}$$

3) Initialize vector sequence as

$$\boldsymbol{\chi}_0^1 = [0], \boldsymbol{\chi}_1^1 = [-1/\sqrt{2W_1}], \boldsymbol{\chi}_2^1 = [1/\sqrt{2W_1}] \tag{10}$$

4) Expand vector sequence for $j = 2,\ldots,n$ according to

$$\boldsymbol{\chi}_i^{j+1}=\begin{cases}\begin{bmatrix}\boldsymbol{\chi}_0^j\\0\end{bmatrix}, i=0\\ \begin{bmatrix}\boldsymbol{\chi}_i^j\\-1/\sqrt{2W_{j+1}}\end{bmatrix}, i=1,\ldots,j\\ \begin{bmatrix}0_j\\1/\sqrt{2W_{j+1}}\end{bmatrix}, i=j+1\end{cases} \tag{11}$$

5) Generate sigma points

$$\boldsymbol{X}_j=\bar{\boldsymbol{x}}+\sqrt{\boldsymbol{P}_{xx}}\boldsymbol{\chi}_j^n \tag{12}$$

In this strategy the weights on each sigma point are different, as are the distances from the centre. In the other words, the sigma points are of different importance. The distribution of the sigma points is not centrosymmetric but axisymmetric. For raising state dimension, the weight of the sigma points decreases, and their distance from the centre increases. Compared with those generated directly by higher dimensions, the sigma points generated by the expansion of lower dimensions have larger weights and are closer to the centre.

2.2.3 Spherical simplex sampling

The spherical simplex sigma points consists of n+2 points that lie on a hypersphere centred at the origin (see Figure 1c). The sigma points are determined as follows [17].

1) Choose $0\le W_0<1$.

2) Choose weight sequence

$$W_i=(1-W_0)/(\mathrm{n}+1) \tag{13}$$

3) Initialize vector sequence as

$$\boldsymbol{\chi}_0^1=[0],\boldsymbol{\chi}_1^1=[-1/\sqrt{2W_1}],\boldsymbol{\chi}_2^1=[1/\sqrt{2W_1}] \tag{14}$$

4) Expand vector sequence for $j=2,\ldots,n$ according to

$$\boldsymbol{\chi}_i^j=\begin{cases}\begin{bmatrix}\boldsymbol{\chi}_0^{j-1}\\0\end{bmatrix}, i=0\\ \begin{bmatrix}\boldsymbol{\chi}_i^{j-1}\\-1/\sqrt{j(j+1)W_1}\end{bmatrix}, i=1,\ldots,j\\ \begin{bmatrix}0_{j-1}\\j/\sqrt{j(j+1)W_1}\end{bmatrix}, i=j+1\end{cases} \tag{15}$$

5) Generate sigma points

$$\boldsymbol{X}_j=\bar{\boldsymbol{x}}+\sqrt{\boldsymbol{P}_{xx}}\boldsymbol{\chi}_j^n \tag{16}$$

In this strategy the weights and the distances from the origin of each sigma point are equal. Each sigma point is of the same importance. Note that the distribution of the sigma points is not centrosymmetric.

2.3 Unscented Parameter Estimation

The UPE is a straightforward extension of the UT to the recursive estimation. Parameter estimation, sometimes referred to as system identification, involves determining a nonlinear transformation

$$\boldsymbol{y}_k = G(\boldsymbol{z}_k, \boldsymbol{w}) \tag{17}$$

where $\boldsymbol{z}_k$ is the input, $\boldsymbol{y}_k$ is the output, and $\boldsymbol{w}$ are the parameters. The prediction error of the system is defined as $\boldsymbol{q}_k = \boldsymbol{d}_k - G(\boldsymbol{z}_k, \boldsymbol{w})$, and the goal of learning involves solving for the parameters $\boldsymbol{w}$ in order to minimize the prediction error. An extensive description of UPE can be found in, so just a brief statement is given here.

By writing a new state-space representation, the parameters $\boldsymbol{w}$ can be efficiently estimated

$$\begin{aligned} \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k + \boldsymbol{r}_k \\ \boldsymbol{d}_k &= G(\boldsymbol{z}_k, \boldsymbol{w}_k) + \boldsymbol{e}_k \end{aligned} \tag{18}$$

where $\boldsymbol{w}_k$ corresponds to a stationary process with identity state transition matrix, driven by process noise $\boldsymbol{r}_k$ with covariance $\boldsymbol{R}^r$. The output $\boldsymbol{d}_k$ corresponds to a nonlinear observation on $\boldsymbol{w}_k$, driven by measurement noise $\boldsymbol{e}_k$ with covariance $\boldsymbol{R}^e$. Moreover, Eq. (18) can be transformed further as

$$\begin{aligned} \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k + \boldsymbol{r}_k \\ 0 &= -\boldsymbol{q}_k + \boldsymbol{e}_k \end{aligned} \tag{19}$$

which shows that the prediction error $\boldsymbol{q}_k$ is expected to be zero. Thus the UPE method can be applied directly as an efficient method for learning the parameters. The detailed implementation of UPE is outlined in Figure 2, where $\rho_{RLS} \in (0,1]$ is often referred to as the forgetting factor, which provides for an approximate exponentially decaying weighting on past data.

3. Solving systems of nonlinear equations using UPE

An algorithm using the UPE technique for solving systems of nonlinear equations is proposed. The original problem (18) is converted to a new parameter estimation problem. The estimated parameters $\boldsymbol{w}_k$ are chosen to be $\boldsymbol{x}_k$, no input $\boldsymbol{z}_k$ is foreseen, the output $\boldsymbol{d}_k$ is chosen to be zero, and the nonlinear transformation $\boldsymbol{G}(\cdot)$ is chosen to be the nonlinear function $\boldsymbol{F}(\cdot)$. Thus, the problem can be stated as

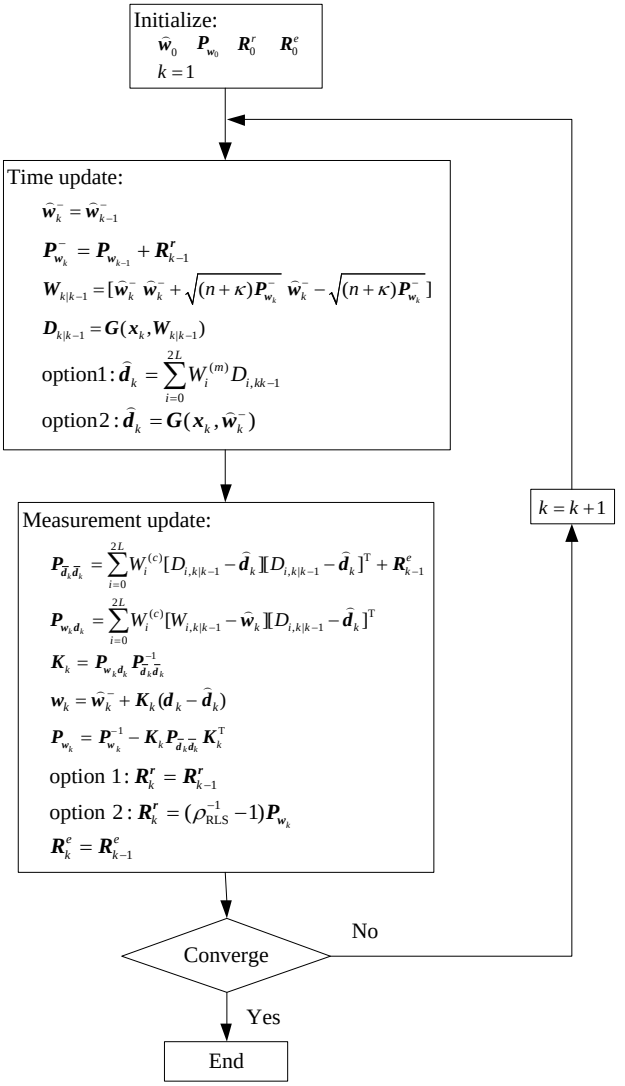

Figure 2. UPE flowchart (symmetrical sampling).

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{x}_k + \boldsymbol{r}_k \\ 0 &= \boldsymbol{F}(\boldsymbol{x}_k) + \boldsymbol{e}_k \end{aligned} \tag{20}$$

where $\boldsymbol{r}_k$ and $\boldsymbol{e}_k$ are process noise and observation noise, with covariance $\boldsymbol{R}^r$ and $\boldsymbol{R}^e$, respectively. Eq. (20) is formally equivalent to Eq. (19), and can be solved through the UPE method shown above. The UPE method has some distinct advantages in solving systems of nonlinear equations. When used to solve systems of nonlinear equations, the UPE method has two main advantages over other methods: 1) Jacobians are evaluated through sigma points propagation, and thus no differentiation is required, 2) no good initial guess is needed.

The computational costs of the UPE method lies in the sampling strategy used: each sample point corresponds to a function evaluation. Symmetric sampling requires $2n+1$ sigma points per iteration, while simplex samplings reduce this number down to $n+2$. In other words, as long as $n > 1$, the simplex samplings overcomes the symmetric sampling in function evaluations per iteration, and thus in overall computational costs. Moreover, the iterations for searching the root may be different when using different

sampling strategies due to iteration routes. Based on the same initial approximation, one sampling strategy may be able to converge to the exact solution, while another sampling strategy may take more iterations or even diverge. Therefore, in order to solve systems of nonlinear equations efficiently and robustly, the focus is to choose the most appropriate sampling strategy of the UT.

In addition to the sampling strategy, there are some adjustable parameters that have influence on the process of searching the root. The covariance $\boldsymbol{P}_w$ determines the spread of sigma points around $\bar{\boldsymbol{x}}$. The larger the covariance is, the more likely the exact solution is to be incorporated in the set of sigma points. However, if the covariance is too large, though the specified information is captured correctly, it does so at the cost of non-local effects. The covariances $\boldsymbol{R}^r$ and $\boldsymbol{R}^e$ are theoretically set to be zero when used for solving systems of nonlinear equations. However, for the sake of numerical stability, these covariances are usually set to small values. The covariance of process noise $\boldsymbol{R}^r$ affects the convergence rate and tracking performance. Roughly speaking, the larger the covariance is, the more quickly the older data is discarded. The covariance $\boldsymbol{R}^e$ determines the final convergence precision of parameter estimation. The less the covariance $\boldsymbol{R}^e$ is, the higher the precision of parameter estimation is. It is important to choose appropriate parameters using UPE for solving systems of nonlinear equations, especially for some highly nonlinear cases. If the parameters are inappropriately chosen, they may cause the process of parameter estimation to take more iterations or even diverge. The selection of optimal adjustable parameters is still under research and is not the focus of this paper.

4. Hybrid UPE algorithm

The UPE algorithm shows some interesting advantages in solving systems of nonlinear equations. However, when the initial approximation is close to the solution, NM is more efficient due to its local fast convergence. The features of both NM and UPE are summarized in Table 1.

Table 1. Advantages and disadvantages of NM and UPE.

Methods	Advantages	Disadvantages
NM	1. Converge quadratically if the initial value is close to the solution. 2. If the Jacobian is provided, only one function evaluation is required per iteration. 3. Reach convergence with high precision.	1. May not converge if the initial value is far from the solution. 2. The Jacobian is required at each iteration. This may be unavailable analytically, singular, or ill-conditioned.
UPE	1. The Jacobian is not required. 2. A good initial value is not required due to its large convergence ability.	1. $2n+1$ or $n+2$ function evaluations are needed at each iteration. 2. A Cholesky factorization is required per iteration to generate sigma points.

To improve the performances of UPE further, a hybrid UPE algorithm (H-UPE) is proposed. This combines the advantages of NM and UPE. In essence, the UPE method is used in the first iterations to take advantage of its larger convergence ability. Then, the NM method is used in the final iterations to take advantage of its local fast convergence and high precision. Considering the different features of the sampling strategies, it is important to choose a proper sampling strategy for the UPE algorithm. The spherical simplex sampling is chosen to be the default sampling strategy. If the spherical simplex sampling fails to converge, the symmetric sampling is used alternatively. The detailed H-UPE algorithm is summarized in Algorithm 1.

Algorithm 1 H-UPE for solving systems of nonlinear equations

Step 1. Define $\boldsymbol{x}_0$, $\boldsymbol{F}(\boldsymbol{x})$, ε_U, ε.

Step 2. Use the UPE algorithm proposed for solving $\boldsymbol{F}(\boldsymbol{x})=0$.

Step 2.1 The spherical simplex sampling is used as default, until $\|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|<\varepsilon_U$ or $\|\boldsymbol{F}(\boldsymbol{x}_k)\|<\varepsilon_U$. If converged, let $\boldsymbol{x}_k$ be the new initial approximation $\boldsymbol{x}_0'$ and go to step 3. If not, turn to step 2.2.

Step 2.2 The symmetric sampling is used alternatively, until $\|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|<\varepsilon_U$ or $\|\boldsymbol{F}(\boldsymbol{x}_k)\|<\varepsilon_U$. If converged, let $\boldsymbol{x}_k$ be the new initial approximation $\boldsymbol{x}_0'$ and go to step 3.

Step 3. Based on the new initial approximation $\boldsymbol{x}_0'$, use NM or other faster methods for solving $\boldsymbol{F}(\boldsymbol{x})=\boldsymbol{0}$, until $\|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|<\varepsilon$ or $\|\boldsymbol{F}(\boldsymbol{x}_k)\|<\varepsilon$.

ε_U is the stopping criterion of UPE, whereas ε is the stopping criterion of the NM (and thus it is the tolerance of the whole method). Note that the main parameter of the H-UPE is ε_U, which acts as a switch. This has a great influence on the convergence and performances of the proposed algorithm. The larger the switching tolerance ε_U is, the earlier the algorithm turns to NM and benefits from fast convergence. However, if ε_U is too large, it may cause the NM to diverge, due to its local convergence. Through a large amount of numerical tests, the switching tolerance is referenced to be $[1,1e2]$ according to diverse nonlinearity of systems of nonlinear equations. ,

5. Numerical examples

5.1 Statement of the problems

To compare the convergence performance of different sampling strategies of the UPE alone, some numerical examples are presented here to illustrate the comparison among the symmetrical sampling (UPE-S), the minimal skew simplex sampling (UPE-MS) and the spherical simplex sampling (UPE-SS). Moreover, to validate the hybrid algorithm proposed above, the same examples are used to compare the performance of H-UPE with existing methods, namely, NM with the finite differential technique (NM), NM with the trust-region dogleg technique (TR) (see [3]), symmetrical rank 1 algorithm (SR1) (see [18, 19]). We consider the following problems for numerical tests. Note that the analytical Jacobian matrixes are not provided for each problem in numerical computations.

Example 1. Consider the system of two equations (see [20]):

$$f_1=(x_1-1)^6-x_2$$
$$f_2=x_2-1$$

One of the exact solutions of the system is $\boldsymbol{r}=[0,1]^{\mathrm{T}}$. The initial approximations chosen are $\boldsymbol{x}_0=[0.1,0.1]^{\mathrm{T}}$ and $\boldsymbol{x}_0=[0.9,0.5]^{\mathrm{T}}$.

Example 2. Consider the system of two equations (see [20]):

$$f_1=x_1+e^{x_2}-\cos(x_2)$$
$$f_2=3x_1-x_2-\sin(x_2)$$

One of the exact solutions of the system is $\boldsymbol{r}=[0,0]^{\mathrm{T}}$. The initial approximations chosen are $\boldsymbol{x}_0=[1,1]^{\mathrm{T}}$ and $\boldsymbol{x}_0=[-1,-3]^{\mathrm{T}}$.

Example 3. Consider the system of two equations (see [21]):

$$f_1=e^{-0.2x_1}-x_2$$
$$f_2=e^{-x_1}-x_2+0.5$$

The exact solutions of the system are $\boldsymbol{r}_1=[1.3127,0.7691]^{\mathrm{T}}$ and $\boldsymbol{r}_2=[2.9836,0.5506]^{\mathrm{T}}$. The initial approximations chosen are $\boldsymbol{x}_0=[10,10]^{\mathrm{T}}$ and $\boldsymbol{x}_0=[100,100]^{\mathrm{T}}$.

Example 4. Consider the system of two equations (see [22]):

$$f_1 = x_1^2 - x_2 + 1$$

$$f_2 = x_1 - \cos(\frac{\pi}{2} x_2)$$

The exact solutions of the system are $\boldsymbol{r}_1 = [0,1]^{\mathrm{T}}$, $\boldsymbol{r}_2 = [-1,2]^{\mathrm{T}}$ and $\boldsymbol{r}_3 = \left[-\sqrt{2}/2, 1.5\right]^{\mathrm{T}}$. The initial approximations chosen are $\boldsymbol{x}_0 = [1,1]^{\mathrm{T}}$ and $\boldsymbol{x}_0 = [4,6]^{\mathrm{T}}$.

Example 5. Consider the system of three equations (see [23]):

$$A = bh - (b-2t)(h-2t)$$

$$I_y = bh^3/12 - (b-2t)(h-2t)^3/12$$

$$I_n = 2(h-t)^2(b-t)^2 t/(h+b-2t)$$

where *h* is the height, *b* is the width, and *t* is the thickness, of a rectangle beam. When $A = 165$, $I_y = 9369$, $I_n = 6835$, one of the exact solutions of the system is $\boldsymbol{r}_1 = [22.8949, 12.5655, 2.7898]^{\mathrm{T}}$. Regardless of physical meaning, another exact solution is $\boldsymbol{r}_2 = [23.2714, 8.9431, 12.9128]^{\mathrm{T}}$. The initial approximations chosen are $\boldsymbol{x}_0 = [30,30,5]^{\mathrm{T}}$ and $\boldsymbol{x}_0 = [10,20,20]^{\mathrm{T}}$.

Example 6. Consider the system of six equations (see [24]):

$$f_1 = x_1 + x_2^2 x_4 x_6/4 + 0.75$$

$$f_2 = x_2 + 0.405 e^{1+x_1 x_2} - 1.405$$

$$f_3 = x_3 - x_4 x_6/2 + 1.5$$

$$f_4 = x_4 - 0.605 e^{1-x_3^2} - 0.395$$

$$f_5 = x_5 - x_2 x_6/2 + 1.5$$

$$f_6 = x_6 - x_1 x_5$$

The solution of the system is $\boldsymbol{r} = [-1,1,-1,1-1,1]^{\mathrm{T}}$. The initial approximations chosen are $\boldsymbol{x}_0 = [0.1,0.1,\ldots,0.1]^{\mathrm{T}}$ and $\boldsymbol{x}_0 = [-1,-1,\ldots,-1]^{\mathrm{T}}$.

Numerical computations have been carried out in MATLAB environment, rounding to 16 significant decimal digits. The stopping criterion used in single performance of the program is (i) $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_\infty < \varepsilon$ (ii) $\|\boldsymbol{F}(\boldsymbol{x}_k)\|_\infty < \varepsilon$, where $\varepsilon = 10^{-15}$, except for example 4 where $\varepsilon = 10^{-11}$. Therefore we check that iterates converge to a limit and moreover that this limit is a solution of the system of nonlinear equations.

In comparing the performance of the methods, we also include the CPU time. The mean CPU time (*t*) is calculated by taking the mean of 100 performances of the program. For the hardware and the software used in the present work, an estimation of the cost of the elementary functions is shown in Table 2, which is measured in microseconds.

Table 2. Estimation of computational cost of elementary functions.

	$x \cdot y$	x/y	$\sqrt{x}$	e^x	$\ln(x)$	$\sin(x)$	$\cos(x)$	$\arccos(x)$	$\arctan(x)$	x^y
t	13.6	13.6	13.1	18.2	20.2	14.8	15.2	17.4	17.4	31.0

*Computed with MATLAB R2011b in a processor Inter (R) Core (TM) i5-2450M CPU @2.5 GHz (64-bit Machine) Microsoft Windows 7, where $x = \sqrt{3} - 1$ and $y = \sqrt{5}$.

As for the detailed settings of the methods, the maximum iteration of each method is set to 1000. For NM, the perturbation for computing Jacobian with finite differences is set to 1e-7. In the UPE and H-UPE, the covariance $\boldsymbol{P}_{\boldsymbol{w}_0}$ is set to $[1e-8,1]\boldsymbol{I}$, $\boldsymbol{R}_0^r$ is set to $1e-10\boldsymbol{I}$ and $\boldsymbol{R}_0^e$ is set to $1e-20\boldsymbol{I}$. In the symmetrical sampling, κ is set to 0. In the minimal skew simplex sampling and the spherical simplex sampling, the weight of the first sigma point is set to 0.5. Moreover, the switching tolerance ε_U of H-UPE is set to 1, except of example 4, in which ε_U is set to 1e2. For each method, the total number of iterations (*k*), the mean CPU time (CPUtime), the total number of function evaluations (TNFE) and the solution finally obtained (Sol.) are analyzed in Table 3-Table 8. The six problems above have been analysed with two different initial guesses each, so in total 12 problems have been faced.

5.2 Results

From numerical results, the comparison among the three sampling strategies shows their different convergence performances. In terms of robustness, UPE-S converges in all of the 12 cases, UPE-SS converges in 10 cases and UPE-MS converges in 5 cases. This proves that UPE-S assures the best convergence stability whereas UPE-MS suffers from the worst numerical stability. In terms of computational costs, UPE-SS requires the least number of iterations and TNFE, while UPE-S involves more iterations and TNFE. This is because, compared with $2n+1$ sigma points needed per iteration in UPE-S, only $n+2$ sigma points are needed in UPE-MS and UPE-SS. Though UPE-S and UPE-SS are comparable in computational time, when coming down to actual complex nonlinear systems, UPE-SS shows its distinct advantage with less function evaluations. Considering these two points here, we can conclude that UPE-SS is the more efficient among sampling strategies, so it is chosen to be the primary sampling strategy in the H-UPE algorithm. However, when UPE-SS fails to converge, UPE-S which is more robust is alternatively used. We also found that the forgetting factor ρ has little influence on the process of searching a root.

Table 3. Performances of the methods in Example 1.

$\boldsymbol{x}_0$	Methods	k	TNFE	CPUtime	Sol.
$[0.1,0.9]^{\mathrm{T}}$	NM	5	18	0.312	$\boldsymbol{r}$
	TR	5	18	4.899	$\boldsymbol{r}$
	SR1	-	-	-	-
	UPE-S	5	26	1.732	$\boldsymbol{r}$
	UPE-MS	5	21	2.307	$\boldsymbol{r}$
	UPE-SS	5	21	1.086	$\boldsymbol{r}$
	H-UPE	5	19	0.502	$\boldsymbol{r}$
$[1,0.5]^{\mathrm{T}}$	NM	-	-	-	-
	TR	-	-	-	-
	SR1	20	21	0.903	$\boldsymbol{r}$
	UPE-S	28	141	12.747	$\boldsymbol{r}$
	UPE-MS	-	-	-	-
	UPE-SS	26	113	8.375	$\boldsymbol{r}$
	H-UPE	26	150	1.570	$\boldsymbol{r}$

Table 4. Performances of the methods in Example 2.

$\boldsymbol{x}_0$	Methods	k	TNFE	CPUtime	Sol.
$[1,1]^{\mathrm{T}}$	NM	6	21	0.460	$\boldsymbol{r}$
	TR	6	21	5.424	$\boldsymbol{r}$
	SR1	40	41	1.413	$\boldsymbol{r}$
	UPE-S	6	31	1.535	$\boldsymbol{r}$
	UPE-MS	36	145	30.754	$\boldsymbol{r}$
	UPE-SS	7	29	1.853	$\boldsymbol{r}$
	H-UPE	6	22	0.565	$\boldsymbol{r}$
$[-1,-3]^{\mathrm{T}}$	NM	242	729	8.579	$\boldsymbol{r}$
	TR	-	-	-	-
	SR1	-	-	-	-
	UPE-S	44	221	13.224	$\boldsymbol{r}$
	UPE-MS	-	-	-	-
	UPE-SS	-	-	-	-
	H-UPE	37	112	1.903	$\boldsymbol{r}$

Compared with the existing methods, UPE-S and UPE-SS show some interesting advantages. For the first initial approximation of each example, when $\boldsymbol{x}_0$ is relatively close to the solution, UPE-S and UPE-SS are comparable or sometimes superior to other methods in terms of the number of iterations and TNFE. However, they take more computational time due to calculation of the Cholesky factorization and the outer products per iteration. For the second initial approximation of each example, when $\boldsymbol{x}_0$ is relatively far from the solution, NM, SR1 and TR generally need more iterations or even fail to converge to the solution, while UPE-S and UPE-SS can still converge to the solution in a few iterations, especially for Example 4 and Example 6. Therefore, it proves that a good initial approximation is not required for the UPE algorithm proposed. Considering the performance of H-UPE, H-UPE is comparable to UPE-S and UPE-SS in the number of iterations and TNFE but takes less computational time benefitting from fast local convergence of NM used alternatively in the final iterations. In general, the computational time of H-UPE is less than 50% of that of UPE-S or UPE-SS. Also, H-UPE was able to converge in all of the 12 cases. Therefore, we can conclude that H-UPE can be used to solve systems of nonlinear equations robustly and efficiently.

Table 5. Performances of the methods in Example 3.

$\boldsymbol{x}_0$	Methods	k	TNFE	CPUtime	Sol.
$[10,10]^{\mathrm{T}}$	NM	12	35	0.628	$\boldsymbol{r}_1$
	TR	12	35	7.379	$\boldsymbol{r}_2$
	SR1	16	17	0.605	$\boldsymbol{r}_1$
	UPE-S	12	61	2.445	$\boldsymbol{r}_1$
	UPE-MS	12	49	5.080	$\boldsymbol{r}_1$
	UPE-SS	12	49	2.289	$\boldsymbol{r}_1$
	H-UPE	12	45	1.365	$\boldsymbol{r}_1$
$[100,100]^{\mathrm{T}}$	NM	-	-	-	-
	TR	15	44	8.500	$\boldsymbol{r}_2$
	SR1	17	18	0.635	$\boldsymbol{r}_1$
	UPE-S	9	46	2.181	$\boldsymbol{r}_1$
	UPE-MS	17	69	7.045	$\boldsymbol{r}_1$
	UPE-SS	12	49	2.375	$\boldsymbol{r}_1$
	H-UPE	11	41	1.173	$\boldsymbol{r}_1$

Table 6. Performances of the methods in Example 4.

$\boldsymbol{x}_0$	Methods	k	TNFE	CPUtime	Sol.
$[1,1]^{\mathrm{T}}$	NM	7	24	0.438	$\boldsymbol{r}_1$
	TR	7	24	6.937	$\boldsymbol{r}_1$
	SR1	-	-	-	-
	UPE-S	7	36	1.526	$\boldsymbol{r}_1$
	UPE-MS	24	97	44.394	$\boldsymbol{r}_1$
	UPE-SS	7	29	2.140	$\boldsymbol{r}_1$
	H-UPE	7	26	0.763	$\boldsymbol{r}_1$
$[4,6]^{\mathrm{T}}$	NM	-	-	-	-
	TR	-	-	-	-
	SR1	-	-	-	-
	UPE-S	10	51	2.998	$\boldsymbol{r}_1$
	UPE-MS	-	-	-	-
	UPE-SS	-	-	-	-
	H-UPE	10	38	1.201	$\boldsymbol{r}_1$

It is also worth noting that in the search for the exact solution, some adjustable parameters have influence on the iterative process of UPE, especially for some highly nonlinear cases. These adjustable parameters include the weighting factor α, the covariance $\boldsymbol{P}_{w0}$, the process noise covariance $\boldsymbol{R}_0^r$, and the measurement noise covariance $\boldsymbol{R}_0^e$. Through numerous numerical examples, we find that the covariances $\boldsymbol{P}_{w0}$ and $\boldsymbol{R}_0^e$ have the most influence on the performance of UPE in solving the systems of nonlinear equations. For Example 3, when the initial approximation $\boldsymbol{x}_0$ is chosen to be $[10,10]^{\mathrm{T}}$ which is close to the exact solution $\boldsymbol{r}_1$, the number of iterations needed to converge with different $\boldsymbol{P}_{w0}$ and $\boldsymbol{R}_0^e$ is summarized and listed in Table 9.

Table 7. Performances of the methods in Example 5.

$\boldsymbol{x}_0$	Methods	k	TNFE	CPUtime	Sol.
$[30,20,5]^{\mathrm{T}}$	NM	6	28	0.570	$\boldsymbol{r}_1$
	TR	8	36	6.423	$\boldsymbol{r}_1$
	SR1	-	-	-	-
	UPE-S	6	43	1.471	$\boldsymbol{r}_1$
	UPE-MS	-	-	-	-
	UPE-SS	6	31	1.425	$\boldsymbol{r}_1$
	H-UPE	6	27	0.712	$\boldsymbol{r}_1$
$[10,20,20]^{\mathrm{T}}$	NM	-	-	-	-
	TR	19	68	10.885	$\boldsymbol{r}_1$
	SR1	-	-	-	-
	UPE-S	36	253	8.434	$\boldsymbol{r}_2$
	UPE-MS	-	-	-	-
	UPE-SS	23	116	5.746	$\boldsymbol{r}_1$
	H-UPE	22	110	4.290	$\boldsymbol{r}_1$

Table 8. Performances of the methods in Example 6.

$\boldsymbol{x}_0$	Methods	k	TNFE	CPUtime	Sol.
$[0.1,\ldots,0.1]^{\mathrm{T}}$	NM	17	126	1.812	$\boldsymbol{r}$
	TR	8	63	7.512	$\boldsymbol{r}$
	SR1	33	34	0.874	$\boldsymbol{r}$
	UPE-S	18	235	5.472	$\boldsymbol{r}$
	UPE-MS	-	-	-	-
	UPE-SS	16	129	8.289	$\boldsymbol{r}$
	H-UPE	15	115	2.008	$\boldsymbol{r}$
$[-1,\ldots,-1]^{\mathrm{T}}$	NM	-	-	-	-
	TR	-	-	-	-
	SR1	-	-	-	-
	UPE-S	21	274	5.971	$\boldsymbol{r}$
	UPE-MS	-	-	-	-
	UPE-SS	16	129	6.101	$\boldsymbol{r}$
	H-UPE	16	122	2.003	$\boldsymbol{r}$

From Table 9, we can see that when the covariance $\boldsymbol{P}_{w0}$ is fixed, the number of iterations decreases as the covariance $\boldsymbol{R}_0^e$ decreases. When the covariance $\boldsymbol{R}_0^e$ is less than $1e-20\boldsymbol{I}$, the minimum number of iterations is obtained. And when the covariance $\boldsymbol{R}_0^e$ is more than $1e-12\boldsymbol{I}$, it even causes the iteration to diverge. Moreover, when the covariance $\boldsymbol{R}_0^e$ is fixed, the number of iterations generally increases as the covariance $\boldsymbol{P}_{w0}$ increases. However, when the covariance $\boldsymbol{P}_{w0}$ is too big, i.e. $1e2\boldsymbol{I}$, it causes the iterations to increase because the property of local sampling is lost. In the UPE algorithm proposed, the covariance $\boldsymbol{P}_{w_0}$ indicates the distribution of the sigma points around the initial approximation $\boldsymbol{x}_0$. The

larger the covariance $\boldsymbol{P}_{w_0}$ is, the more widely the distribution of the sigma points is, and the more likely the true value of the estimated parameters or the solution of the system $\boldsymbol{x}$ may be involved in. Moreover, the covariance $\boldsymbol{R}_0^e$ indicates the distribution of the measurement noise. The smaller the covariance $\boldsymbol{R}_0^e$ is, the less the influence of the measurement noise on the measurement update is, and the higher the precision of parameter estimation is. It is obviously the best when the covariance $\boldsymbol{R}_0^e$ is the smallest. Therefore, we can summarize that, for the root finding of systems of nonlinear equations, the covariance $\boldsymbol{P}_{w_0}$ can be set to be a larger value, and the covariance $\boldsymbol{R}_0^e$ can be set to be a smaller value, especially for the case when the initial approximation is far from the exact solution or the system of nonlinear equations is highly nonlinear. The selection of the optimal parameters is still under research and is not the focus of this paper.

Table 9. TNFE of UPE with different $\boldsymbol{P}_{w0}$ and $\boldsymbol{R}_0^e$ when $\boldsymbol{x}_0 = [10,10]^{\mathrm{T}}$ for Example 3.

$\boldsymbol{R}_0^e$ \ $\boldsymbol{P}_{w0}$	$1e2\boldsymbol{I}$	$1e1\boldsymbol{I}$	$\boldsymbol{I}$	$1e-1\boldsymbol{I}$	$1e-2\boldsymbol{I}$	$1e-4\boldsymbol{I}$	$1e-6\boldsymbol{I}$	$1e-8\boldsymbol{I}$
$1e-11\boldsymbol{I}$	133	217	116	123	132	132	132	232
$1e-12\boldsymbol{I}$	48	70	39	42	47	47	47	44
$1e-13\boldsymbol{I}$	23	25	16	18	22	23	23	22
$1e-14\boldsymbol{I}$	16	13	10	11	16	16	16	16
$1e-15\boldsymbol{I}$	14	9	8	9	14	14	14	14
$1e-16\boldsymbol{I}$	13	7	7	8	13	13	13	13
$1e-17\boldsymbol{I}$	13	7	6	8	12	12	12	12
$1e-18\boldsymbol{I}$	13	6	6	7	12	12	12	12
$1e-20\boldsymbol{I}$	13	6	6	7	12	12	12	12
$1e-22\boldsymbol{I}$	13	6	6	7	12	12	12	12

6. Conclusion

In this paper, an algorithm based on UPE is proposed for solving systems of nonlinear equations, and the comparison of computational costs is proposed among common sampling strategies, namely, the symmetric sampling, the minimal skew simplex sampling and the spherical simplex sampling of UPE. We found that the UPE algorithm overcomes the difficulty of guessing a good initial approximation encountered in NM, and the spherical sampling is the most efficient strategy thanks to its reduced computational cost and its ability to converge in most cases.

To improve the UPE algorithm further, a hybrid algorithm H-UPE is proposed, which combines the UPE and the NM. Through converting the original problem to a new state-space representation, UPE is used in the first iterations to take advantage of its larger convergence ability. Then NM is used in the final iterations to take advantage of its fast local convergence. The H-UPE algorithm is attractive because it has a larger convergence domain than Newton's method and takes less CPU time than the original UPE method. Numerical examples show that the proposed algorithm can be used to solve systems of nonlinear equations robustly and efficiently.

References:

[1]. Kass, R.E., Numerical Methods for Unconstrained Optimization and Nonlinear Equations. 1985, American Statistical Association. 247 - 248.
[2]. Broyden, C.G., A class of methods for solving nonlinear simultaneous equations, Mathematics of computation (1965) 577-593.
[3]. Coleman, T.F., Y. Li, An interior trust region approach for nonlinear minimization subject to bounds, SIAM Journal on Optimization 6 (2) (1996) 418-445.
[4]. Mor E, J.J., D.C. Sorensen, Computing a trust region step, SIAM Journal on Scientific and Statistical

Computing 4 (3) (1983) 553-572.
[5]. Conn, A.R., N.I.M. Gould, P.L. Toint, Trust-region methods, ed. M.S.O. Optimization. Vol. 1. 2000: Society for Industrial and Applied Mathematics. xix, 959.
[6]. Nocedal, J., S.J. Wright, Conjugate gradient methods. 2006: Springer.
[7]. Frontini, M., E. Sormani, Third-order methods from quadrature formulae for solving systems of nonlinear equations, Applied Mathematics and Computation 149 (3) (2004) 771-782.
[8]. Cordero, A., J.R. Torregrosa, Variants of Newton’s Method using fifth-order quadrature formulas, Applied Mathematics and Computation 190 (1) (2007) 686-698.
[9]. Darvishi, M.T., A. Barati, A fourth-order method from quadrature formulae to solve systems of nonlinear equations, Applied Mathematics and Computation 188 (1) (2007) 257-261.
[10]. Darvishi, M.T., A. Barati, A third-order Newton-type method to solve systems of nonlinear equations, Applied Mathematics and Computation 187 (2) (2007) 630-635.
[11]. Cordero, A., E. Martínez, J.R. Torregrosa, Iterative methods of order four and five for systems of nonlinear equations, Journal of Computational and Applied Mathematics 231 (2) (2009) 541-551.
[12]. Wan, E.A., The Unscented Kalman Filter, in Kalman filtering and neural networks. 2001, Interscience.
[13]. Uhlmann, J.K., Simultaneous map building and localization for real time applications. 1994.
[14]. Julier, S.J., J.K. Uhlmann, Unscented Filtering and Nonlinear Estimation, Proceedings of the IEEE 92 (3) (2004) 401-422.
[15]. Julier, S.J., J.K. Uhlmann. A new extension of the Kalman filter to nonlinear systems. 1997: Orlando, FL.
[16]. Julier, S.J., J.K. Uhlmann. Reduced sigma point filters for the propagation of means and covariances through nonlinear transformations. 2002: IEEE.
[17]. Julier, S.J. The spherical simplex unscented transformation. 2003: IEEE.
[18]. Davidon, W.C., Variance algorithm for minimization, The Computer Journal 10 (4) (1968) 406 - 410.
[19]. Broyden, C.G., Quasi-Newton Methods and their Application to Function Minimisation, Mathematics of Computation 21 (99) (1967) 368-381.
[20]. Cordero, A., J.R. Torregrosa, Variants of Newton’s method for functions of several variables, Applied Mathematics and Computation 183 (1) (2006) 199-208.
[21]. Tingzhan, L., L. Wei, H. Ying, Step-adjusting Newton method, Journal of Communication University of China (Science and Techonology) 19 (1) (2012) 8-10. (in Chinese).
[22]. Xiuhua, H., H. Qingwen, A modified Newton's method for solving systems of nonlinear equations. Journal of Henan Normal University (Natural Science), 1998. 26(1) 16-18. (in Chinese).
[23]. Yunkang, S., Z. Wenzhong, Quadratic programming algorithm and applications in solving systems of nonlinear equations, Chinese Journal of Computational Mechanics 19 (2) (2002) 245-246. (in Chinese).
[24]. Krzyworzcka, S., Extension of the Lanczos and CGS methods to systems of nonlinear equations, Journal of computational and applied mathematics 69 (1) (1996) 181-190.