Open Access Library Journal
2019, Volume 6, e5652
ISSN Online: 2333-9721
ISSN Print: 2333-9705

# An Algorithm for the Derivative-Free Unconstrained Optimization Based on a Moving Random Cone Data Set

**Mariam Almadi Mohammed Mu'lla**

Department of Mathematics, University of Kordofan, El-Obeid, North Kordofan, Sudan
Email: marimdx2014@gmail.com

**How to cite this paper:** Mu'lla, M.A.M. (2019) An Algorithm for the Derivative-Free Unconstrained Optimization Based on a Moving Random Cone Data Set. *Open Access Library Journal*, **6**: e5652.
https://doi.org/10.4236/oalib.1105652

**Received:** July 30, 2019
**Accepted:** September 7, 2019
**Published:** September 10, 2019





## Abstract

In this paper, we suggest and analyze some new derivative free iterative methods for solving nonlinear equation $f(x)=0$ using a trust-region method. We also, give several examples to illustrate the efficiency of these methods. Comparison with other similar method is also given. This technique can be used to suggest a wide class of new iterative methods for solving optimization problem. For, solving linearly unconstrained optimization problems without derivatives, a derivative-free Funnel method for unconstrained non-linear optimization is proposed. The study presents new interpolation-based techniques. The main work of this paper depends on some matrix computation techniques. A linear system is solved to obtain the required quadratic model at each iteration. Interpolation points are based on polynomial which is then minimized in a trust-region.

## Subject Areas

Mathematical Analysis, Mathematical Logic and Foundation of Mathematics

## Keywords

Optimization Problem, Convergence, Trust-Region Methods, Model-Based Optimization, Derivative-Free Optimization, Interpolation Examples

## 1. Introduction

The main of this paper is to study methods to solve optimization problem whose objective function does not possess partial derivatives available at hard [1]. This is due to the difficulty of finding derivatives. Derivative-free optimization (DFO) models attempt to express, in mathematical terms, the goal of this paper to solving problems in the "best" way, optimization, in lower dimensional subspaces.

These new methods can be considered to develop this method. The new method starts with an initial data $N=\frac{(n+1)(n+2)}{2}$ points, where, $n$ is the dimension of the problem. An initial guess $x_0$ is provided first, then using random Householder's and Given's, matrices, the $(N-1)$ other points are generated [2] [3], with $x_0$ thought of as a centre. The other points are equidistant from, $x_0$, with distance $\Delta$ a pre-assigned radius. During the iterations there is need to regenerate the interpolation data points. This is done when no progress in function decrease has taken place. The interpolation points are selected to be $D$ units distant from the $k$th iterate of $x_k$. The Householder matrix $H_n=I_n-2uu^{\mathrm{T}}$, $\|u\|_2=1$ is used in the generation of the rest $(N-1)$ points [2] [4]. Given a random vector $z$ where its components are uniformly distributed in $(0,1)$, *i.e.*, $z_i\sim U(0,1)$, $H_n$ is obtained so that $H_nz=\|z\|e_1$. Now if $z$ is a column of $H_n$ then a point $y$ is defined by, $y=x_k+Dz$ [1]. The resulting algorithm is shown to be numerically highly competitive.

## 2. Basic Material

Many interpolation-based trust-region methods construct local polynomial interpolation-based models of the objective function and compute steps by minimizing these models inside a region using the standard trust-region methodology [5] [6] [7]. The models are built so as to interpolate previously computed function values at a subset of past iterates or at specially constructed points. For the model to be well-defined, the interpolation points must be poised [8] [9] [10]. To provide the reader with some necessary information about used notations, we have to recall some basic material about the general trust-region framework, multivariate interpolation, Lagrange polynomials and the definition of poised-ness and well-poised-ness [11] [12].

## 3. Lagrange Polynomials

If the interpolation set $Y$ is poised, the basis of Lagrange polynomials $\{\ell_i(x)\}p_i=1$ exists and is uniquely defined by [11].

### 3.1. Definition

Given a set of interpolation points $Y=\{y^1,y^2,\cdots,y^p\}$ a basis of $p$ polynomials $\ell_j(x),j=1,\cdots,p$ in $p_n^d$ is called a basis of Lagrange polynomials if

$$\ell_j(y^i)=\delta_{ij}=\begin{cases}1 & \text{if } i=j,\\ 0 & \text{if } i\neq j.\end{cases} \tag{1}$$

The unique polynomial $m(x)$ which interpolates $f(x)$ on $Y$ using this basis of Lagrange polynomials can be expressed as

$$m(x)=\sum_{i=1}^{p}f(y^i)\ell_i(x). \tag{2}$$

Moreover, for every poised set $Y=\{y^1,y^2,\cdots,y^p\}$, we have that

$$\sum_{i=1}^{p} \ell_i(x) = 1 \text{ for all } x \in IR^n. \tag{3}$$

The accuracy of $m(x)$ as an approximation of the objective function $f$ in some region $B \subset IR^n$ can be quantified using the following notion [13] [14]. A poised set $Y = \{y^1, y^2, \cdots, y^p\}$ is said to be $\Lambda$-poised in $B$ for some $\Lambda > 0$ if and only if for the basis of Lagrange polynomials associated with $Y$, [15].

$$\Lambda \geq \max_{1 \leq i \leq p} \max_{x \in B} |\ell_i(x)|. \tag{4}$$

The right hand side of (3) is related to the Lebesgue constant $\Lambda_n$ of the set which is defined as

$$\Lambda_n = \max_{x \in B} \sum_{i=1}^{n} |\ell_i(x)|, \tag{5}$$

see for instance [16] [17] [18]. Given the following relations

$$\max_{1 \leq i \leq n} |\ell_i(x)| \leq \sum_{i=1}^{n} |\ell_i(x)| \leq n \max_{1 \leq i \leq n} |\ell_i(x)|, \tag{6}$$

we conclude that

$$\Lambda \leq \Lambda_n \leq n\Lambda. \tag{7}$$

It is a measure of the accuracy of the polynomial interpolation at the set of points below. This suggests to look for a set of interpolation points with a small Lebesgue constant. Hence, conversely, the smaller $\Lambda$, the better the geometry of the set $Y$, importantly for our purposes, Lagrange polynomial values and $\Lambda$-posedness can be used to bound the model function and model gradient error. In particular, it is shown in Ciarlet and Raviart [15], [1] that for any $x$ in the convex hull of $Y$.

### 3.2. Example

Using the natural basis $\bar{\varnothing} = \left\{1, x_1, x_2, \frac{1}{2}x_1^2, \frac{1}{2}x_2, x_1x_2\right\}$ and a sample set $Y = \{y^1, y^2, y^3, y^4\}$, with $y^1 = (0,0)$, $y^2 = (0,1)$, $y^3 = (1,0)$, $y^4 = (1,1)$.

The matrix $M(\varnothing, Y)$

$$M(\varnothing, Y) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0.5 & 0 \\ 1 & 1 & 0 & 0.5 & 0 & 0 \\ 1 & 1 & 1 & 0.5 & 0.5 & 1 \end{bmatrix}.$$

Choosing now the first four columns of $M(\varnothing, Y)$, the system is determined but not well defined since the matrix is singular. We see now that the set $Y$ is not poised with respect to the sub-basis $\varnothing = \left\{1, x_1, x_2, \frac{1}{2}x_1^2\right\}$, but if we selected the sub-basis $\varnothing = \{1, x_1, x_2, x_1x_2\}$, the set $Y$ is well-poised and the corresponding matrix consisting of the first, the second, the third, and the sixth columns of $M(\varnothing, Y)$ is well-conditioned and a unique solution to this determined system exists. is given by [11] [13].

## 4. Unconstrained DFO Algorithm

We consider the bound-constrained optimization problem

$$\min_{x\in IR^n} f(x), \tag{8}$$

where $f$ is a nonlinear function from $IR^n$ into $IR$, which is bounded below, and where $l$ and $u$ are vectors of (possibly infinite) lower and upper bounds on $x$. We denote the feasible domain of this problem by $F$ [6].
a model of the form

$$m_k(x_k, s) = f(x_k) + g_k^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}} H_k s \tag{9}$$

(where $g_k$ and $H_k$ are the function's gradient and Hessian, respectively) is minimized inside a trust region $B_\infty(x_k, \Delta_k)$, as derivatives are not given, $g_k$ and $H_k$ are approximated by determining its coefficients (here represented by the vector, $\alpha$) from the interpolation conditions [14]

$$m(y^i) = \sum_{j=1}^{p} \alpha_j \varnothing_j(y^j) = f(y^i), \ i = 1, \cdots, p. \tag{10}$$

The points $y^1, \cdots, y^p$ considered in the interpolation conditions (10) form the interpolation, set $Y_k$. The set $Y_k$ contains in our case at least $n+1$ points and is chosen as a subset of $X_k$, the set of all points where the value of the objective function $f$ is known. How to choose this interpolation set is of course one of the main issues we have to address below, as not every set $Y_k$ is suitable because of posedness issues [11] [19]. We propose to handle the bound constraints by an "active-set" approach where a bound $l_i$ or $u_i$ is called active at $x$ if $l_i = x_i$ or $x_i = u_i$, respectively. The bound constraints $l_i$ and $u_i$ are inactive if $l_i < x_i < u_i$ at $x$.

### 4.1. The Ingredients Strategy

In a trust-region algorithm for choosing the trust-region radius $\Delta_k$ at each iteration. We base this choice on the agreement between the model function $m_k$ and the objective function $f$ at previous iterations. Given a step $p_k$ we define the ratio:

$$\rho_k = \frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)}, \tag{11}$$

The numerator is actual reduction, and the denominator is the predicted reduction (that is, the reduction in $f$ predicted by the model function). The method has been developed and especially model-based trust-region methods have been shown to perform well. It improves the efficiency of method while maintaining its good theoretical convergence properties, furthermore, the unconstrained method applying a model-based derivative-free method [1]. The interpolation points are chosen randomly, at each iteration, the generation of the random points is descanted in the functions. The function proved to be quite helpful to restore the method on the right track. This is because of randomness. The method proved to be efficient and reliable.

The accuracy of the method is acceptable and relies on the contours of the objective function. The step $p_k$ is obtained by minimizing the model $m_k$ over a region that includes $\rho_k = 0$, the predicted reduction will always be nonnegative. Hence, if $\rho_k$ is negative, the new objective value $f(x_k + p_k)$ is greater than the current value $f(x_k)$, so the step must be rejected. On the other hand, if $\rho_k$ is close to 1, there is good agreement between the model $m_k$ and the function $f$ over this step, so it is safe to expand the trust region for the next iteration. If $\rho_k$ is positive but significantly smaller than 1, do not alter the trust region, but if it is close to zero or negative, we shrink the trust region by reducing $\Delta_k$ at the next iteration [1] [20].

Here $\hat{\Delta}$ (radius) is an overall bound on the step lengths. That the radius is increased only if $\|p_k\|$ actually reaches the boundary of the trust region. If the step stays strictly inside the region, we infer that the current value of $\Delta_k$ is not interfering with the progress of the algorithm, so leave its value unchanged for the next iteration. The shape of the points looks like a moving cone. To turn Algorithm into a practical algorithm, we need to focus on solving the trust-region sub-problem

$$\min_{p\in IR^n} m_k(p) = f_k + g_k^{\mathrm{T}} p + \frac{1}{2} p^{\mathrm{T}} B_k p \tag{12}$$

In discussing this matter, we sometimes drop the iteration subscript $k$ and restate the problem as follows

$$\min_{p\in IR^n} m(p) = f + g^{\mathrm{T}} p + \frac{1}{2} p^{\mathrm{T}} B p$$

A first step to exact solutions is given by the following:

Assume that at the current iterate $x_k$ we have a set of sample points

$$Y = \{y^1, y^2, \cdots y^q\},$$

with $y^i \in IR^n, i = 1, 2, \cdots, q$ again assume that $x_k$ is an element of this set and that no point in $Y$ has a lower function value than $x_k$. This point depends on Householder's and Given's matrices [2] [21]. The choice of the points was guided by a desire to model-based method. Wish to construct a quadratic model of the form [3] [9]

$$m_k(x_k + p) = c + g^{\mathrm{T}} p + \frac{1}{2} p^{\mathrm{T}} G p = f(y) \tag{13}$$

In general $q = \dfrac{(n+1)(n+2)}{2}$ the matrix is given as the follows:

$$\begin{aligned} m(x+p) &= c + \begin{bmatrix} g_1 & \cdots & g_n \end{bmatrix} \begin{bmatrix} p_1 \\ \vdots \\ p_n \end{bmatrix} + \frac{1}{2} \begin{bmatrix} p_1 & \cdots & p_n \end{bmatrix} \begin{bmatrix} g_{11} & \cdots & g_{1n} \\ \vdots & \ddots & \vdots \\ g_{n1} & \cdots & g_{nn} \end{bmatrix} \begin{bmatrix} p_1 \\ \vdots \\ p_n \end{bmatrix} \\ &= \begin{bmatrix} f(y_1) \\ \vdots \\ f(y_n) \end{bmatrix} \end{aligned} \tag{14}$$

$$\begin{bmatrix} 1 & p_1 \cdots p_n & \frac{1}{2}p_1^2 \cdots \frac{1}{2}p_n^2 & p_1p_2\, p_1p_3 \cdots p_1p_n \\ \vdots & \vdots & \vdots & \vdots \\ 1 & & & \end{bmatrix} \begin{bmatrix} c \\ g_1 \\ g_2 \\ \vdots \\ g_n \\ G_{11} \\ \vdots \\ G_{1n} \end{bmatrix} = \begin{bmatrix} f(y_1) \\ \vdots \\ f(y_n) \end{bmatrix}$$

**Algorithm (4.1)**

1) Input the $Y = \left[{}^1y, {}^2y, \cdots, {}^Ny\right]$,

then set $N = \frac{1}{2}(n+1)(n+2)$ size of $Y$.

2) Solve the model (3) to find $c, g, G$.

$$c = h_1;$$

$$g = h_{(2:n+1)};$$

*for* $i = 1, 2, \cdots, n$

The matrix $G_{ii} = h_{n+1+i}$;

*end*

$$m = 2n+1;$$

1) *for* $i = 1, 2, \cdots, n-1$

$$ii = m + \frac{(i-1)(2n-i)}{2};$$

2) *for* $j = i+1, \cdots, n$

$$G_{(i,j)} = h_{(ii+j-i)};$$

$$G_{(j,i)} = G_{(i,j)};$$

*end*

Output $c, g, G$

### 4.2. Theorem (First Order Necessary Conditions)

If $x^*$ is a local minimize and $f$ is continuously differentiable in an open neighborhood of $x^*$, then $\nabla f(x^*) = 0$ [1] [22].

### 4.3. Theorem (Second Order Necessary Conditions)

If $x^*$ is a local minimize of $f$ and $\nabla^2 f$ exists and is continuous in an open neighborhood of $x^*$, then $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive semi definite [10] [11] [18].

### 4.4. Theorem (Second Order Sufficient Conditions)

Suppose that $\nabla^2 f$ is continuous in an open neighborhood of $x^*$ and that $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite. Then $x^*$ is a strict local minimize of $f$[1].

### 4.5. Theorem

Let $f$ be twice Lipschitz continuously differentiable in a neighborhood of a point $x^*$ at which second-order sufficient conditions Theorem 4.4 are satisfied. Suppose the sequence $\{x_k\}$ converges to $x^*$ and that for all $k$ sufficiently large, the trust-region algorithm based on $B_k = \nabla^2 f(x_k)$ chooses steps $p_k$ that satisfy the Cauchy-point-based model reduction criterion and are asymptotically similar to Newton steps $p_k^N$ whenever $\|p_k^N\| \le \frac{1}{2}\Delta_k$, that is [23],

$$\|p_k - p_k^N\| = o(\|p_k^N\|). \tag{11}$$

Then the trust-region bound $\Delta_k$ becomes inactive for all $k$ sufficiently large and the sequence $\{x_k\}$ converges super-linearly to $x^*$ [20] [23].

## 5. The Selected Test Problems

The following functions are used in testing method have been selected from [22] and [24]

1) $f(x) = 100\left(x_2 - x_1^2\right)^2 + \left(1 - x_1^2\right)$

2) $f(x) = (x_1 - x_3)^4 + 7(x_2 - 1)^2 + 10(x_3 - 10)^4$

### The Output (Tables 1-12)

**Table 1.** Summary of structural iterative solution overview of (DFO) of the function $f(x) = 100\left(x_2 - x_1^2\right)^2 + \left(1 - x_1^2\right)$ the initial point is $x_0 = [1, 1.6, 1]^{\mathrm{T}}$.

| $xx =$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1.5129 | 1.5129 | 1.5129 | 1.5146 | 1.5160 | 1.5160 | 1.5173 | 1.5178 | 1.5187 | 1.5187 |
| 2.3718 | 2.3718 | 2.3718 | 2.3711 | 2.3701 | 2.3701 | 2.3709 | 2.3705 | 2.3702 | 2.3702 |
| 4.9805 | 4.9805 | 4.9805 | 4.9813 | 4.9807 | 4.9807 | 4.9790 | 4.9790 | 4.9791 | 4.9791 |

**Table 2.** The value of the function $f(x) = 100\left(x_2 - x_1^2\right)^2 + \left(1 - x_1^2\right)$ at each iteration in **Table 1**.

| $f_x =$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| −0.6015 | −0.6015 | −0.6015 | −0.6977 | −0.7813 | −0.7813 | −0.8325 | −0.8578 | −0.9005 | −0.9005 |

**Table 3.** Summary of structural iterative Solution overview of (DFO) of the function $f(x) = 100\left(x_2 - x_1^2\right)^2 + \left(1 - x_1^2\right)$ using the last point in the iterative in **Table 2** $x_1 = [1.5187; 2.3702; 4.9791]^{\mathrm{T}}$.

| $xx =$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1.5194 | 1.5194 | 1.5194 | 1.5194 | 1.5194 | 1.5194 | 1.5194 | 1.5194 | 1.5194 | 1.5194 |
| 2.3689 | 2.3689 | 2.3689 | 2.3689 | 2.3689 | 2.3689 | 2.3689 | 2.3689 | 2.3689 | 2.3689 |
| 4.9789 | 4.9789 | 4.9789 | 4.9789 | 4.9789 | 4.9789 | 4.9789 | 4.9789 | 4.9789 | 4.9789 |

**Table 4.** The value of the function $f(x)=100\left(x_2-x_1^2\right)^2+\left(1-x_1^2\right)$ at each iteration in Table 3.

| $f_x=$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| −0.9432 | −0.9443 | −0.9443 | −0.9443 | −0.9443 | −0.9443 | −0.9443 | −0.9452 | −0.9452 | −0.9452 |

**Table 5.** Summary of structural iterative Solution overview of (DFO) of the function $f(x)=100\left(x_2-x_1^2\right)^2+\left(1-x_1^2\right)$ using the last point in the iterative in Table 4 $x_2=\left[1.5194;2.3689;4.9789\right]^{\mathrm{T}}$.

| $xx=$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1.5281 | 1.5281 | 1.5281 | 1.5284 | 1.5285 | 1.5285 | 1.5285 | 1.5285 | 1.5285 | 1.5285 |
| 2.3693 | 2.3693 | 2.3693 | 2.3690 | 2.3690 | 2.3690 | 2.3690 | 2.3690 | 2.3690 | 2.3690 |
| 4.9708 | 4.9708 | 4.9708 | 4.9705 | 4.9704 | 4.9704 | 4.9704 | 4.9704 | 4.9704 | 4.9704 |

**Table 6.** The value of the function $f(x)=100\left(x_2-x_1^2\right)^2+\left(1-x_1^2\right)$ at each iteration in Table 5.

| $f_x=$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| −1.2180 | −1.2180 | −1.2180 | −1.2264 | −1.2299 | −1.2299 | −1.2299 | −1.2299 | −1.2299 | −1.2299 |

**Table 7.** Summary of structural iterative Solution overview of (DFO) of the function $f(x)=\left(x_1-x_3\right)^4+7\left(x_2-1\right)^2+10\left(x_3-10\right)^4$ using the initial point in Table 6 $x_0=\left[0;0;0;0\right]^{\mathrm{T}}$.

| $xx=$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| −0.0964 | −0.4549 | −1.0881 | 2.3169 | 2.3169 | 2.3290 | 2.4203 | 2.4203 | 2.1784 | 2.1784 |
| −0.7355 | −2.3894 | −2.1112 | −0.7489 | −0.7475 | −0.7235 | −0.4007 | −0.4006 | 0.5426 | 0.5427 |
| 0.9289 | 3.8014 | 8.0268 | 9.3400 | 9.3408 | 9.3526 | 9.3281 | 9.3281 | 9.4005 | 9.4005 |
| 3.7442 | 4.0715 | 4.5942 | 1.3074 | 1.3083 | 1.2972 | 1.1319 | 1.1318 | 0.3410 | 0.3411 |

**Table 8.** The value of the function $f(x)=\left(x_1-x_3\right)^4+7\left(x_2-1\right)^2+10\left(x_3-10\right)^4$ at each iteration in Table 7.

| $f_x=$ 1.0e + 004 | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 6.7821 | 1.4986 | 0.0499 | 0.0024 | 0.0023 | 0.0023 | 0.0016 | 0.0016 | 0.0003 | 0.0003 |

**Table 9.** Summary of structural iterative Solution overview of (DFO) of the function $f(x)=\left(x_1-x_3\right)^4+7\left(x_2-1\right)^2+10\left(x_3-10\right)^4$ using the last point in the iterative in Table 8 $x_1=\left[2.1784;0.5427;9.4005;0.3411\right]^{\mathrm{T}}$.

| $xx=$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 2.1816 | 2.1815 | 2.1815 | 2.1815 | 2.1623 | 2.1624 | 2.1624 | 2.1626 | 2.1737 | 2.1743 |
| 0.5507 | 0.5522 | 0.5508 | 0.5508 | 0.6030 | 0.6030 | 0.6031 | 0.6038 | 0.6257 | 0.6608 |
| 9.3986 | 9.4015 | 9.4050 | 9.4050 | 9.4486 | 9.4486 | 9.4486 | 9.4495 | 9.4679 | 9.5020 |
| 0.3479 | 0.3498 | 0.3511 | 0.3511 | 0.4060 | 0.4060 | 0.4060 | 0.4052 | 0.4156 | 0.4375 |

**Table 10.** The value of the function $f(x)=(x_1-x_3)^4+7(x_2-1)^2+10(x_3-10)^4$ at each iteration in **Table 9**.

| $f_x=$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 3.1698 | 3.1352 | 3.1143 | 3.1141 | 2.5203 | 2.5198 | 2.5197 | 2.5090 | 2.2481 | 1.8850 |

**Table 11.** Summary of structural iterative Solution overview of (DFO) of the function $f(x)=(x_1-x_3)^4+7(x_2-1)^2+10(x_3-10)^4$ using the last point in the iterative in **Table 10** $x_2=[2.1743;0.6608;9.5020;0.4375]^T$.

| $xx=$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 2.1799 | 2.1799 | 2.1799 | 2.1843 | 2.1843 | 2.1850 | 2.1850 | 2.1851 | 2.1850 | 2.1850 |
| 0.6730 | 0.6730 | 0.6730 | 0.6900 | 0.6900 | 0.6912 | 0.6943 | 0.6943 | 0.6966 | 0.6966 |
| 9.5097 | 9.5097 | 9.5097 | 9.5167 | 9.5167 | 9.5174 | 9.5197 | 9.5198 | 9.5219 | 9.5219 |
| 0.4306 | 0.4306 | 0.4306 | 0.4480 | 0.4480 | 0.4481 | 0.4479 | 0.4479 | 0.4484 | 0.4484 |

**Table 12.** The value of the function $f(x)=(x_1-x_3)^4+7(x_2-1)^2+10(x_3-10)^4$ at each iteration in **Table 11**.

| $f_x=$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1.7791 | 1.7790 | 1.7789 | 1.6613 | 1.6613 | 1.6512 | 1.6273 | 1.6270 | 1.6083 | 1.6082 |

## 6. Conclusion

In this paper it has been shown that using a randomly selected data set point, within an interpolation based method for derivative free optimization was adequate and practical. In the generating of these randomly selected points Householders and Given's matrices were used. The randomness comes from the random seed vectors, which were then transformed by Householders and Given's matrices into the required.

## Acknowledgements

I would like to thank my supervisor, Dr. Muhsin Hassan Abdallah who was a great help to me.

## Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

## References

[1] Nocedal, J. and Wright, S.J. (2000) Numerical Optimization Mathematics Subject Classification. 2nd Edition, Library of Congress Control Number: 2006923897. https://doi.org/10.1007/b98874

[2] Wilkinson, J.H. (1960) Householder's Method for the Solution of the Algebraic Eigenproblem. *The Computer Journal*, **3**, 23-27. https://doi.org/10.1093/comjnl/3.1.23

[3] Powell, M.J.D. (2014) Powell. LINCOA software. http://mat.uc.pt/zhang/software.htmllincoa

[4] Powell, M.J.D. (1970) A Hybrid Method for Nonlinear Equations. In: Robinowitz, P., Ed., *Numerical Methods for Nonlinear Algebraic Equations*, Gordon and Breach Science, London, 87-144.

[5] Conn, A.R., Gould, N.I.M. and Toint, P.L. (2000) Trust-Region Methods. MPS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, Philadelphia, PA. https://doi.org/10.1137/1.9780898719857

[6] Dennis, J.E. and Schnabel, A.B. (1989) A View of Unconstrained Optimization. In: *Handbooks in Operations Research and Management*, Elsevier Science Publishers, Amsterdam, The Netherlands, 1-72. https://doi.org/10.1016/S0927-0507(89)01002-9

[7] Golub, G.H. and von Matt, U. (1991) Quadratically Constrained Least Squares and Quadratic Problems. *Numerische Mathematik*, **59**, 561-580. https://doi.org/10.1007/BF01385796

[8] Conn, A.R., Scheinberg, K. and Vicente, L.N. (2008) Geometry of Interpolation Sets in Derivative Free Optimization. *Mathematical Programming*, **111**, 141-172. https://doi.org/10.1007/s10107-006-0073-5

[9] Powell, M.J.D. (1998) Direct Search Algorithms for Optimization Calculations. *Acta Numerica*, **7**, 287-336. https://doi.org/10.1017/S0962492900002841

[10] Fletcher, R. (1987) Practical Methods of Optimization. 2nd Edition, John Wiley and Sons, Chichester.

[11] Gratton, S., Toint, P.L. and Troltzsch, A. (2011) An Active-Set Trust-Region Method for Derivative-Free Nonlinear Bound-Constrained Optimization. *Optimization Methods and Software*, **26**, 873-894.

[12] Powell, M.J.D. (1970) A New Algorithm for Unconstrained Optimization. In: Rosen, J.B., Mangasarian, O.L. and Ritter, K., Eds., *Nonlinear Programming*, Academic Press, New York, 31-65. https://doi.org/10.1016/B978-0-12-597050-1.50006-3

[13] Gay, D.M. (1981) Computing Optimal Local Constrained Step. *SIAM Journal on Scientific and Statistical Computing*, **2**, 186-197. https://doi.org/10.1137/0902016

[14] Moré, J.J. and Sorenson, D.C. (1983) Computing a Trust Region Step. *SIAM Journal on Scientific and Statistical Computing*, **4**, 553-572. https://doi.org/10.1137/0904038

[15] Ciarlet, P.G. and Raviart, P.A. (1972) General Lagrange and Hermite Interpolation in $IR^n$ with Applications to Finite Element Methods. *Archive for Rational Mechanics and Analysis*, **46**, 177-199. https://doi.org/10.1007/BF00252458

[16] Erdős, P. (1961) Problems and Results on the Theory of Interpolation. II. *Acta Mathematica Academiae Scientiarum Hungarica*, **12**, 235-244. https://doi.org/10.1007/BF02066686

[17] Smith, S.J. (2006) Lebesgue Constants in Polynomial Interpolation. *Annales Mathematicae et Informaticae*, **33**, 109-123.

[18] Byrd, R.H., Schnabel, R.B. and Schultz, A.A. (1988) Approximate Solution of the Trust Regions Problem by Minimization over Two-Dimensional Subspaces. *Mathematical Programming*, **40**, 247-263. https://doi.org/10.1007/BF01580735

[19] Paviani, D.A. and Himmelblau, D.M. (1969) Constrained Nonlinear Optimization by Heuristic Programming. *Operations Research*, **17**, 872-882. https://doi.org/10.1287/opre.17.5.872

[20] Conn, A.R., Scheinberg, K. and Vicente, L.N. (2008) Introduction to Derivative-Free Optimization. MPS-SIAM Series on Optimization. SIAM, Philadelphia,

PA. https://doi.org/10.1137/1.9780898718768

[21] Yuan, Y.-X. (2015) A Review of Trust Region Algorithms for Optimization. Chinese Academy of Sciences, Beijing.

[22] Powell, M.J.D. (1998) Direct Search Algorithms for Optimization Calculations. *Acta Numerica*, 7, 287-336. https://doi.org/10.1017/S0962492900002841

[23] Dennis, J.E. and Mei, H.H.W. (1979) Two New Unconstrained Optimization Algorithms Which Use Function and Gradient Values. *Journal of Optimization Theory and Applications*, **28**, 453-482. https://doi.org/10.1007/BF00932218

[24] Omojokun, E.O. (1989) Trust Region Algorithms for Optimization with Nonlinear Equality and Inequality Constraints. Ph.D. Thesis, University of Colorado, Boulder, CO.