A DERIVATIVE-FREE ALGORITHM FOR THE LEAST-SQUARE MINIMIZATION

HONGCHAO ZHANG*, ANDREW R. CONN †, AND KATYA SCHEINBERG ‡

Abstract. We develop a framework for a class of derivative-free algorithms for the least-squares minimization problem. These algorithms are based on polynomial interpolation models and are designed to take advantages of the problem structure. Under suitable conditions, we establish the global convergence and local quadratic convergence properties of these algorithms. Promising numerical results indicate the algorithm is efficient and robust when finding both low accuracy and high accuracy solutions. Comparisons are made with standard derivative-free software packages that do not exploit the special structure of the least-squares problem or that use finite differences to approximate the gradients.

Key words. Derivative-Free Optimization, Least-squares, Trust Region, Levenberg-Marquardt Method, System of Nonlinear Equations, Global Convergence, Local Convergence

AMS subject classifications. 65K05, 90C30, 90C56

1. Introduction.

In this paper, we design a class of derivative-free optimization (DFO) algorithms for the following least-squares problem:

$$\min\ \Phi(\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{m} f_i^2(\mathbf{x}) = \frac{1}{2}\|F(\mathbf{x})\|^2, \tag{1.1}$$

where our norm is the standard Euclidian norm, $F(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x}))^\top$ and $f_i(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, m$, are general nonlinear twice continuously differentiable functions, but none of their first order or second order derivatives are explicitly available. These least-squares optimization problems arise frequently in computational science, engineering and industry. For example, it is the most common approach for solving a system of nonlinear equations. However, because the function values are often extracted from economic models, physical or chemical experiments or measurements, or require expensive and complex computer simulations (for example, Monte Carlo simulations or simulations based on solving differential equations), in practice it is often impossible or impractical (say, in the case of legacy codes) to obtain their derivatives. Consequently, it is often desirable to treat these functions as black boxes, necessitating that derivative-free algorithms be used to solve (1.1). The goal of these algorithms is to find the primal variables which achieve the minimum sum of the squares of this system of nonlinear functions with the least possible number of function evaluations. Not infrequently, constrained versions of this problem are also of interest, although we will not consider such extensions here, except for some brief comments.

There are mainly three classes of derivative-free optimization methods in the literature. The first is a class of direct search methods that explore the variable space by sampling points from a predefined class of geometric patterns or involve some random process [1, 2]. These methods frequently do not assume smoothness of the

*hozhang@math.lsu.edu, http://www.math.ufl.edu/∼hozhang, Department of Mathematics, Louisiana State University, 140 Lockett Hall, Baton Rouge, LA 70803-4918, USA.

†arconn@us.ibm.com, Department of Mathematical Sciences, IBM T.J. Watson Research Center, Route 134, P.O. Box 218, Yorktown Heights, New York 10598, USA.

‡katyascheinberg@gmail.com, Dept of IEOR, Columbia University, Mudd, 500 W 120st, New York, NY 10027, USA.

objective function and therefore can be applied to a broad class of problems. But, on the other hand, a relatively large number of function evaluations are often required. The second class of methods combine finite difference techniques with quasi-Newton methods, for instance [18]. These methods are theoretically simple and relatively easy to implement. However, because they use finite differences to approximate derivatives, they are not always robust, especially in the presence of noise, which is often the case for derivative-free applications. The third class of methods is based on the sequential minimizations of models, typically quadratic or linear, based upon evaluations of the objective function at sample sets [4, 17, 21, 22]. Some recent research, [19], indicates the difference between trust-region model-based methods and other approaches, with the former being apparently superior, even for noisy and piecewise-smooth problems. Nevertheless, the latter are still widely used in the engineering community.

The algorithm described in this paper belongs to the third class of methods, which is based on modeling the objective function by multivariate interpolation in combination with trust region techniques. But compared with other model-based methods in the literature, we take full advantages of the least-squares problem structure as well as adaptively maintaining the robustness of trust-region methods. It is well-known that in model-based derivative-free optimization, whenever the optimization suggests that a better point is unavailable in the region of interest it is important to ensure that the model is sufficiently good, which in practise usually means that one needs to ensure that the local geometry of the sampling set, on which the models depend, is appropriate. We know that, in the absence of any special structure (for example a diagonal Hessian for the true function) $(n+1)(n+2)/2$ sampling points are required to build a fully quadratic interpolation model. However, this computing cost is unacceptable for many iterations when n is large unless (which is not typical) the function evaluations are inexpensive. Consequently our approach will leverage Powell's least Frobenius norm updating strategy [22] to asymptotically build at least fully-linear models for each of the separate nonlinear functions in the sum of squares. Thereby exploiting structure whilst maintaining relatively modest computational costs. More specifically, as suggested in [22], in our numerical experiments we only use $2n+1$ (significantly less than $(n+1)(n+2)/2$ for large n) sampling points for each model in the trust-region subproblem. Since the same $2n+1$ interpolation points are used for each function, the computational cost for building these models at each iteration is only of the order of n^2, except whenever the occasional iteration requires a shift of the model origin. To achieve this goal whilst retaining global convergence and fast local convergence, care needs to be taken in the trust region management and the choice of the sample points for the model. Powell chooses to do this by maintaining a separate neighbourhood for each purpose. However, it is possible to adaptively update a single trust-region radius so as to serve both purposes simultaneously and rigorously prove the global convergence. We propose a generalised Levenberg-Marquardt algorithm combined with trust region techniques to minimize the least-squares residue of the problem. Similar to the general approach proposed in [4], under suitable conditions, we establish the global convergence of our method for structured least-squares minimization using a trust region framework. Local quadratic convergence is also established under some local error bound conditions [25, 27]. These conditions are considerably weaker than a non-singularity assumption on the Jacobian.

We agree with the point of view taken by Moré and Wild that in derivative-free optimization, users are often interested (not necessarily simultaneously) in both low accuracy and high accuracy solutions [19]. Hence it is useful to test any new derivative-free solver to investigate its shorter-term (i.e. low accuracy) and longer-term (i.e. closer to asymptotic and, hopefully, higher accuracy) behaviour on a suitable collection of benchmark problems, which should include both problems with and without noise. For long-term behavior, we essentially assume there is an unlimited computational budget and we let the solver find the best possible solution. But in many practical situations, a low accuracy solution may be all that is required or that can be obtained within a given computational budget. For short-term behavior testing, we want to know, at least approximately, the average ratio of the obtained function reduction as a function of the number of function evaluations. In Section 5, below, we test both the short-term and long-term behaviour of our algorithms on the benchmark problems developed in [16, 19].

The paper is organized as follows. In Section 2 we introduce our derivative-free algorithms for least-squares minimization (DFLS) along with some preliminary definitions and lemmata. The global convergence of the class of algorithms is established in Section 3, while local quadratic convergence is established in Section 4. The numerical performances of our algorithms for both the short-term and long-term behaviour compared with other derivative-free software packages are provided in Section 5. Finally, some conclusions are made in Section 6.

Notation. Unless otherwise specified our norm $\|\cdot\|$ is the standard Euclidean norm. Given any set $\mathbf{S}$, $|\mathbf{S}|$ denotes the cardinality of $\mathbf{S}$. We let $\mathcal{B}$ denote a closed ball in $\mathbb{R}^n$ and $\mathcal{B}(\mathbf{z}, \Delta)$ denote the closed ball centered at z, with radius $\Delta > 0$. By $\mathcal{P}_n^d$ we mean the space of all polynomials of degree $\leq d$ in $\mathbb{R}^n$

There are several constants used in this paper which are denoted by κ (sometimes with acronyms for the subscripts that are meant to be helpful). We collected their definition in this subsection, for convenience. The actual meaning of the constants will become clear when each of them is introduced in the paper.

$\hat{\kappa}_{ef}$, κ_{ef}	'error in the function value'
$\hat{\kappa}_{eg}$, κ_{eg}	'error in the gradient'
κ_{eH}	'error in the Hessian'
κ_H^1 , κ_H^2 and κ_H^3	'associated with the definition of the model Hessian'
κ_m	'bound on all the separate models'
κ_g	'bound on the gradient of all the separate models'
κ_H	'bound on the Hessian of all the separate models'
κ_H^ϕ	'bound on the (modified) Hessian of ϕ'

2. The Algorithm.

We first slightly extend the definition of a Λ–poised set $\mathbf{Y} \subset \mathbb{R}^n$ in [6] as follows:

Definition 2.1. *Let $\Lambda > 0$ and $\mathcal{P}$ be a space of polynomials on $\mathbb{R}^n$ with a basis $\varphi = \{\psi_0(\mathbf{x}), \psi_1(\mathbf{x}), \ldots, \psi_p(\mathbf{x})\}$. Then, a set $\mathbf{Y} = \{\mathbf{y}^0, \mathbf{y}^1, \ldots, \mathbf{y}^p\}$ is said to be Λ–poised in $\mathcal{B}$ for $\mathcal{P}$ (in the interpolation sense) with respect to the basis φ if and only if for any $\mathbf{x} \in \mathcal{B} \subset \mathbb{R}^n$ there exists $\lambda(\mathbf{x}) \in \mathbb{R}^{p+1}$ such that:*

$$\sum_{i=0}^{p} \lambda_i(\mathbf{x}) \varphi(\mathbf{y}^i) \;=\; \varphi(\mathbf{x}) \qquad \textit{with} \qquad \|\lambda(\mathbf{x})\|_\infty \;\leq\; \Lambda.$$

Other equivalent definitions of a Λ–poised set given in Chapter 3 of [6] can be similarly extended. Note that unfortunately, in the underdetermined case the bound depends on the basis used and so we need to tie our definition to a particular choice of basis. In fact we state it in terms of the natural basis and relate it to the linear case so that some basis independent statements can be made (see Theorem 5.8 in [7]).

Throughout the paper, we will also require the following assumption.

ASSUMPTION 2.1. *Assume we are given any set $\mathbf{Y} \subset \mathcal{B}(\mathbf{z}, \Delta)$ with $n+1 \leq |\mathbf{Y}| \leq (n+2)(n+1)/2$ and $\mathbf{z} \in \mathbb{R}^n$. Then we can apply a finite number of substitutions of the points in $\mathbf{Y}$, in fact, at most $|\mathbf{Y}|-1$, such that the new resultant set is Λ–poised in $\mathcal{B}(\mathbf{z}, \Delta)$ for a polynomial space $\mathcal{P}$, with dimension $|\mathbf{Y}|$ and $\mathcal{P}_n^1 \subseteq \mathcal{P} \subseteq \mathcal{P}_n^2$.*

To see that this is not unduly restrictive and numerical procedures can be applied for achieving this, the reader is referred to [5] and [6] for examples. In the following we call a given set $\mathbf{Y} \subset \mathcal{B}(\mathbf{z}, \Delta)$ with $n+1 \leq |\mathbf{Y}| \leq (n+1)(n+2)/2$, Λ–poised whenever it is Λ–poised in $\mathcal{B}(\mathbf{z}, \Delta)$ for some polynomial space $\mathcal{P}$ with dimension $|\mathbf{Y}|$ on the natural basis of $\mathcal{P}$, where for any polynomial space $\mathcal{P}$ between $\mathcal{P}_n^1$ and $\mathcal{P}_n^2$, we define its natural basis to be the basis of $\mathcal{P}$ which includes the natural basis of $\mathcal{P}_n^1$ and is a subset of the natural basis of of $\mathcal{P}_n^2$.

The proof of the following lemma is essentially that of Theorem 5.4 in [6] but there are slight differences and we felt it was worthwhile to provide an abbreviated proof here for completeness (see the cited proofs for more details).

LEMMA 2.2. *Given any $\Delta > 0$, $\mathbf{z} \in \mathbb{R}^n$ and $\mathbf{Y} = \{\mathbf{0}, \mathbf{y}^1, \ldots, \mathbf{y}^p\} \subset \mathcal{B}(\mathbf{z}, \Delta)$ Λ–poised in $\mathcal{B}(\mathbf{z}, \Delta)$ with $n+1 \leq |\mathbf{Y}| \leq (n+1)(n+2)/2$, let $\mathbf{p}(\cdot) \in \mathcal{P}_n^2$ be a interpolating polynomial of f on $\mathbf{Y}$, i.e.*

$$\mathbf{p}(\mathbf{y}^i) = f(\mathbf{y}^i),\ i = 1, \ldots, |\mathbf{Y}|.$$

If $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and ∇f is Lipschitz continuous with constant L_1 in an open set containing $\mathcal{B}(\mathbf{z}, \Delta)$, then for any $\mathbf{s} \in \mathcal{B}(\mathbf{0}, \Delta)$ we have

$$\begin{aligned} \|\nabla f(\mathbf{z}+\mathbf{s}) - \nabla \mathbf{p}(\mathbf{z}+\mathbf{s})\| &\leq \hat{\kappa}_{eg}(n, \Lambda, L_1)\ (\|\nabla^2 \mathbf{p}\| + 1)\ \Delta, \\ |f(\mathbf{z}+\mathbf{s}) - \mathbf{p}(\mathbf{z}+\mathbf{s})| &\leq \hat{\kappa}_{ef}(n, \Lambda, L_1)\ (\|\nabla^2 \mathbf{p}\| + 1)\ \Delta^2, \end{aligned}$$

where $\hat{\kappa}_{eg}$ and $\hat{\kappa}_{ef}$ are positive constants depending only on n, Λ and L_1.

Proof. The proof is similar to that of Theorems 4.1 and 4.2 (estimates in the linear and quadratic cases, respectively) in [5].

By our definition, Y is Λ–poised in $\mathcal{B}(\Delta)$ for some polynomial space $\mathcal{P}$ with dimension $|\mathbf{Y}|$ and $\mathcal{P}_n^1 \subseteq \mathcal{P} \subseteq \mathcal{P}_n^2$. Hence, let $\{1, x_1, x_2, \ldots, x_n, \phi_{n+1}(\mathbf{x}), \ldots, \phi_p(\mathbf{x})\}$ be the natural bases of $\mathcal{P}$ and

$$\mathbf{M}_{p\times p} = \begin{bmatrix} y_1^1 & \cdots & y_n^1 & \phi_{n+1}(\mathbf{y}^1) & \cdots & \phi_p(\mathbf{y}^1) \\ y_1^2 & \cdots & y_n^2 & \phi_{n+1}(\mathbf{y}^2) & \cdots & \phi_p(\mathbf{y}^2) \\ \vdots & & \vdots & \vdots & & \vdots \\ y_1^p & \cdots & y_n^p & \phi_{n+1}(\mathbf{y}^p) & \cdots & \phi_p(\mathbf{y}^p) \end{bmatrix},$$

Analogous to the proof of Theorem 5.1 in [7] there is a constant $\theta > 0$ such that

$$\|\mathbf{M}_{p\times p}^{-1}\| \leq \frac{\theta\,\Lambda}{\Delta}, \tag{2.1}$$

where θ is dependent on n, but independent of $\mathbf{Y}$ and Λ. Now, since $\mathbf{p}(\cdot) \in \mathcal{P}_n^2$, denoting $H = \nabla^2 \mathbf{p}$, we can write

$$\mathbf{p}(\mathbf{x}) = c + \mathbf{g}^{\mathsf{T}}\mathbf{x} + \frac{1}{2}\mathbf{x}^{\mathsf{T}} H \mathbf{x}. \tag{2.2}$$

Now, as in the proofs cited, defining $e^f(\mathbf{x})$ and $\mathbf{e}^g(\mathbf{x})$ to be the error functions

$$e^f(\mathbf{x}) = \mathbf{p}(\mathbf{x}) - f(\mathbf{x}) \tag{2.3}$$

and

$$\mathbf{e}^g(\mathbf{x}) = \nabla \mathbf{p}(\mathbf{x}) - \nabla f(\mathbf{x}) = \mathbf{g} + H\mathbf{x} - \nabla f(\mathbf{x}), \tag{2.4}$$

using the interpolation conditions, it is not difficult to show that for $i = 1, \dots, p$

$$\begin{aligned}
&(\mathbf{y}^i - \mathbf{x})^\top \mathbf{e}^g(\mathbf{x}) + \frac{1}{2}(\mathbf{y}^i)^\top H \mathbf{y}^i + (\mathbf{y}^i)^\top H\mathbf{x} + \frac{1}{2}\mathbf{x}^\top H \mathbf{x} \\
&= \int_0^1 (\nabla f(\mathbf{x} + t(\mathbf{y}^i - \mathbf{x})) - \nabla f(\mathbf{x}))^\top (\mathbf{y}^i - \mathbf{x}) dt - e^f(\mathbf{x}) \\
&= R(\mathbf{x}, \mathbf{y}^i) - e^f(\mathbf{x}),
\end{aligned} \tag{2.5}$$

where

$$R(\mathbf{x}, \mathbf{y}) := \int_0^1 (\nabla f(\mathbf{x} + t(\mathbf{y} - \mathbf{x})) - \nabla f(\mathbf{x}))^\top (\mathbf{y} - \mathbf{x}) dt.$$

Hence, noting that $\mathbf{y}^0 = 0$, denoting

$$\mathbf{M}_{p\times n} = \begin{bmatrix} y_1^1 & \dots & y_n^1 \\ y_1^2 & \dots & y_n^2 \\ \vdots & & \vdots \\ y_1^p & \dots & y_n^p \end{bmatrix},$$

we are able to setup the following linear system of equations

$$\mathbf{M}_{p\times n} \mathbf{e}^g(\mathbf{x}) = \mathbf{b}, \tag{2.6}$$

where $\mathbf{b} \in \mathbb{R}^p$ with $b_i = R(\mathbf{x}, \mathbf{y}^i) - R(\mathbf{x}, \mathbf{0}) - \frac{1}{2}(\mathbf{y}^i)^\top H \mathbf{y}^i - (\mathbf{y}^i)^\top H \mathbf{x}$.

Furthermore,

$$\|b\| \le (c + \|H\|)\Delta^2, \tag{2.7}$$

where $c > 0$ is a constant that is a scalar multiple of L_1. Now let $\mathbf{M}_{p\times p} = QR$, where Q is an $p \times p$ orthogonal matrix and R is an upper triangular matrix. Then $\mathbf{M}_{p\times n} = Q_1 R_1$, where Q_1 consists of the first n columns of Q, and R_1 is the first n rows and columns of R. Since $\mathbf{M}_{p\times p}$ is nonsingular, we have R is nonsingular and therefore R_1 is nonsingular. Then, by (2.6), we have

$$\mathbf{e}^g(\mathbf{x}) = (\mathbf{M}_{p\times n}^\top \mathbf{M}_{p\times n})^{-1} \mathbf{M}_{p\times n}^\top \mathbf{b} = R_1^{-1} Q_1^\top \mathbf{b}$$

From the construction, we know R_1^{-1} is the first n rows and columns of R^{-1}. Hence, $\|R_1^{-1}\| \le \|R^{-1}\|$. Again, by the construction $\|Q_1^\top \mathbf{b}\| \le \|Q^\top \mathbf{b}\|$. So it follows from Q being an orthogonal matrix that

$$\|R_1^{-1} Q_1^\top \mathbf{b}\| \le \|R_1^{-1}\| \|Q_1^\top \mathbf{b}\| \le \|R^{-1}\| \|Q^\top \mathbf{b}\| = \|R^{-1} Q^\top\| \|\mathbf{b}\|.$$

Hence, it follows from (2.1) and (2.7) that

$$\begin{aligned}
\|\mathbf{e}^g(\mathbf{x})\| &= \|R_1^{-1} Q_1^\top \mathbf{b}\| \le \|R^{-1} Q^\top\| \|\mathbf{b}\| = \|\mathbf{M}_{p\times p}^{-1}\| \|\mathbf{b}\| \\
&\le \frac{\theta\, \Lambda}{\Delta}(c + \|H\|)\Delta^2 \le \kappa_1(n, \Lambda, L_1)\,(\|\nabla^2 \mathbf{p}\| + 1)\,\Delta,
\end{aligned}$$

where $\kappa_1 > 0$ depends only on n, Λ and L_1. $\square$

Given a Λ-poised set $\mathbf{Y} \subset \mathcal{B}(\mathbf{z}, \Delta)$ with $n+1 \leq |\mathbf{Y}| \leq (n+1)(n+2)/2$, for each $i = 1, \ldots, m$, let $m_i(\mathbf{x}) \in \mathcal{P}_n^2$ be a polynomial interpolating model of $f_i(\mathbf{x})$ on $\mathbf{Y}$. Then, based on the above lemma, there exist positive constants κ_{eg} and κ_{ef} such that for any $\mathbf{s} \in \mathcal{B}(\mathbf{0}, \Delta)$,

$$\|\nabla f_i(\mathbf{z}+\mathbf{s}) - \nabla m_i(\mathbf{z}+\mathbf{s})\| \leq \kappa_{eg}\Delta, \tag{2.8}$$

$$|f_i(\mathbf{z}+\mathbf{s}) - m_i(\mathbf{z}+\mathbf{s})| \leq \kappa_{ef}\Delta^2, \tag{2.9}$$

for all $i = 1, \ldots, m$, where κ_{eg} and κ_{ef} are positive constants only depending on n, Λ, F and $\max\{\|\nabla^2 m_i\|, i = 1, \ldots, m\}$. Hence, if $\max\{\|\nabla^2 m_i\|, i = 1, \ldots, m\} \leq \kappa_H$, a fixed constant[1], then κ_{eg} and κ_{ef} will be fixed constants too. In particular, κ_{eg} and κ_{ef} depend neither on $\mathbf{z}$ nor Δ.

Now suppose we have a Λ-poised set $\mathbf{Y} \subset \mathcal{B}(\mathbf{z}, \Delta)$ with $n+1 \leq |\mathbf{Y}| \leq (n+1)(n+2)/2$, then for any $\mathbf{y} \in \mathcal{B}(\mathbf{z}, \Delta)$, we define a local quadratic model $\phi(\mathbf{y}, \mathbf{s})$ of $\Phi(\cdot)$ around $\mathbf{y}$ as

$$\phi(\mathbf{y}, \mathbf{s}) = c_\phi(\mathbf{y}) + \mathbf{g}_\phi(\mathbf{y})^{\top}\mathbf{s} + \frac{1}{2}\mathbf{s}^{\top} H_\phi(\mathbf{y})\mathbf{s}, \tag{2.10}$$

where

$$c_\phi(\mathbf{y}) = \frac{1}{2}\mathbf{m}(\mathbf{y})^{\top}\mathbf{m}(\mathbf{y}), \qquad \mathbf{g}_\phi(\mathbf{y}) = J(\mathbf{y})^{\top}\mathbf{m}(\mathbf{y}),$$

J is the Jacobian of $\mathbf{m}(\cdot)$ and

$$H_\phi(\mathbf{y}) = \begin{cases} J(\mathbf{y})^{\top} J(\mathbf{y}) & \text{if } \|\mathbf{g}_\phi(\mathbf{y})\| \geq \kappa_H^1, \\ J(\mathbf{y})^{\top} J(\mathbf{y}) + \kappa_H^3 \|\mathbf{m}(\mathbf{y})\| I & \text{if } \|\mathbf{g}_\phi(\mathbf{y})\| < \kappa_H^1 \text{ and } c_\phi(\mathbf{y}) < \kappa_H^2 \|\mathbf{g}_\phi(\mathbf{y})\|, \\ J(\mathbf{y})^{\top} J(\mathbf{y}) + \sum_{i=1}^m m_i(\mathbf{y})\, \nabla^2 m_i & \text{otherwise.} \end{cases}$$

Here, I denotes the identity matrix, κ_H^1, κ_H^2 and κ_H^3 are positive constants,

$$\mathbf{m}(\mathbf{y}) = (m_1(\mathbf{y}), m_2(\mathbf{y}), \ldots, m_m(\mathbf{y}))^{\top}, \; J(\mathbf{y}) = (\nabla m_1(\mathbf{y}), \nabla m_2(\mathbf{y}), \ldots, \nabla m_m(\mathbf{y}))^{\top}$$

and $m_i(\cdot) \in \mathcal{P}_n^2$, $i = 1, \ldots, m$, are the polynomial interpolating models of $f_i(\cdot)$ on $\mathbf{Y}$. By Lemma 2.2 and the above discussions, we know $m_i(\cdot)$ satisfies (2.8) and (2.9) for all $i = 1, \ldots, m$. Typical values of κ_H^1, κ_H^2 and κ_H^3 are 1, 1 and 0.01 respectively. It is not unusual to 'convexify' non-convex problems by adding a multiple of a positive definite Hessian to the second-order terms. This is what we do in the middle term of the definition of the Hessian above and such modifications result in what are termed "regularized Hessians". From the local analysis in Section 4, we will see that this regularization essentially prevents a situation when the trust region step could be unnecessarily large along the null space of the Jacobian, which would cause the loss of local quadratic convergence for zero residue problems. Since we may be doing this even when $J(\mathbf{y})^{\top} J(\mathbf{y}) + \sum_{i=1}^m m_i(\mathbf{y})\, \nabla^2 m_i$ is positive semi-definite, strictly speaking it may sometimes be an unnecessary regularization.

By way of further motivation for the definition of $H_\phi(\mathbf{y})$ we remark that, if $\mathbf{g}_\phi(\mathbf{y})$ is sufficiently large in magnitude, it is reasonable to assume that the first-order changes

[1] By 'fixed constant' we mean to emphasize that these constants will stay the same throughout a particular instance of the algorithm, i.e. for all iterations.

in the least-squares objective dominate locally and so one may take a Gauss-Newton approximation and ignore the curvature of the residuals. This is true even if we do not have a relatively small residual problem locally i.e $\mathbf{m}(\mathbf{y})$ may not be small in magnitude. If this first-order change is **relatively small** but the residuals are also small then the second definition can be considered as a sensible compromise in that some damping via the residual curvature is introduced using the identity (replacing $\pm\nabla^2 m_i$) weighted by the current overall model residual. Note the emphasis on relatively. The second case is triggered both by the size of the residual and the stationarity condition, which must be related but not necessarily very small. Otherwise a genuine second-order model is used. If one considers that the middle expression is approximating the terms involving the Hessian of the m_i's (as a regularisation) by the identity matrices, then mabye it is more reasonable to use the l_1-norm for the regularisation rather than the l_2-norm. For different reasons, this norm would be advantageous for the trust region when one includes bound constraints.

This is at variance with a standard Levenberg-Marquardt algorithm which can be considered as a regularized Gauss-Newton method that is designed with small residual problems in mind. Thus it defines

$$H_\phi(\mathbf{y}) = \begin{cases} J(\mathbf{y})^\top J(\mathbf{y}) & \text{if the residual is considered small} \\ J(\mathbf{y})^\top J(\mathbf{y}) + \kappa_H^3\ I & \text{otherwise.} \end{cases}$$

It makes no attempt to use the true Hessian, so in that sense it is counting on the problem being a small residual one.

We now derive a derivative-free algorithm for least-square minimization that takes into account the structure of the problem but that otherwise is close in spirit to the ideas presented in [5] for global convergence and the algorithm applied in Powell's NEWUOA software [21], that emphasizes practical efficiency. We note that the details on how to ensure that the interpolating set $\mathbf{Y}$ remains well-poised are omitted since they can readily be garnered from [5] and [6]. Throughout the algorithm statement below, we fix the number of points in the sampling set i.e. $|\mathbf{Y}_k| = N_P$, for all $k \geq 0$, where $N_P \in [n+1, (n+1)(n+2)/2]$ is an integer constant. We denote the resulting iterates by x_k where k is a positive integer.

A Derivative-Free algorithm for least-squares minimization (DFLS)

Step 0 **(Initialization)** Choose the starting guess $\mathbf{x}_0$, $0 < \rho_0 \leq \bar{\Delta}_0 \leq \Delta_0 \leq \Delta_{max}$ and N_P, the number of sampling points, with $N_P \geq n+1$. Choose an initial set of interpolation points, $\mathbf{Y}_0$, with $\mathbf{x}_0 \in \mathbf{Y}_0 \subset \mathcal{B}(\mathbf{x}_0, \bar{\Delta}_0)$. Choose $\epsilon_\beta \in (0,1)$ and $\beta > 0$. Set $k = 0$.

Step 1 **(Criticality step)** Choose a base point $\mathbf{y}_k \in \mathbf{Y}_k$ and calculate

$$\mathbf{g}_{\phi_k} = J(\mathbf{y}_k)^\top \mathbf{m}(\mathbf{y}_k) + H_\phi(\mathbf{y}_k)(\mathbf{x}_k - \mathbf{y}_k), \tag{2.11}$$

where $\mathbf{y}_k \in \mathbf{Y}_k$. If $\|\mathbf{g}_{\phi_k}\| \leq \epsilon_\beta$, let $\bar{\Delta}_k^{(0)} = \bar{\Delta}_k$; possibly modifying $\mathbf{Y}_k$ as needed to make sure $\mathbf{Y}_k$ is Λ-poised in $\mathcal{B}(\mathbf{x}_k, \bar{\Delta}_k)$, and determining the corresponding interpolation model $\mathbf{m}(\mathbf{y}_k)$. Here $\bar{\Delta}_k = \min\{\bar{\Delta}_k^{(0)}, \beta\|\mathbf{g}_{\phi_k}\|\}$, where $\mathbf{g}_{\phi_k}$ is recalculated using (2.11) with the new $\mathbf{Y}_k$, if $\mathbf{Y}_k$ has changed. It is shown in [4], Lemma 7.3, that unless y_k is a first-order stationary point, this can be achieved in a finite number of steps.

Step 2 **(Step calculation)** Solve the following trust region subproblem :

$$\begin{array}{ll} \min & \phi_k(\mathbf{d}) \\ s.t. & \|\mathbf{d}\| \le \Delta_k, \end{array} \tag{2.12}$$

where $\phi_k(\mathbf{d}) = \phi(\mathbf{y}_k, (\mathbf{x}_k - \mathbf{y}_k) + \mathbf{d})$ with $\phi(\cdot,\cdot)$ defined by (2.10), to obtain the step[2] $\mathbf{d}_k$.

Step 3 **(Safety step)** This step applies only when $\|\mathbf{d}_k\| < \frac{1}{2}\rho_k$ and $\|\mathbf{g}_{\phi_k}\| > \epsilon_\beta$.

```
3.1 Let $i = 0$, $\Delta_k^{(0)} = \Delta_k$.
3.2 Choose $\bar{\Delta}_k^{(i)} \in [\rho_k, \Delta_k^{(i)}]$.
3.3 If $\bar{\Delta}_k^{(i)} > \rho_k$, then
        If $\mathbf{Y}_k$ is $\Lambda$-poised in $\mathcal{B}(\mathbf{x}_k, \bar{\Delta}_k^{(i)})$, then
            Let $\Delta_k^{(i+1)} = \max\{\Delta_k^{(i)}/10, \rho_k\}$ . $i = i + 1$, go to 3.2.
        Else
            Let $\Delta_{k+1} = \bar{\Delta}_k^{(i)}, \rho_{k+1} = \rho_k$.
        Endif
    Else (i.e. $\bar{\Delta}_k^{(i)} = \rho_k$)
        If $\mathbf{Y}_k$ is $\Lambda$-poised in $\mathcal{B}(\mathbf{x}_k, \bar{\Delta}_k^{(i)})$, then
            Let $\Delta_{k+1} = \rho_k/2, \rho_{k+1} = \rho_k/10$.
        Else
            Let $\Delta_{k+1} = \bar{\Delta}_k^{(i)}, \rho_{k+1} = \rho_k$.
        Endif
    Endif
    Let $\mathbf{x}_{k+1} = \mathbf{x}_k$ and choose $\bar{\Delta}_{k+1} \in [\rho_{k+1}, \Delta_{k+1}]$.
    Modify $\mathbf{Y}_k$ to form $\mathbf{Y}_{k+1}$ such that $\mathbf{Y}_{k+1}$ is $\Lambda$-poised in $\mathcal{B}(\mathbf{x}_{k+1}, \bar{\Delta}_{k+1})$.
    Set $k = k + 1$, go to Step 1.
```

Step 4 **(Acceptance of the trial step)** This step applies only when $\|\mathbf{d}_k\| \ge \frac{1}{2}\rho_k$ or $\|\mathbf{g}_{\phi_k}\| \le \epsilon_\beta$.

Compute $\Phi(\mathbf{x}_k + \mathbf{d}_k)$ and $r_k := Ared_k/Pred_k$, where the actual reduction is defined by

$$Ared_k = \Phi(\mathbf{x}_k) - \Phi(\mathbf{x}_k + \mathbf{d}_k), \tag{2.13}$$

while the predicted reduction is defined by

$$Pred_k = \phi_k(\mathbf{0}) - \phi_k(\mathbf{d}_k). \tag{2.14}$$

If $r_k > 0$, then $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{d}_k$; otherwise, $\mathbf{x}_{k+1} = \mathbf{x}_k$.

Step 5 **(Trust region radius update)** Set

$$\tilde{\Delta}_{k+1} = \begin{cases} \frac{1}{2}\|\mathbf{d}_k\| & \text{if } r_k < 0.1, \\ \max\{\frac{1}{2}\Delta_k,\ \|\mathbf{d}_k\|\} & \text{if } 0.1 \le r_k < 0.7, \\ \max\{\Delta_k,\ 2\|\mathbf{d}_k\|\} & \text{if } r_k \ge 0.7, \end{cases} \tag{2.15}$$

and

$$\Delta_{k+1} = \min\{\max\{\tilde{\Delta}_{k+1},\ \rho_k\}, \Delta_{max}\}. \tag{2.16}$$

If $r_k \ge 0.1$, then

[2] In other words we build the model at y_k but shift our center to x_k

```
            Let $\rho_{k+1} = \rho_k$ and choose $\bar{\Delta}_{k+1} \in [\rho_k, \Delta_{k+1}]$.
            The interpolating points set $\mathbf{Y}_k$ is updated to take into consideration of
            the new point $\mathbf{x}_{k+1}$ to form $\mathbf{Y}_{k+1} \in \mathcal{B}(\mathbf{x}_{k+1}, \bar{\Delta}_{k+1})$ .
            Set $k = k + 1$, go to Step 1.
        Endif
Step 6  (Model improvement) This step applies only when $r_k < 0.1$.
        If $\mathbf{Y}_k$ is $\Lambda$-poised in $\mathcal{B}(\mathbf{x}_k, \Delta_k)$ and $\Delta_k = \rho_k$, then
            Let $\Delta_{k+1} = \rho_k/2$ and $\rho_{k+1} = \rho_k/10$. Choose $\bar{\Delta}_{k+1} \in [\rho_{k+1}, \Delta_{k+1}]$.
            The interpolating points set $\mathbf{Y}_k$ is updated to take into consideration of
            the new point $\mathbf{x}_{k+1}$ to form $\mathbf{Y}_{k+1} \in \mathcal{B}(\mathbf{x}_{k+1}, \bar{\Delta}_{k+1})$ .
        Else
            Let $\rho_{k+1} = \rho_k$. Choose $\bar{\Delta}_{k+1} \in [\rho_{k+1}, \Delta_{k+1}]$. Possibly modification of
            $\mathbf{Y}_k$ is needed to form $\mathbf{Y}_{k+1}$ such that $\mathbf{Y}_{k+1}$ is $\Lambda$-poised in $\mathcal{B}(\mathbf{x}_{k+1}, \bar{\Delta}_{k+1})$.
            $\Delta_{k+1} = \Delta_k$
        Endif
        Set $k = k + 1$, go to Step 1.
```

We remark that in theory it is a good idea to choose the base-point in Step 1 to be the best point seen to date. However, in practise this involves shifting the bases of the models, which requires $\mathcal{O}(n^3)$ work, while in most steps we are trying to keep the work down to $\mathcal{O}(n^2)$. The trust region for these methods serves two purposes: it restricts the step size to the neighborhood where the model is assumed to be good, and it also defines the neighborhood in which the points are sampled for the construction of the model. Powell in [23] suggests using two different trust regions to separate these two roles, and in some sense Δ_k and ρ_k are these two regions. Note that ρ_k is a lower bound on the size of the sampling region radius $\bar{\Delta}_k$ except when a measure of stationarity is close to zero and Step 1 is applied. For example, when the current iterate is far away from stationarity, it is clearly a bad idea to have all the sample points very close to the center of the trust region. However, when a measure of the stationarity is close to zero (that is the current iterate may be close to a stationary point), it is important to ensure that the sampling set is well-poised in a region with a size comparable to the measure of stationarity, so that the models become more accurate. In this way, a better iterate should be generated in the next iteration, which will essentially lead to global and local convergence. This is exactly the purpose served by the Criticality Step (Step 1). On the other hand, as we have already mentioned, it is possible to dispense with two regions and only use one, as is done in [4].

Another important feature of using ρ_k in the algorithm is to prevent the trust region shrinking too fast before the model becomes sufficiently "good" in the current trust region. In other words, when a "bad" trial trust-region point is generated, instead of reducing the trust-region radius immediately, the algorithm should first make sure the model is a "good" model, i.e. the sampling points are well-poised in the current trust region. Otherwise, the trust-region radius will be reduced too quickly for the wrong reasons, consequently allowing only very small stepsizes which, at best, will make the algorithm very inefficient. These ideas are implemented in Step 3, Step 5 and Step 6. Notice that, following Powell, we apply a safety step (Step 3) when the iterate is relatively far away from stationarity and the trial trust-region step is too small (less than half of the lower bound of the size of the sampling region radius). We do not even calculate the true function value at this trial point, since most probably this is a "bad" point in the sense that it may not only have a higher function value (remember we assume the true function evaluation is expensive), but we also might destroy the geometry of future sampling sets. In addition, the updating procedure of the trust-region radius in Step 3 means that repeatedly executing this step eventually forces the trust-region radius to the lower bound ρ_k. Hence, Step 3 will finish in a finite number of steps. And when ρ_k goes to zero, it also defines a natural stopping criterion for this class of methods.

Theoretically, as we already alluded to earlier, we always have the possibility of building the trust region model at the current iterate, i.e. letting $\mathbf{y}_k = \mathbf{x}_k$ at Step 2. However, in practice it is not a good idea to move the base point of the model to be the current iterate, $\mathbf{x}_k$, at every iteration. One of the important reasons lies in the fact that shifting the base point of the model often requires $O(n^3)$ computer operations, while we want to keep the amount of work to be $O(n^2)$ at each iteration with a corresponding number of sampling points N_P of $O(n)$. However, to avoid unacceptable numerical errors, in practice it may be desirable to move the base point to the current iterate when the trust region step is sufficiently small compared with the distance between the current iterate and the base point. For example, in our implementation we move $\mathbf{y}_k$ to be $\mathbf{x}_k$ when $\|\mathbf{d}_k\| \leq 0.1\|\mathbf{x}_k - \mathbf{y}_k\|$. For details, the reader is referred to [21]. In general, we expect $\mathbf{y}_k$ to be different from $\mathbf{x}_k$. Nevertheless, when we discuss local convergence properties in Section 4, we always assumes $\mathbf{y}_k = \mathbf{x}_k$. We also remark that in Step 4 we are accepting simple decrease and in Step 5 the trust region management relates to the size of the actual step taken rather than just to the trust-region radius. Again, as one would expect, one only reduces the trust region radius when one is certain failure is because of genuine characteristics of the optimization problem and not because one has a bad model as a consequence of the poor geometry of the interpolation points. In addition, notice that in Step 5, when a sufficient function reduction is obtained, i.e. when $r_k \geq 0.1$, the algorithm only requires that the next sampling set, Y_{k+1}, is formed by incorporating $\mathbf{x}_{k+1}$, without regard to the well-poisedness of Y_{k+1}. This is desirable since checking that the sampling set is well-poised sometimes requires a significant amount of computation. We do want to avoid this unnecessary work as long as the algorithm performs well and global convergence is guaranteed. In practice a normal procedure in this case is just to exchange $\mathbf{x}_{k+1}$ with the sampling point in Y_k which is farthest away from the base point.

Now, we define $conv(L_{enl}(\mathbf{x}_0))$ to be the convex hull of $L_{enl}(\mathbf{x}_0)$ with

$$L_{enl}(\mathbf{x}_0) = \bigcup_{\mathbf{x} \in L(\mathbf{x}_0)} \mathcal{B}(\mathbf{x}, \Delta_{max}) \quad \text{and} \quad L(\mathbf{x}_0) = \{\mathbf{x} \in \mathbb{R}^n : \Phi(\mathbf{x}) \leq \Phi(\mathbf{x}_0)\}.$$

Throughout this paper, we also need the following assumptions.

ASSUMPTION 2.2. *The level set $L(\mathbf{x}_0) = \{\mathbf{x} \in \mathbb{R}^n : \Phi(\mathbf{x}) \leq \Phi(\mathbf{x}_0)\}$ is bounded.*

ASSUMPTION 2.3. *There exists a constant κ_H, independent of k, the iteration count for the algorithm DFLS, such that if m_i, $i = 1, \ldots, m$, is the polynomial interpolating model of f_i on a Λ-posed sampling set $\mathbf{Y}_k$ constructed as in the DFLS algorithm, then*

$$\|\nabla^2 m_i\| \leq \kappa_H \tag{2.17}$$

for all $i = 1, \ldots, m$.

Based on Assumption 2.2, the following lemmas holds.

LEMMA 2.3. *Under Assumption 2.2, there exist positive constants L_0, L_1 and L_2,* *such that*

$$\|F(\mathbf{x})\| \le L_0, \quad \|F(\mathbf{x}) - F(\mathbf{y})\| \le L_1\|\mathbf{x}-\mathbf{y}\| \quad \textit{and} \quad \|\nabla F(\mathbf{x})\| \le L_1,$$
$$\|\nabla F(\mathbf{x}) - \nabla F(\mathbf{y})\| \le L_2\|\mathbf{x}-\mathbf{y}\| \quad \textit{and} \quad \|\nabla^2 f_i(\mathbf{x})\| \le L_2, \quad i = 1,\dots,m,$$

for any $\mathbf{x}, \mathbf{y} \in conv(L_{enl}(\mathbf{x}_0))$.

Proof. Since $L(\mathbf{x}_0)$ is bounded by Assumption 2.2 and $\Delta_{max} > 0$ is a constant, we know $L_{enl}(\mathbf{x}_0)$ is bounded. Hence, its convex hull $conv(L_{enl}(\mathbf{x}_0))$ is also bounded. Then, it follows from $f_i(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}, i = 1,\dots,m$, are twice continuously differentiable that this lemma holds. □

The following lemma indicates that there are interpolation models such that Assumption 2.3 holds.

Lemma 2.4. *Suppose* m_i*,* $i = 1,\dots,m$*, is the polynomial interpolating model of* f_i *on a* Λ*-posed sampling set* $\mathbf{Y}_k$ *constructed as in iteration* k *of the DFLS algorithm satisfying*

$$m_i(\mathbf{x}) = \underset{q}{\text{Argmin}} \quad \|\nabla^2 q(\mathbf{x})\| \qquad (2.18)$$
$$s.t. \quad q(\mathbf{y}) = f_i(\mathbf{y}), \quad \textit{for all } \mathbf{y} \in \mathbf{Y}_k,$$

then there exists a constant κ_H*, independent of* k*, such that (2.17) holds.*

Proof. Fix any index $i \in [1, m]$. From the DFLS algorithm, we know $\mathbf{Y}_k$ is Λ-posed in $\mathcal{B}(\mathbf{x}_k, \bar{\Delta}_k)$ for a polynomial space $\mathcal{P} \subseteq \mathcal{P}_n^2$. Therefore, there exists a set $\mathbf{S}_k$ with $|\mathbf{S}_k| = (n+1)(n+2)/2 - |\mathbf{Y}_k|$ (we take $\mathbf{S}_k$ to be empty, when $|\mathbf{Y}_k| = (n+1)(n+2)/2$) such that $\bar{\mathbf{Y}}_k := \mathbf{Y}_k \cup \mathbf{S}_k$ is Λ-posed in $\mathcal{B}(\mathbf{x}_k, \bar{\Delta}_k)$ for $\mathcal{P}_n^2$. (By the procedures in [5] and [6], we can actually construct such a set $\mathbf{S}_k$.) Now, consider $\bar{m}_i(\mathbf{x})$ to be the quadratic interpolating polynomial of f_i on the $(n+1)(n+2)/2$ Λ-poised points of $\bar{\mathbf{Y}}_k$. Since $\bar{\mathbf{Y}}_k$ is Λ-posed in $\mathcal{B}(\mathbf{x}_k, \bar{\Delta}_k)$ for $\mathcal{P}_n^2$, we know $\bar{m}_i(\mathbf{x})$ is unique and it follows from the Theorem 3 in [5] that there are exists a constant κ_{eH} such that for any $\mathbf{s} \in \mathcal{B}(\mathbf{x}_k, \bar{\Delta}_k)$,

$$\|\nabla^2 \bar{m}_i - \nabla^2 f_i(\mathbf{s})\| \le \kappa_{eH}\bar{\Delta}_k.$$

Hence, since $\mathbf{x}_k \in \mathcal{B}(\mathbf{x}_k, \bar{\Delta}_k)$, we have

$$\|\nabla^2 \bar{m}_i\| \le \|\nabla^2 f_i(\mathbf{x}_k)\| + \kappa_{eH}\bar{\Delta}_k. \qquad (2.19)$$

Because $\mathbf{Y}_k \subseteq \bar{\mathbf{Y}}_k$, by the construction of $\bar{m}_i(\mathbf{x})$ and $m_i(\mathbf{x})$ satisfying (2.18), we have

$$\|\nabla^2 m_i\| \le \|\nabla^2 \bar{m}_i\|.$$

Thus, from (2.19), we know

$$\|\nabla^2 m_i\| \le \|\nabla^2 f_i(\mathbf{x}_k)\| + \kappa_{eH}\bar{\Delta}_k.$$

It follows from DFLS algorithm that $\mathbf{x}_k \in L(\mathbf{x}_0)$ and $\bar{\Delta}_k \le \Delta_{max}$. Therefore,

$$\|\nabla^2 m_i\| \le \max_{\mathbf{x} \in L(\mathbf{x}_0)} \|\nabla^2 f_i(\mathbf{x})\| + \kappa_{eH}\Delta_{max}.$$

Now, by Lemma 2.3, we know

$$\|\nabla^2 m_i\| \le \kappa_H := L_2 + \kappa_{eH}\Delta_{max}.$$

□

We remark that we are in fact choosing the "flatest" interpolating polynomial, which given that we are trying to emulate a Gauss-Newton method, seems reasonable.

LEMMA 2.5. *Under Assumptions 2.2 and 2.3, there exist constants κ_m and κ_g, independent of k, such that if m_i, $i = 1, \ldots, m$, is the polynomial interpolating model of f_i on a Λ-posed sampling set $\mathbf{Y}_k$ constructed as in the DFLS algorithm, then for any $\mathbf{y} \in \mathbf{Y}_k$,*

$$|m_i(\mathbf{y})| \leq \kappa_m \quad \text{and} \quad \|\nabla m_i(\mathbf{y})\| \leq \kappa_g, \tag{2.20}$$

for all $i = 1, \ldots, m$. Hence, there exists constant κ_H^ϕ such that

$$\|H_\phi(\mathbf{y})\| \leq \kappa_H^\phi, \tag{2.21}$$

for any $\mathbf{y} \in \mathbf{Y}_k$.

Proof. Fix any index $i \in [1, m]$. From the DFLS algorithm can see that for all $k \geq 0$,

$$\mathbf{Y}_k \subseteq L_{enl}(\mathbf{x}_0). \tag{2.22}$$

Since $f_i(\mathbf{y}) = m_i(\mathbf{y})$ for any $\mathbf{y} \in \mathbf{Y}_k$, it follows from Lemma 2.3 that

$$|m_i(\mathbf{y})| \leq \kappa_m := L_0$$

for any $\mathbf{y} \in \mathbf{Y}_k$. Now, by Assumption 2.3, $\bar{\mathbf{Y}}_k$ being Λ-posed in $\mathcal{B}(\mathbf{x}_k, \bar{\Delta}_k)$, $\mathbf{y} \in \mathbf{Y_k} \subset \mathcal{B}(\mathbf{x}_k, \bar{\Delta}_k)$ and (2.8), we have

$$\|\nabla f_i(\mathbf{y}) - \nabla m_i(\mathbf{y})\| \leq \kappa_{eg} \bar{\Delta}_k,$$

with κ_{eg} a fixed constant. Therefore, by (2.22), we know

$$\|\nabla m_i(\mathbf{y})\| \leq \max_{\mathbf{x} \in L_{enl}(\mathbf{x}_0)} \|\nabla f_i(\mathbf{y})\| + \kappa_{eg} \bar{\Delta}_k.$$

Then, it follows from Lemma 2.3 and $\bar{\Delta}_k \leq \Delta_{max}$ that

$$\|\nabla m_i(\mathbf{y})\| \leq \kappa_g := L_1 + \kappa_{eg} \Delta_{max}.$$

Hence, by the definition of $H_\phi(\cdot)$ and (2.20), (2.21) holds. □

3. Global Convergence.

We will need the following lemma, which can be obtained directly from the well-known result (due to Powell [24], see also [3], p.125). We omit its proof here.

LEMMA 3.1. *Let $\mathbf{d}_k$ be the solution of the trust region subproblem (2.12), then*

$$\phi_k(\mathbf{0}) - \phi_k(\mathbf{d}_k) \geq \frac{1}{2} \|\mathbf{g}_{\phi_k}\| \min \left\{ \Delta_k, \frac{\|\mathbf{g}_{\phi_k}\|}{\|H_\phi(\mathbf{y}_k)\|} \right\}, \tag{3.1}$$

where $\mathbf{g}_{\phi_k}$ is defined by (2.11).

With regard to the Step 1 (Criticality step) in the DFLS algorithm, we have the following lemma. It is proved in [4] (Lemma 5.1).

LEMMA 3.2. *If $\|\nabla \Phi(\mathbf{x}_k)\| \neq 0$, Step 1 (Criticality step) in the DFLS algorithm terminates in a finite number of improvement steps.*

For the function value reduction ratio $r_k = Ared_k/Pred_k$ defined in Step 4 of the DFLS algorithm, we have the following, that corresponds to a relatively standard result[3] (see, for example [3], Theorem 6.4.2).

LEMMA 3.3. *Under Assumptions 2.2 and 2.3, suppose $\mathbf{Y}_k \subset \mathcal{B}(\mathbf{x}_k, \Delta_k)$ is Λ-poised. Then, for any $\epsilon > 0$, there exists a positive constant c_ϵ, independent of k, such that if $\Delta_k \le c_\epsilon \|\mathbf{g}_{\phi_k}\|$, then*

$$|r_k - 1| \le \epsilon. \tag{3.2}$$

Proof. For any $\mathbf{y}_k \in \mathbf{Y}_k$ and $\mathbf{s} \in \mathbb{R}^n$, it follows from (1.1), (2.8), (2.10), Lemma 2.3, Lemma 2.5 and $f_i(\mathbf{y}_k) = m_i(\mathbf{y}_k)$, for $i = 1, \ldots, m$, that

$$\begin{aligned}
&\Phi(\mathbf{s}) - \phi(\mathbf{y}_k, \mathbf{s} - \mathbf{y_k}) \\
&= \Phi(\mathbf{s}) - [c_\phi(\mathbf{y}_k) + \mathbf{g}_\phi(\mathbf{y}_k)^{\mathsf{T}}(\mathbf{s} - \mathbf{y}_k) + \frac{1}{2}(\mathbf{s} - \mathbf{y}_k)^{\mathsf{T}} H_\phi(\mathbf{y}_k)(\mathbf{s} - \mathbf{y}_k)] \\
&\le \frac{1}{2}\|F(\mathbf{s})\|^2 - [\frac{1}{2}\|\mathbf{m}(\mathbf{y}_k)\|^2 + (J(\mathbf{y}_k)^{\mathsf{T}}\mathbf{m}(\mathbf{y}_k))^{\mathsf{T}}(\mathbf{s} - \mathbf{y}_k)] + \kappa_H^\phi \|\mathbf{s} - \mathbf{y}_k\|^2 \\
&\le \frac{1}{2}\|F(\mathbf{s})\|^2 - [\frac{1}{2}\|F(\mathbf{y}_k)\|^2 + (J(\mathbf{y}_k)^{\mathsf{T}}F(\mathbf{y}_k))^{\mathsf{T}}(\mathbf{s} - \mathbf{y}_k)] + \kappa_H^\phi \|\mathbf{s} - \mathbf{y}_k\|^2 \\
&\le F(\mathbf{y}_k)^{\mathsf{T}}[\nabla F(\mathbf{y}_k) - J(\mathbf{y}_k)](\mathbf{s} - \mathbf{y}_k) + c_1\|\mathbf{s} - \mathbf{y}_k\|^2 \\
&\le c_2 \Delta_k \|\mathbf{s} - \mathbf{y}_k\| + c_1 \|\mathbf{s} - \mathbf{y}_k\|^2,
\end{aligned} \tag{3.3}$$

where $c_1 > 0$ and $c_2 > 0$ are appropriately chosen constants. Note that in order to obtain the last inequality above we used the fact that (2.8) holds with κ_{eg} being a fixed constant. Now substituting $\mathbf{s}$ in (3.3) by $\mathbf{x}_k$ and $\mathbf{x}_k + \mathbf{d}_k$ respectively, since $\mathbf{y}_k \in \mathbf{Y}_k \subset \mathcal{B}(\mathbf{x}_k, \bar{\Delta}_k) \subseteq \mathcal{B}(\mathbf{x}_k, \Delta_k)$ and $\|\mathbf{d}_k\| \le \Delta_k$, we have

$$\begin{aligned}
\Phi(\mathbf{x}_k) - \phi_k(\mathbf{0}) &= \Phi(\mathbf{x}_k) - \phi(\mathbf{y}_k, \mathbf{x}_k - \mathbf{y_k}) \\
&\le c_2 \Delta_k \|\mathbf{x}_k - \mathbf{y}_k\| + c_1 \|\mathbf{x}_k - \mathbf{y}_k\|^2 \\
&\le c\Delta_k^2,
\end{aligned}$$

and

$$\begin{aligned}
\Phi(\mathbf{x}_k + \mathbf{d}_k) - \phi_k(\mathbf{d}_k) &= \Phi(\mathbf{x}_k + \mathbf{d}_k) - \phi(\mathbf{y}_k, \mathbf{x}_k - \mathbf{y_k} + \mathbf{d}_k) \\
&\le c_2 \Delta_k \|\mathbf{x}_k - \mathbf{y}_k + \mathbf{d}_k\| + c_1 \|\mathbf{x}_k - \mathbf{y}_k + \mathbf{d}_k\|^2 \\
&\le c\Delta_k^2.
\end{aligned}$$

Where again $c_1 > 0$ and $c_2 > 0$ are appropriately chosen constants and $c = c_1 + c_2$. The above inequalities together with (3.1) imply

$$\begin{aligned}
|r_k - 1| &= \left| \frac{\Phi(\mathbf{x}_k) - \Phi(\mathbf{x}_k + \mathbf{d}_k)}{\phi_k(\mathbf{0}) - \phi_k(\mathbf{d}_k)} - 1 \right| \\
&\le 4c \frac{\Delta_k^2}{\|\mathbf{g}_{\phi_k}\| \min\left\{\Delta_k, \frac{\|\mathbf{g}_{\phi_k}\|}{\|H_\phi(\mathbf{y}_k)\|}\right\}}.
\end{aligned}$$

Noticing $\|H_\phi(\mathbf{y}_k)\| \le \kappa_H^\phi$ by (2.21), and choosing $c_\epsilon = \min\{\epsilon/4c, 1/\kappa_H^\phi\}$, we have (3.2). □

[3] This result states that, supposing we have good geometry, when Δ_k becomes small compared with the criticality criteria, the overall model is extremely good relatively.

The following Lemma corresponds to, for example [3], Theorem 6.4.3. It states that if one supposes that the criticality measure is bounded away from zero then so is the trust-region radius.

Lemma 3.4. *Under Assumptions 2.2, 2.3, suppose that there exists a constant $\delta_1 > 0$ such that $\|\mathbf{g}_{\phi_k}\| \geq \delta_1$ for all k. Then, there exists a constant $\delta_2 > 0$ such that*

$$\Delta_k \geq \delta_2,$$

for all k.

Proof. We prove this lemma by way of contradiction. Suppose there exists an infinite subsequence $\{k_i\}$ such that

$$\lim_{i\to\infty} \Delta_{k_i} = 0.$$

By the DFLS algorithm, we know $\rho_k \leq \Delta_k$ for all k. Hence,

$$\lim_{i\to\infty} \rho_{k_i} = 0. \tag{3.4}$$

Without loss of generality, we can choose a subsequence such that $\rho_{k_i} < \rho_{k_i-1}$. So, by Step 3 or Step 6 of the DFLS algorithm, we know

$$\rho_{k_i} = \rho_{k_i-1}/10, \quad \Delta_{k_i} = \rho_{k_i-1}/2 < \Delta_{k_i-1}, \quad \|\mathbf{d}_{k_i-1}\| \leq \rho_{k_i-1}, \tag{3.5}$$

and $\mathbf{Y}_{k_i-1} \subset \mathcal{B}(\mathbf{x}_{k_i-1}, \rho_{k_i-1}) \subseteq \mathcal{B}(\mathbf{x}_{k_i-1}, \Delta_{k_i-1})$ is Λ-poised. It also follows from the DFLS algorithm that $\{\rho_k\}$ is a non-increasing sequence. Hence, by (3.4) and (3.5), we have

$$\lim_{i\to\infty} \rho_{k_i-1} = \lim_{i\to\infty} \|\mathbf{d}_{k_i-1}\| = 0. \tag{3.6}$$

Since $\mathbf{d}_{k_i-1}$ is a solution of the trust region subproblem (2.12), by Lemma 3.1, we have

$$\begin{aligned}\|\mathbf{d}_{k_i-1}\| &\geq \min\left\{\frac{\|\mathbf{g}_{\phi_{k_i-1}}\|}{\|H_\phi(\mathbf{y}_{k_i-1})\|},\ \Delta_{k_i-1}\right\} \\ &\geq \min\left\{\frac{\delta_1}{\kappa_H^\phi}, \Delta_{k_i-1}\right\},\end{aligned} \tag{3.7}$$

where $\kappa_H^\phi > 0$ is the constant in (2.21) and we used the fact that $\mathbf{d}_{k_i-1}$ is a solution of the trust region subproblem (2.12) with, by definition,

$$\begin{aligned}\phi_{k_i-1}(\mathbf{d}) &= \phi(\mathbf{y}_{k_i-1}, (\mathbf{x}_{k_i-1} - \mathbf{y}_{k_i-1}) + \mathbf{d}) \\ &= c_\phi(\mathbf{y}_{k_i-1}) + \mathbf{g}_\phi(\mathbf{y}_{k_i-1})^\mathsf{T}((\mathbf{x}_{k_i-1} - \mathbf{y}_{k_i-1}) + \mathbf{d}) \\ &\quad + \frac{1}{2}((\mathbf{x}_{k_i-1} - \mathbf{y}_{k_i-1}) + \mathbf{d})^\mathsf{T} H_\phi(\mathbf{y}_{k_i-1})((\mathbf{x}_{k_i-1} - \mathbf{y}_{k_i-1}) + \mathbf{d}) \\ &= \phi_{k_i-1}(\mathbf{0}) + (\mathbf{g}_\phi(\mathbf{y}_{k_i-1}) + H_\phi(\mathbf{y}_{k_i-1})(\mathbf{x}_{k_i-1} - \mathbf{y}_{k_i-1}))^\mathsf{T}\mathbf{d} + \frac{1}{2}\mathbf{d}^\mathsf{T} H_\phi(\mathbf{y}_{k_i-1})\mathbf{d} \\ &= \phi_{k_i-1}(\mathbf{0}) + \mathbf{g}_{\phi_{k_i-1}}^\mathsf{T}\mathbf{d} + \frac{1}{2}\mathbf{d}^\mathsf{T} H_\phi(\mathbf{y}_{k_i-1})\mathbf{d},\end{aligned}$$

where, by definition of (2.11),

$$\mathbf{g}_{\phi_{k_i-1}} = \mathbf{g}_\phi(\mathbf{y}_{k_i-1}) + H_\phi(\mathbf{y}_{k_i-1})(\mathbf{x}_{k_i-1} - \mathbf{y}_{k_i-1}).$$

So, $\mathbf{d}_{k_i-1}$ is a solution of the following problem

$$\begin{array}{ll} \min & \mathbf{g}_{\phi_{k_i-1}}^\mathsf{T}\mathbf{d} + \frac{1}{2}\mathbf{d}^\mathsf{T} H_\phi(\mathbf{y}_{k_i-1})\mathbf{d} \\ s.t. & \|\mathbf{d}\| \le \Delta_{k_i-1} \end{array}$$

Now suppose $\mathbf{d}_{k_i-1} < \Delta_{k_i-1}$, we have $H_\phi(\mathbf{y}_{k_i-1})\mathbf{d}_{k_i-1} + \mathbf{g}_{\phi_{k_i-1}} = \mathbf{0}$. Hence

$$\|\mathbf{g}_{\phi_{k_i-1}}\| = \|H_\phi(\mathbf{y}_{k_i-1})\mathbf{d}_{k_i-1}\| \le \|H_\phi(\mathbf{y}_{k_i-1})\|\|\mathbf{d}_{k_i-1}\|$$

So, $\|\mathbf{d}_{k_i-1}\| \ge \|\mathbf{g}_{\phi_{k_i-1}}\|/\|H_\phi(\mathbf{y}_{k_i-1})\|$ in the case of $\mathbf{d}_{k_i-1} < \Delta_{k_i-1}$. Hence, in general

$$\|\mathbf{d}_{k_i-1}\| \ge \min\{\|\mathbf{g}_{\phi_{k_i-1}}\|/\|H_\phi(\mathbf{y}_{k_i-1})\|, \Delta_{k_i-1}\}.$$

Now, it follows from (3.6), (3.7) and $\|\mathbf{d}_{k_i-1}\| \le \Delta_{k_i-1}$ that

$$\|\mathbf{d}_{k_i-1}\| = \Delta_{k_i-1}, \tag{3.8}$$

for all large i. Hence

$$\lim_{i\to\infty} \Delta_{k_i-1} = 0. \tag{3.9}$$

Since $\mathbf{Y}_{k_i-1} \subset \mathcal{B}(\mathbf{x}_{k_i-1}, \Delta_{k_i-1})$ is Λ-poised, it follows from $\|\mathbf{g}_{\phi_{k_i-1}}\| \ge \delta_1 > 0$ and Lemma 3.3 that there exists a constant ϵ_{δ_1} such that if $\Delta_{k_i-1} \le \epsilon_{\delta_1}$, then $|r_{k_i-1} - 1| \le 0.3$. Hence, by (3.9), we have $r_{k_i-1} \ge 0.7$ for all large i. This together with $\|\mathbf{d}_{k_i-1}\| = \Delta_{k_i-1} > \frac{1}{2}\rho_{k_i-1}$ (where we have used 3.5) imply that, for large i, Step 3 and Step 6 will not be involved at the $k_i - 1$-th iteration. Therefore, from the DFLS algorithm, $r_{k_i-1} \ge 0.7$ and (3.8) , we have

$$\Delta_{k_i} = \min\{\max\{\tilde{\Delta}_{k_i},\ \rho_{k_i-1}\}, \Delta_{max}\} \ge \max\{2\|\mathbf{d}_{k_i-1}\|,\ \rho_{k_i-1}\} = 2\Delta_{k_i-1} > \Delta_{k_i-1},$$

for all large i. This contradicts the middle expression in (3.5). □

The norm of the model gradient $\|\mathbf{g}_{\phi_k}\|$ and $\|\nabla\phi(\mathbf{x}_k)\|$ have the following relationship.

Lemma 3.5. *Under Assumptions 2.2 and 2.3, for any subsequence $\{\mathbf{x}_{k_i}\}$ generated by the DFLS algorithm, if*

$$\lim_{i\to\infty} \|\mathbf{g}_{\phi_{k_i}}\| = 0, \tag{3.10}$$

then

$$\lim_{i\to\infty} \|\nabla\Phi(\mathbf{x}_{k_i})\| = 0. \tag{3.11}$$

Proof. By (3.10) and Step 1 (Criticality step) of DFLS algorithm, we know $\mathbf{Y}_{k_i} \subset \mathcal{B}(\mathbf{x}_{k_i}, \bar{\Delta}_{k_i}) \subseteq \mathcal{B}(\mathbf{x}_{k_i}, \beta\|\mathbf{g}_{\phi_{k_i}}\|)$ is Λ-poised, for all large i. So, by Lemma 2.3, Lemma 2.5, (2.11) and (2.8), we have

$$\begin{aligned} \|\nabla\Phi(\mathbf{x}_{k_i}) - \mathbf{g}_{\phi_{k_i}}\| &= \|\nabla\Phi(\mathbf{x}_{k_i}) - J(\mathbf{y}_{k_i})^\mathsf{T}\mathbf{m}(\mathbf{y}_{k_i}) - H_\phi(\mathbf{y}_{k_i})(\mathbf{x}_{k_i} - \mathbf{y}_{k_i})\| \\ &= \|\nabla F(\mathbf{x}_{k_i})^\mathsf{T} F(\mathbf{x}_{k_i}) - J(\mathbf{y}_{k_i})^\mathsf{T} F(\mathbf{y}_{k_i}) - H_\phi(\mathbf{y}_{k_i})(\mathbf{x}_{k_i} - \mathbf{y}_{k_i})\| \\ &\le \|\nabla F(\mathbf{x}_{k_i})^\mathsf{T} F(\mathbf{x}_{k_i}) - \nabla F(\mathbf{y}_{k_i})^\mathsf{T} F(\mathbf{y}_{k_i})\| \\ &\quad + \|\nabla F(\mathbf{y}_{k_i})^\mathsf{T} F(\mathbf{y}_{k_i}) - J(\mathbf{y}_{k_i})^\mathsf{T} F(\mathbf{y}_{k_i})\| + \kappa_H^\phi\|\mathbf{x}_{k_i} - \mathbf{y}_{k_i}\| \\ &\le L_0\|\nabla F(\mathbf{y}_{k_i}) - J(\mathbf{y}_{k_i})\| + c_1\|\mathbf{x}_{k_i} - \mathbf{y}_{k_i}\| \\ &\le c_2\|\mathbf{x}_{k_i} - \mathbf{y}_{k_i}\| \le c_2\beta\|\mathbf{g}_{\phi_{k_i}}\|, \end{aligned}$$

with c_1 and c_2 properly chosen constants. Hence (3.11) holds. □

Now we have the following global convergence theorem.

Theorem 3.6. *Under Assumptions 2.2 and 2.3, the sequence* $\{\mathbf{x}_k\}$ *generated by the DFLS algorithm satisfies*

$$\liminf_{k\to\infty} \|\nabla\Phi(\mathbf{x}_k)\| = \liminf_{k\to\infty} \|\nabla F(\mathbf{x}_k)^{\mathsf{T}} F(\mathbf{x}_k)\| = 0.$$

Proof. First, we prove that there exists a subsequence $\{\mathbf{x}_{k_i}\}$ generated by the DFLS algorithm such that (3.10) holds. Now suppose (3.10) does not hold, i.e. there exists a $\delta_1 > 0$ such that

$$\|\mathbf{g}_{\phi_k}\| > \delta_1, \tag{3.12}$$

for all sufficiently large k. Hence, by Lemma 3.4, there exists a constant $\delta_2 > 0$ such that

$$\Delta_k \geq \delta_2. \tag{3.13}$$

Now define

$$\mathcal{I} = \{k : r_k \geq 0.1\}.$$

Then, since $\Phi(\mathbf{x})$ is bounded below by zero, it follows from the DFLS algorithm, Lemma 2.5, Lemma 3.1, and (3.12) that

$$\begin{aligned} +\infty &> \sum_{k=0}^{\infty}[\Phi(\mathbf{x}_k) - \Phi(\mathbf{x}_{k+1})] \\ &\geq \sum_{k\in\mathcal{I}}[\Phi(\mathbf{x}_k) - \Phi(\mathbf{x}_{k+1})] \\ &\geq \sum_{k\in\mathcal{I}} 0.1\ Pred_k \geq \sum_{k\in\mathcal{I}} \frac{0.1}{2}\ \delta_1 \min\{\Delta_k, \delta_1/\kappa_H^{\phi}\}. \end{aligned}$$

Hence, if $|\mathcal{I}| = \infty$, then

$$\lim_{k\in\mathcal{I}, k\to\infty} \Delta_k = 0. \tag{3.14}$$

On the other hand, if $|\mathcal{I}| < \infty$, i.e. $r_k < 0.1$ for all sufficiently large k, and then using the mechanisms of Step 3, Step 5 and Step 6, and taking into consideration that $\mathbf{Y}_{k+1}$ will be Λ-poised whenever Step 3 or Step 6 is invoked and Δ_k is not reduced and Δ_k is reduced in Steps 3 and 6 whenever $\mathbf{Y}_k$ is Λ-poised, it follows from the hypothesis that $|\mathcal{I}| < \infty$, that Δ_k cannot be increased for sufficiently large k and we can conclude that

$$\lim_{k\to\infty} \Delta_k = 0. \tag{3.15}$$

Either (3.14) or (3.15) contradicts (3.13). Hence, (3.10) holds. Thus, by Lemma 3.5, (3.11) holds and the theorem is proved. □

4. Local Convergence. In this section, we would like to discuss local convergence properties of the DFLS algorithm for zero residue problems under a certain type of non-singularity condition. Consequently, in this section we have the following additional assumptions on the objective function (1.1).

ASSUMPTION 4.1.
(I) We assume (1.1) is a zero residue problem, i.e.

$$\Phi(\mathbf{x}^*) = 0, \qquad \textit{for any } \mathbf{x}^* \in \mathbf{X}^*,$$

where $\mathbf{X}^* \in \mathbb{R}^n$ is the solution set of (1.1).
(II) $\|F(\mathbf{x})\|$ provides a local error bound on $\mathbf{X}^*$, i.e. there exist positive constants α and ρ such that

$$\|F(\mathbf{x})\| \geq \alpha \; dist(\mathbf{x}, \mathbf{X}^*), \qquad \textit{whenever} \quad dist(\mathbf{x}, \mathbf{X}^*) \leq \rho, \tag{4.1}$$

where $dist(\mathbf{x}, \mathbf{X}^*) = \inf_{\mathbf{y} \in \mathbf{X}^*} \|\mathbf{x} - \mathbf{y}\|$.

Since $\Phi(\mathbf{x})$ is continuous, we know $\mathbf{X}^*$ is closed. Hence, it follows from $\mathbf{X}^*$ being nonempty that for any $\mathbf{x} \in \mathbb{R}^n$, there exists a $\bar{\mathbf{x}} \in \mathbf{X}^*$ such that

$$dist(\mathbf{x}, \mathbf{X}^*) = \min_{\mathbf{y} \in \mathbf{X}^*} \|\mathbf{x} - \mathbf{y}\| = \|\bar{\mathbf{x}} - \mathbf{x}\|. \tag{4.2}$$

In Assumption 4.1, the local error bound condition (II), first proposed in [25, 26], is a certain generalization of the nonsingularity assumption of the Jacobian at the solution, as is readily seen by linearising F about x^*. Similar conditions have subsequently been widely used and studied in [11, 12, 14, 15, 27]. When $\nabla F(\mathbf{x}^*)$ is nonsingular at a solution $\mathbf{x}^*$, $\mathbf{x}^*$ is an isolated solution. Hence, $\|F(\mathbf{x})\|$ provides a local error bound at $\mathbf{x}^*$. However, $\nabla F(\mathbf{x}^*)$ might be singular but nevertheless $\|F(\mathbf{x})\|$ may provide a local error bound at $\mathbf{x}^*$. One can refer to the examples provided in [8] and [27]. It is well-known that the Levenberg-Marguardt method has local quadratic convergence when the Jacobian at $\mathbf{x}^*$ is nonsingular. In the following we will first show, under proper conditions and assumptions, that a certain class of DFLS algorithms will eventually reduce to a "regularized Levenberg-Marguardt-type" algorithm and then we can show the local convergence rate is quadratic, even without the nonsingularity assumption of the Jacobian at the solution.

For different choices of the base point $\mathbf{y}_k$ in the DFLS algorithm, we have different algorithms. In this section, with respect to local convergence properties, we consider a subclass of DFLS algorithms which makes a specific choice for the base points. Hence, in this section we make the following assumption on the choice of base points.

ASSUMPTION 4.2. *For sufficiently large k, we choose the base point $\mathbf{y}_k \in \mathbf{Y}_k$ to be $\mathbf{x}_k$, the current iterate (which corresponds to the lowest objective function value seen to date) i.e.*

$$\mathbf{y}_k = \mathbf{x}_k,$$

for k sufficiently large .

To discuss the local convergence properties of the DFLS algorithm, we first assume that the sequence $\{\mathbf{x}_k\}$ generated by the algorithm converges to the solution set $\mathbf{X}^*$. Then, we show it converges to a solution $\mathbf{x}^* \in \mathbf{X}^*$ quadratically. Thus, we have the following assumption.

ASSUMPTION 4.3. *Assume the sequence of $\{\mathbf{x}_k\}$ generated by DFLS algorithm converges to the solution set $\mathbf{X}^*$, i.e.*

$$\lim_{k \to \infty} dist(\mathbf{x}_k, \mathbf{X}^*) = 0.$$

We begin the discussion with the following lemma.

LEMMA 4.1. *Under Assumptions 2.2, 2.3, 4.1, 4.2, and 4.3, we have that*

$$\lim_{k\to\infty} \|\mathbf{g}_{\phi_k}\| = 0, \tag{4.3}$$

and there exists a constant $L_3 > 0$ such that

$$\begin{aligned} \|J(\mathbf{x}_k) - \nabla F(\mathbf{x}_k)\| &= \|\nabla \mathbf{m}(\mathbf{x}_k) - \nabla F(\mathbf{x}_k)\| \\ &\le \bar{M}\|\mathbf{g}_\phi(\mathbf{x}_k)\| \le L_3\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|, \end{aligned} \tag{4.4}$$

for all large k, where $\bar{M} := m\beta\kappa_{eg}$, with κ_{eg} and β defined in (2.8) and Step 0 of the algorithm, respectively. Here and in what follows $\bar{\mathbf{x}}_k$ is as defined in (4.2).

Proof. By Assumptions 4.1 and 4.3, we have

$$\lim_{k\to\infty} \|F(\mathbf{x}_k)\| = 0. \tag{4.5}$$

Since $\mathbf{y}_k = \mathbf{x}_k$ for all sufficiently large k by Assumption 4.2, it follows from Lemma 2.5 that

$$\|\mathbf{g}_{\phi_k}\| = \|\mathbf{g}_\phi(\mathbf{x}_k)\| = \|J(\mathbf{x}_k)^\mathsf{T}\mathbf{m}(\mathbf{x}_k)\| \le m\kappa_g\|F(\mathbf{x}_k)\|, \tag{4.6}$$

Hence, by (4.5), we have (4.3) holds.

Now, by Assumption (4.3), it follows from Step 1 (Criticality step) of the DFLS algorithm that $\mathbf{Y}_k$ is Λ-poised in $\mathcal{B}(\mathbf{x}_k, \beta\|\mathbf{g}_{\phi_k}\|)$. Since $\mathbf{g}_{\phi_k} = \mathbf{g}_\phi(\mathbf{x}_k) = J(\mathbf{x}_k)^\mathsf{T}\mathbf{m}(\mathbf{x}_k)$, by (2.8), (4.6) and Lemma 2.3, we have

$$\begin{aligned} \|J(\mathbf{x}_k) - \nabla F(\mathbf{x}_k)\| &= \|\nabla \mathbf{m}(\mathbf{x}_k) - \nabla F(\mathbf{x}_k)\| \\ &\le \sum_{i=1}^{m} \|\nabla m_i(\mathbf{x}_k) - \nabla f_i(\mathbf{x}_k)\| \\ &\le m\kappa_{eg}\beta\|\mathbf{g}_{\phi_k}\| \le m^2\kappa_{eg}\beta\kappa_g\|F(\mathbf{x}_k)\| \\ &\le m^2\kappa_{eg}\beta\kappa_g L_1\|\mathbf{x}_k - \bar{\mathbf{x}}_k\| := L_3\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|. \end{aligned}$$

where $L_3 = m^2\kappa_{eg}\beta\kappa_g L_1$. □

The following lemma shows that, in the zero residual case, the regularized Hessian will be eventually chosen (i.e. the middle term in the definition of H_ϕ) by the DFLS algorithm for building the trust region subproblem (2.12).

LEMMA 4.2. *Under Assumptions 2.2, 2.3, 4.1, 4.2, and 4.3, for any $\kappa_H^2 > 0$, if k is sufficiently large, then*

$$c_\phi(\mathbf{x}_k) \le \kappa_H^2\|\mathbf{g}_\phi(\mathbf{x}_k)\|, \tag{4.7}$$

and

$$\begin{aligned} \phi_k(\mathbf{d}) = \phi(\mathbf{x}_k, \mathbf{d}) &= \frac{1}{2}\|\mathbf{m}(\mathbf{x}_k) + J(\mathbf{x}_k)\,\mathbf{d}\|^2 + \lambda_k\|\mathbf{d}\|^2 \\ &= \frac{1}{2}\|\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\,\mathbf{d}\|^2 + \lambda_k\|\mathbf{d}\|^2, \end{aligned} \tag{4.8}$$

where $\lambda_k = \frac{1}{2}\kappa_H^3\|\mathbf{m}(\mathbf{x}_k)\|$, κ_H^2 and κ_H^3 are given in the definition of H_ϕ and $\phi(\cdot,\cdot)$ is defined by (2.10).

Proof. First, we assume k is large enough that Lemma 4.1 holds and $\|\mathbf{x}_k - \bar{\mathbf{x}}_k\| = \delta$ is sufficiently small.

Since $\|F(\mathbf{x})\|$ provides a local error bound on $\mathbf{X}^*$, it follows from (4.1) that for sufficiently large k, or equivalently δ sufficiently small, $\|F(\mathbf{x}_k)\| \geq \alpha \ \|\mathbf{x}_k - \bar{\mathbf{x}}_k\|$ for some $\alpha > 0$. Therefore, noticing $\Phi(\bar{\mathbf{x}}_k) = 0$ and $\nabla\Phi(\bar{\mathbf{x}}_k) = \mathbf{0}$, we have

$$\begin{aligned}\frac{1}{2}\alpha^2 \ \|\mathbf{x}_k - \bar{\mathbf{x}}_k\|^2 \leq \frac{1}{2}\|F(\mathbf{x}_k)\|^2 &= \Phi(\mathbf{x}_k) \\ &= \frac{1}{2}(\mathbf{x}_k - \bar{\mathbf{x}}_k)^{\mathsf{T}}\bar{H}(\mathbf{x}_k - \bar{\mathbf{x}}_k) + R_2(\mathbf{x}_k, \bar{\mathbf{x}}_k), \end{aligned} \tag{4.9}$$

where $\bar{H} = \nabla^2\Phi(\bar{\mathbf{x}}_k)$ is the Hessian at $\bar{\mathbf{x}}_k$ and R_2 is the remainder term. By choosing δ smaller if necessary, we know

$$|R_2(\mathbf{x}_k, \bar{\mathbf{x}}_k)| \leq \frac{\alpha^2}{3}\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|^2.$$

So, in this case, (4.9) gives

$$\frac{\alpha^2}{3}\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|^2 \leq (\mathbf{x}_k - \bar{\mathbf{x}}_k)^{\mathsf{T}}\bar{H}(\mathbf{x}_k - \bar{\mathbf{x}}_k) \leq \|\mathbf{x}_k - \bar{\mathbf{x}}_k\|\|\bar{H}(\mathbf{x}_k - \bar{\mathbf{x}}_k)\|.$$

Hence, we have

$$\frac{\alpha^2}{3}\|\mathbf{x}_k - \bar{\mathbf{x}}_k\| \leq \|\bar{H}(\mathbf{x}_k - \bar{\mathbf{x}}_k)\|. \tag{4.10}$$

On the other hand,

$$\nabla\Phi(\mathbf{x}_k) = \nabla\Phi(\mathbf{x}_k) - \nabla\Phi(\bar{\mathbf{x}}_k) = \bar{H}(\mathbf{x}_k - \bar{\mathbf{x}}_k) + R_1(\mathbf{x}_k, \bar{\mathbf{x}}_k), \tag{4.11}$$

where R_1 is the remainder term. Choose k sufficiently large if necessary so that

$$|R_1(\mathbf{x}_k, \bar{\mathbf{x}}_k)| \leq \frac{\alpha^2}{6}\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|. \tag{4.12}$$

combining (4.10)-(4.12), we have

$$\|\nabla F(\mathbf{x}_k)^{\mathsf{T}}F(\mathbf{x}_k)\| = \|\nabla\Phi(\mathbf{x}_k)\| \geq \frac{\alpha^2}{6}\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|. \tag{4.13}$$

Thus, by Assumption 2.2, Lemma 2.3 and (4.13), for δ small,

$$\begin{aligned}\frac{c_\phi(\mathbf{x}_k)}{\|\nabla F(\mathbf{x}_k)^{\mathsf{T}}F(\mathbf{x}_k)\|} &= \frac{1}{2}\frac{\mathbf{m}(\mathbf{x}_k)^{\mathsf{T}}\mathbf{m}(\mathbf{x}_k)}{\|\nabla F(\mathbf{x}_k)^{\mathsf{T}}F(\mathbf{x}_k)\|} \\ &= \frac{1}{2}\frac{F(\mathbf{x}_k)^{\mathsf{T}}F(\mathbf{x}_k)}{\|\nabla F(\mathbf{x}_k)^{\mathsf{T}}F(\mathbf{x}_k)\|} \\ &\leq \frac{L_1^2\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|^2}{2\|\nabla F(\mathbf{x}_k)^{\mathsf{T}}F(\mathbf{x}_k)\|} \\ &\leq \frac{3L_1^2}{\alpha^2} \ \|\mathbf{x}_k - \bar{\mathbf{x}}_k\|. \end{aligned} \tag{4.14}$$

Now, by (4.4), we have

$$\begin{aligned}\frac{\|\nabla F(\mathbf{x}_k)^\mathsf{T} F(\mathbf{x}_k) - J(\mathbf{x_k})^\mathsf{T}\mathbf{m}(\mathbf{x}_k)\|}{\|J(\mathbf{x_k})^\mathsf{T}\mathbf{m}(\mathbf{x}_k)\|} &= \frac{\|(\nabla F(\mathbf{x}_k) - J(\mathbf{x_k}))^\mathsf{T} F(\mathbf{x}_k)\|}{\|J(\mathbf{x_k})^\mathsf{T}\mathbf{m}(\mathbf{x}_k)\|} \\ &\le \frac{\bar{M}\|\mathbf{g}_\phi(\mathbf{x}_k)\|\|F(\mathbf{x}_k)\|}{\|\mathbf{g}_\phi(\mathbf{x}_k)\|} \\ &= \bar{M}\|F(\mathbf{x}_k)\| \le \bar{M}L_1\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|,\end{aligned}$$

where $\bar{M}$ is the constant defined in (4.4). Hence,

$$1 - \bar{M}L_1\|\mathbf{x}_k - \bar{\mathbf{x}}_k\| \le \frac{\|\nabla F(\mathbf{x}_k)^\mathsf{T} F(\mathbf{x}_k)\|}{\|J(\mathbf{x_k})^\mathsf{T}\mathbf{m}(\mathbf{x}_k)\|} \le 1 + \bar{M}L_1\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|. \tag{4.15}$$

Therefore, by (4.14) and (4.15), for δ small, we have

$$c_\phi(\mathbf{x}_k) \le c\,\delta\|\mathbf{g}_\phi(\mathbf{x}_k)\|, \tag{4.16}$$

Where $c > 0$ is some constant. Hence, For any $\kappa_H^2 > 0$, if k is sufficiently large, then δ will be sufficiently small, and therefore (4.7) holds. Noticing, $\mathbf{g}_{\phi_k} = \mathbf{g}_\phi(\mathbf{x}_k)$ and $\|\mathbf{g}_{\phi_k}\| \le \kappa_H^1$ for all large k, by (4.7) and the definition of $\phi(\cdot,\cdot)$, (4.8) follows immediately. □

Now, for the trust region step size, we have the following lemma.

Lemma 4.3. *Under Assumptions 2.2, 2.3, 4.1, 4.2, and 4.3, we have*

$$\|\mathbf{d}_k\| \le 2dist(\mathbf{x}_k, \mathbf{X}^*) = 2\|\bar{\mathbf{x}}_k - \mathbf{x}_k\|, \tag{4.17}$$

for k sufficiently large.

Proof. First, we assume k is large enough that Lemma 4.1 and Lemma 4.2 hold, and $\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|$ is sufficiently small.

We only need to consider the case $\|\bar{\mathbf{x}}_k - \mathbf{x}_k\| \le \|\mathbf{d}_k\|$. In this case, $\bar{\mathbf{x}}_k - \mathbf{x}_k$ is a feasible point of the trust region subproblem (2.12). Hence, denoting $\mathbf{s}_k = \bar{\mathbf{x}}_k - \mathbf{x}_k$, by Lemma 2.3, (4.1), (4.4) and (4.8), choosing δ small if necessary, we have

$$\begin{aligned}\|\mathbf{d}_k\|^2 &\le \frac{2}{\kappa_H^3\|\mathbf{m}(\mathbf{x}_k)\|}\phi_k(\bar{\mathbf{x}}_k - \mathbf{x}_k) \\ &= \frac{1}{\kappa_H^3\|F(\mathbf{x}_k)\|}\|F(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\,\mathbf{s}_k\|^2 + \|\mathbf{s}_k\|^2 \\ &\le \frac{1}{\kappa_H^3\|F(\mathbf{x}_k)\|}(\|F(\mathbf{x}_k) + \nabla F(\mathbf{x}_k)\,\mathbf{s}_k\| + \|(\nabla\mathbf{m}(\mathbf{x}_k) - \nabla F(\mathbf{x}_k))\,\mathbf{s}_k\|)^2 + \|\mathbf{s}_k\|^2 \\ &\le \frac{1}{\kappa_H^3\|F(\mathbf{x}_k)\|}(\frac{m}{2}L_2\|\mathbf{s}_k\|^2 + L_3\|\mathbf{s}_k\|^2)^2 + \|\mathbf{s}_k\|^2 \\ &\le \frac{1}{\kappa_H^3\alpha\|\mathbf{s}_k\|}(\frac{m}{2}L_2\|\mathbf{s}_k\|^2 + L_3\|\mathbf{s}_k\|^2)^2 + \|\mathbf{s}_k\|^2 \\ &= \frac{(\frac{m}{2}L_2 + L_3)^2}{\kappa_H^3\alpha}\|\mathbf{s}_k\|^3 + \|\mathbf{s}_k\|^2 \\ &\le 2\|\mathbf{s}_k\|^2 = 2\|\bar{\mathbf{x}}_k - \mathbf{x}_k\|^2,\end{aligned}$$

[Note that in order to determine the third inequality we have used $0 = f_i(\bar{\mathbf{x}}_k) = f_i(x_k) + \nabla f_i(x_k)^\top s_k + \frac{1}{2}s_k^\top\nabla^2 f_i(\xi_k)s_k$ and thus $\|f_i(x_k) + \nabla f_i(x_k)^\top s_k\| \le \frac{1}{2}L_2\|s_k\|^2$]
□

Lemma 4.4. *Under Assumptions 2.2, 2.3, 4.1, 4.2, and 4.3, for any $\epsilon > 0$, if k is sufficiently large, then*

$$|r_k - 1| \le \epsilon. \tag{4.18}$$

In addition, there exists a constant $\bar{\Delta} > 0$ such that

$$\Delta_k \ge \bar{\Delta}, \qquad \textit{for all } k \ge 0. \tag{4.19}$$

Proof. First, we assume k is large enough that Lemma 4.1, Lemma 4.2 and Lemma 4.3 hold, and $\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|$ is sufficiently small. In the following proof, we use $\mathbf{s}_k$ to denote $\bar{\mathbf{x}}_k - \mathbf{x}_k$.

Now, we first prove

$$\|\mathbf{m}(\mathbf{x}_k)\| - \|\mathbf{m}(\mathbf{x}_k) + \nabla \mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\| \ge c_3 \|\mathbf{d}_k\|, \tag{4.20}$$

when k is sufficiently large, where $c_3 > 0$ is a constant.

Case I, $\|\mathbf{s}_k\| \le \|\mathbf{d}_k\|$. In this case, $\mathbf{s}_k$ is a feasible point of (2.12), then it follows from (4.1), (4.4) and (4.17) and $\|\mathbf{s}_k\|$ sufficiently small that

$$\begin{aligned}
\|\mathbf{m}(\mathbf{x}_k)\| - \|\mathbf{m}(\mathbf{x}_k) + \nabla \mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\| &\ge \|\mathbf{m}(\mathbf{x}_k)\| - \|\mathbf{m}(\mathbf{x}_k) + \nabla \mathbf{m}(\mathbf{x}_k)\ \mathbf{s}_k\| \\
&\ge \|\mathbf{m}(\mathbf{x}_k)\| - \|\mathbf{m}(\mathbf{x}_k) + \nabla F(\mathbf{x}_k)\ \mathbf{s}_k\| \\
&\quad - \|(\nabla \mathbf{m}(\mathbf{x}_k) - \nabla F(\mathbf{x}_k))\ \mathbf{s}_k\| \\
&\ge \|\mathbf{m}(\mathbf{x}_k)\| - \frac{m}{2} L_2 \|\mathbf{s}_k\|^2 - L_3 \|\mathbf{s}_k\|^2 \\
&\ge \alpha \|\mathbf{s}_k\| - (\frac{m}{2} L_2 + L_3) \|\mathbf{s}_k\|^2 \\
&\ge \frac{\alpha}{2} \|\mathbf{s}_k\| \ge \frac{\alpha}{4} \|\mathbf{d}_k\|,
\end{aligned}$$

where the first inequality came from the fact for k large, the trust region subproblem has the form (2.12), $\mathbf{d}_k$ is a solution of this subproblem, (4.8) holds, and $\|\mathbf{s}_k\| \le \|\mathbf{d}_k\|$ is feasible. Thus we have

$$\frac{1}{2} \|\mathbf{m}(\mathbf{x}_k) + \nabla \mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\|^2 + \lambda_k \|\mathbf{d}_k\|^2 \le \frac{1}{2} \|\mathbf{m}(\mathbf{x}_k) + \nabla \mathbf{m}(\mathbf{x}_k)\ \mathbf{s}_k\|^2 + \lambda_k \|\mathbf{s}_k\|^2.$$

So, it follows from $\|\mathbf{s}_k\| \le \|\mathbf{d}_k\|$ that $\|\mathbf{m}(\mathbf{x}_k) + \nabla \mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\| \le \|\mathbf{m}(\mathbf{x}_k) + \nabla \mathbf{m}(\mathbf{x}_k)\ \mathbf{s}_k\|$. The third inequality again used the note in the proof of Lemma 4.3 and the last inequality used (4.17).

Case II, $\|\mathbf{s}_k\| > \|\mathbf{d}_k\|$. In this case, $\mathbf{t}_k := (\|\mathbf{d}_k\| / \|\mathbf{s}_k\|)\ \mathbf{s}_k$ is a feasible point of (2.12), then by (4.1) and (4.4), we have

$$\begin{aligned}
\|\mathbf{m}(\mathbf{x}_k)\| - \|\mathbf{m}(\mathbf{x}_k) + \nabla \mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\| &\ge \|\mathbf{m}(\mathbf{x}_k)\| - \|\mathbf{m}(\mathbf{x}_k) + \nabla \mathbf{m}(\mathbf{x}_k)\ \mathbf{t}_k\| \\
&\ge \frac{\|\mathbf{d}_k\|}{\|\mathbf{s}_k\|} (\|F(\mathbf{x}_k)\| - \|\mathbf{m}(\mathbf{x}_k) + \nabla \mathbf{m}(\mathbf{x}_k)\ \mathbf{s}_k\|) \\
&\ge \frac{\|\mathbf{d}_k\|}{\|\mathbf{s}_k\|} (\alpha \|\mathbf{s}_k\| - (\frac{m}{2} L_2 + L_3) \|\mathbf{s}_k\|^2) \\
&\ge \frac{\alpha}{2} \|\mathbf{d}_k\|,
\end{aligned}$$

for k sufficiently large, where we used the same argument as in Case I to establish the penultimate inequality.

From Case I and II, we know (4.20) holds with $c_3 = \alpha/4$.

Now, by (2.14), (4.8), (4.17) and (4.20), we have

$$\begin{aligned}
Pred_k &= \phi_k(\mathbf{0}) - \phi_k(\mathbf{d}_k)\\
&= \frac{1}{2}(\|\mathbf{m}(\mathbf{x}_k)\|^2 - \|\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\|^2 - \kappa_H^3\|\mathbf{m}(\mathbf{x}_k)\|\|\mathbf{d}_k\|^2)\\
&\geq \frac{1}{2}\|F(\mathbf{x}_k)\|(\|F(\mathbf{x}_k)\| - \|F(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\| - \kappa_H^3\|\mathbf{d}_k\|^2)\\
&\geq \frac{1}{2}\|F(\mathbf{x}_k)\|(\frac{\alpha}{4}\|\mathbf{d}_k\| - \kappa_H^3\|\mathbf{d}_k\|^2)\\
&\geq \frac{\alpha}{16}\|F(\mathbf{x}_k)\|\|\mathbf{d}_k\|, \qquad (4.21)
\end{aligned}$$

for k sufficiently large. Note that to obtain the first inequality we used that $\|F(\mathbf{x_k})\| \geq \|\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\mathbf{d}_k\|$, from (4.20).

By Lemma 4.4, we have

$$\begin{aligned}
&\|(\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k) - (F(\mathbf{x}_k) + \nabla F(\mathbf{x}_k)\ \mathbf{d}_k)\|\\
&\leq \|\nabla\mathbf{m}(\mathbf{x}_k) - \nabla F(\mathbf{x}_k)\|\|\mathbf{d}_k\| \leq L_3\|\mathbf{s}_k\|\|\mathbf{d}_k\|.
\end{aligned}$$

By Lemma 2.3, we have

$$\|F(\mathbf{x}_k + \mathbf{d}_k) - (F(\mathbf{x}_k) + \nabla F(\mathbf{x}_k)\ \mathbf{d}_k)\| \leq \frac{L_2\sqrt{m}}{2}\|\mathbf{d}_k\|^2.$$

Hence, we get

$$\begin{aligned}
&\|(\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k) + (F(\mathbf{x}_k) + \nabla F(\mathbf{x}_k)\ \mathbf{d}_k)\|\\
&\leq 2\|\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\| + L_3\|\mathbf{s}_k\|\|\mathbf{d}_k\|,
\end{aligned}$$

and

$$\begin{aligned}
&\|F(\mathbf{x}_k + \mathbf{d}_k) + (F(\mathbf{x}_k) + \nabla F(\mathbf{x}_k)\ \mathbf{d}_k)\|\\
&\leq 2\|F(\mathbf{x}_k) + \nabla F(\mathbf{x}_k)\ \mathbf{d}_k\| + \frac{L_2\sqrt{m}}{2}\|\mathbf{d}_k\|^2\\
&\leq 2\|\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\| + 2L_3\|\mathbf{s}_k\|\|\mathbf{d}_k\| + \frac{L_2\sqrt{m}}{2}\|\mathbf{d}_k\|^2.
\end{aligned}$$

Therefore, by $\|\mathbf{d}_k\| \leq 2\|\mathbf{s}_k\|$,

$$\begin{aligned}
&\big|\|\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\|^2 - \|F(\mathbf{x}_k + \mathbf{d}_k)\|^2\big|\\
&\leq \big|\|\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\|^2 - \|F(\mathbf{x}_k) + \nabla F(\mathbf{x}_k)\ \mathbf{d}_k\|^2\big|\\
&\quad + \big|\|F(\mathbf{x}_k) + \nabla F(\mathbf{x}_k)\ \mathbf{d}_k\|^2 - \|F(\mathbf{x}_k + \mathbf{d}_k)\|^2\big|\\
&\leq (2\|\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\| + L_3\|\mathbf{s}_k\|\|\mathbf{d}_k\|)\ (L_3\|\mathbf{s}_k\|\|\mathbf{d}_k\|)\\
&\quad + \left(2\|\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\| + 2L_3\|\mathbf{s}_k\|\|\mathbf{d}_k\| + \frac{L_2\sqrt{m}}{2}\|\mathbf{d}_k\|^2\right)\ \left(\frac{L_2\sqrt{m}}{2}\|\mathbf{d}_k\|^2\right)\\
&= (2L_3\|\mathbf{s}_k\| + L_2\sqrt{m}\|\mathbf{d}_k\|)\ \|\mathbf{d}_k\|\ \|\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\|\\
&\quad +(L_3^2\|\mathbf{s}_k\|\|\mathbf{d}_k\| + L_2L_3\sqrt{m}\|\mathbf{d}_k\|^2)\ \|\mathbf{s}_k\|\ \|\mathbf{d}_k\| + \frac{L_2^2 m}{4}\|\mathbf{d}_k\|^4.
\end{aligned}$$

Hence, again using $\|F(\mathbf{x_k})\| \geq \|\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\mathbf{d}_k\|$ and the above inequality, we have

$$\begin{aligned}|Ared_k - Pred_k| &= \frac{1}{2}\left|\|\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\ \mathbf{d}_k\|^2 - \|F(\mathbf{x_k} + \mathbf{d}_k)\|^2 + \kappa_H^3\|\mathbf{m}(\mathbf{x}_k)\|\|\mathbf{d}_k\|^2\right| \\ &\leq (2L_3\|\mathbf{s}_k\| + (L_2\sqrt{m} + \kappa_H^3)\|\mathbf{d}_k\|)\ \|F(\mathbf{x}_k)\|\ \|\mathbf{d}_k\| \\ &\quad + (L_3^2\|\mathbf{s}_k\|\|\mathbf{d}_k\| + L_2L_3\sqrt{m}\|\mathbf{d}_k\|^2)\ \|\mathbf{s}_k\|\ \|\mathbf{d}_k\| + \frac{L_2^2 m}{4}\|\mathbf{d}_k\|^4 \qquad (4.22)\end{aligned}$$

for k sufficiently large. Hence, by (4.21) and (4.22), we have

$$\begin{aligned}|r_k - 1| \leq \left|\frac{Ared_k - Pred_k}{Pred_k}\right| &\leq \frac{32L_3}{\alpha}\|\mathbf{s}_k\| + \frac{16(L_2\sqrt{m} + \kappa_H^3)}{\alpha}\|\mathbf{d}_k\| \\ &+ \frac{16(L_3^2\|\mathbf{s}_k\|\|\mathbf{d}_k\| + L_2L_3\sqrt{m}\|\mathbf{d}_k\|^2)}{\alpha}\ \frac{\|\mathbf{s}_k\|}{\|F(\mathbf{x}_k)\|} + \frac{4L_2^2 m}{\alpha}\frac{\|\mathbf{d}_k\|^3}{\|F(\mathbf{x}_k)\|}\end{aligned}$$

The above inequality together with (4.1) and (4.17), implies (4.18) holds when k is sufficiently large. Hence, if we choose $\epsilon = 0.3$ in (4.18), by (4.3) and (4.18), Step 3 and Step 6 in the DFLS algorithm will not be invoked for large k. Therefore, from the details of the DFLS algorithm and $r_k \geq 0.7$ for all large k, (4.19) holds. □

The lemma we just proved states that the trust region radius will be bounded away from zero by a positive constant and the model will be a very good one. We can see this essentially implies that the trust region constraint in the DFLS algorithm is eventually inactive. Hence, the algorithm will ultimately reduce to a "regularized Levenberg-Marquardt-type" method. Using a similar approach to that proposed in [12] we are able to show that the iterates generated by our algorithm converge to a local solution quadratically. To begin with we show that the iterates converge to a solution superlinearly.

THEOREM 4.5. *Under Assumptions 2.2, 2.3, 4.1, 4.2, and 4.3, the sequence* $\{\mathbf{x}_k\}$ *generated by the DFLS algorithm converges to a point* $\mathbf{x}^* \in \mathbf{X}^*$ *superlinearly.*

Proof. Firstly, we choose $\bar{k}$ sufficiently large that Lemma 4.1, Lemma 4.3, Lemma 4.4 and the local error bound condition (4.1) hold, for all $k \geq \bar{k}$.

Since $\|\mathbf{x}_k - \bar{\mathbf{x}}_k\| = 0$ as $k \to \infty$, by Assumption 4.3, it follows from (4.19) that there exists an integer, denoted as $\bar{k}$, such that

$$\|\mathbf{x}_k - \bar{\mathbf{x}}_k\| \leq \Delta_k, \qquad \text{for all } k \geq \bar{k}.$$

Hence, denoting $\mathbf{s}_k = \bar{\mathbf{x}}_k - \mathbf{x}_k$, for all $k \geq \bar{k}$ large enough, we have

$$\begin{aligned}\phi_k(\mathbf{d}_k) &\leq \phi_k(\mathbf{s}_k) \leq \|\mathbf{m}(\mathbf{x}_k) + \nabla\mathbf{m}(\mathbf{x}_k)\mathbf{s}_k\|^2 + \frac{\kappa_H^3}{2}\|\mathbf{m}(\mathbf{x}_k)\|\|\mathbf{s}_k\|^2 \\ &\leq (\|F(\mathbf{x}_k) + \nabla F(\mathbf{x}_k)\mathbf{s}_k\| + L_3\|\mathbf{s}_k\|^2)^2 + \frac{\kappa_H^3}{2}\|F(\mathbf{x}_k)\|\|\mathbf{s}_k\|^2 \text{ (using (4.4))} \\ &\leq (\frac{m}{2}L_2\|\mathbf{s}_k\|^2 + L_3\|\mathbf{s}_k\|^2)^2 + \frac{\kappa_H^3}{2}\|F(\mathbf{x}_k)\|\|\mathbf{s}_k\|^2 \text{ (using the fifth inequality in Lemma 2.3)} \\ &\leq \kappa_H^3 L_1\|\mathbf{s}_k\|^3 \text{ (using the second inequality in Lemma 2.3).}\end{aligned}$$

From the above inequality, Lemma 2.3) and (4.17), for $k \geq \bar{k}$ large,

$$\begin{aligned} \|F(\mathbf{x}_{k+1})\| = \|F(\mathbf{x}_k + \mathbf{d}_k)\| &\leq \|F(\mathbf{x}_k) + \nabla F(\mathbf{x}_k)\mathbf{d}_k\| + \frac{\sqrt{m}}{2} L_2 \|\mathbf{d}_k\|^2 \\ &\leq \sqrt{\phi_k(\mathbf{d}_k)} + \frac{\sqrt{m}}{2} L_2 \|\mathbf{d}_k\|^2 \\ &\leq 2\sqrt{\kappa_H^3 L_1} \|\mathbf{s}_k\|^{3/2}. \end{aligned}$$

Hence, by the above inequality and (4.1), we have

$$\alpha \|\mathbf{x}_{k+1} - \bar{\mathbf{x}}_{k+1}\| \leq \|F(\mathbf{x}_{k+1})\| \leq 2\sqrt{\kappa_H^3 L_1} \|\mathbf{x}_k - \bar{\mathbf{x}}_k\|^{3/2}. \tag{4.23}$$

for all large k. Consequently,

$$\sum_{k=0}^{\infty} \|\mathbf{x}_k - \bar{\mathbf{x}}_k\| \leq \infty.$$

Since (4.17) holds for all sufficiently large k, it follows from the above inequality that

$$\sum_{k=0}^{\infty} \|\mathbf{d}_k\| \leq \infty.$$

Therefore, $\mathbf{x}_k$ converges to some point $\mathbf{x}^* \in \mathbf{X}^*$. In addition, it follows from (4.23) and

$$\|\mathbf{x}_k - \bar{\mathbf{x}}_k\| \leq \|\mathbf{x}_k - \bar{\mathbf{x}}_{k+1}\| \leq \|\mathbf{x}_{k+1} - \bar{\mathbf{x}}_{k+1}\| + \|\mathbf{d}_k\|$$

that $\|\mathbf{x}_k - \bar{\mathbf{x}}_k\| \leq 2\|\mathbf{d}_k\|$ for sufficiently large k. This, together with (4.17) and (4.23), imply that

$$\|\mathbf{d}_{k+1}\| \leq \frac{16\sqrt{\kappa_H^3 L_1}}{\alpha} (\|\mathbf{d}_k\|)^{3/2}.$$

This implies $\mathbf{x}_k$ converges to some point $\mathbf{x}^*$ superlinearly. □

In the following we can apply the same singular value decomposition (SVD) technique as is used in [12] to show that $\mathbf{x}_k$ converges to some point $\mathbf{x}^*$ quadratically. Suppose that the $rank(\nabla F(\mathbf{x}^*)) = r \geq 0$, by the singular value decomposition, we can write $\nabla F(\mathbf{x}^*)$ in the following form with the appropriate dimensions:

$$\nabla F(\mathbf{x}^*) = U^* \Sigma^* (V^*)^{\mathsf{T}},$$

where $\Sigma^* = diag(\sigma_1^*, \sigma_2^*, \ldots, \sigma_r^*)$, $(U^*)^{\mathsf{T}} U^* = I_{r\times r}$, $(V^*)^{\mathsf{T}} V^* = I_{r\times r}$ and $\sigma_1^* \geq \sigma_2^* \geq \ldots \geq \sigma_r^* > 0$. Since $\mathbf{x}_k \to \mathbf{x}^*$, it follows from (4.4) and the continuity properties of the SVD that, for k sufficiently large, we can also write $\nabla \mathbf{m}(\mathbf{x}_k)$ in the following form:

$$\nabla \mathbf{m}(\mathbf{x}_k) = U_k \Sigma_k V_k^{\mathsf{T}}$$

where $U_k = (U_{k,1}, U_{k,2}, U_{k,3})$, $U_k^{\mathsf{T}} U_k = I_{m\times m}$, $V_k = (V_{k,1}, V_{k,2}, V_{k,3})$, $V_k^{\mathsf{T}} V_k = I_{n\times n}$, $\Sigma_k = diag(\Sigma_{k,1}, \Sigma_{k,2}, \Sigma_{k,3})$, $rank(\Sigma_{k,1}) = r$, $rank(\Sigma_{k,2}) = rank(\Sigma_k) - r$ and $\Sigma_{k,3} = 0$, with

$$\lim_{k\to\infty} \Sigma_{k,1} = \Sigma^* \quad \text{and} \quad \lim_{k\to\infty} \Sigma_{k,2} = 0.$$

(Note $\Sigma_{k,2}$ and $\Sigma_{k,3}$ may be empty here.)

Because of (4.4), Lemma 2.3, $\|\mathbf{x}_k - \bar{\mathbf{x}}_k\| \le \|\mathbf{x}_k - \mathbf{x}^*\|$ and the perturbation properties of the SVD, we have that

$$\begin{aligned}\|diag(\Sigma_{k,1} - \Sigma_1^*, \Sigma_{k,2}, 0\| &\le \|\nabla \mathbf{m}(\mathbf{x}_k) - \nabla F(\mathbf{x}^*)\| \\ &\le \|\nabla \mathbf{m}(\mathbf{x}_k) - \nabla F(\mathbf{x}_k)\| + \|\nabla F(\mathbf{x}_k) - \nabla F(\mathbf{x}^*)\| \\ &\le L_3\|\mathbf{x}_k - \bar{\mathbf{x}}_k\| + L_2\|\mathbf{x}_k - \mathbf{x}^*\| \le (L_3 + L_2)\|\mathbf{x}_k - \mathbf{x}^*\|,\end{aligned}$$

for all large k. Then, the following lemma can be obtained directly by applying the technique used in Lemma 2.3 in [12]. We omit restating its proof here.

Lemma 4.6. *Under Assumptions 2.2, 2.3, 4.1, 4.2, and 4.3, for k sufficiently large, we have*

(a) $\|U_{k,1}U_{k,1}^{\mathsf{T}}F(\mathbf{x}_k)\| \le L_1\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|$

(b) $\|U_{k,2}U_{k,2}^{\mathsf{T}}F(\mathbf{x}_k)\| \le 2(L_2 + L_3)\|\mathbf{x}_k - \mathbf{x}^*\|^2$

(c) $\|U_{k,3}U_{k,3}^{\mathsf{T}}F(\mathbf{x}_k)\| \le (L_2 + L_3)\|\mathbf{x}_k - \bar{\mathbf{x}}_k\|^2$

Now, we have the local quadratic convergence theorem.

Theorem 4.7. *Under Assumptions 2.2, 2.3, 4.1, 4.2, and 4.3, the sequence* $\{\mathbf{x}_k\}$ *generated by DFLS algorithm converges to a point* $\mathbf{x}^* \in \mathbf{X}^*$ *quadratically.*

Proof. By Theorem 4.5, we know the sequence $\{\mathbf{x}_k\}$ converges to a point $\mathbf{x}^* \in \mathbf{X}^*$ superlinearly. Hence,

$$\lim_{k\to\infty} \|\mathbf{d}_k\| = 0.$$

This together with (4.19) implies that

$$\|\mathbf{d}_k\| < \Delta_k,$$

for all sufficiently large k. Hence, the trust region constraint in the DFLS algorithm is ultimately inactive. Therefore, from the SVD of $\nabla \mathbf{m}(\mathbf{x}_k)$, we have

$$\begin{aligned}\mathbf{d}_k &= -V_{k,1}(\Sigma_{k,1}^2 + \kappa_H^3\|\mathbf{m}(\mathbf{x}_k)\|)^{-1}\Sigma_{k,1}U_{k,1}^{\mathsf{T}}F(\mathbf{x}_k) \\ &\quad -V_{k,2}(\Sigma_{k,2}^2 + \kappa_H^3\|\mathbf{m}(\mathbf{x}_k)\|)^{-1}\Sigma_{k,2}U_{k,2}^{\mathsf{T}}F(\mathbf{x}_k) \\ &= -V_{k,1}(\Sigma_{k,1}^2 + \kappa_H^3\|F(\mathbf{x}_k)\|)^{-1}\Sigma_{k,1}U_{k,1}^{\mathsf{T}}F(\mathbf{x}_k) \\ &\quad -V_{k,2}(\Sigma_{k,2}^2 + \kappa_H^3\|F(\mathbf{x}_k)\|)^{-1}\Sigma_{k,2}U_{k,2}^{\mathsf{T}}F(\mathbf{x}_k),\end{aligned}$$

for sufficiently large k. Then, the result that the sequence $\{\mathbf{x}_k\}$ converges to $\mathbf{x}^*$ quadratically follows directly from the above equality, Lemma 4.6 and the same approach used for Theorem 2.2 in [12]. □

5. Numerical Experiments.

This section gives numerical comparisons of the performance of DFLS relative to the performance of the following codes:

- LMDIF [18]: A version, developed in 1980 by Garbow, Hillstrom and Moré of the Levenberg-Marquardt algorithm that uses finite (forward) differences for approximating the gradients of the sum of the squares objective function.[4]
- NEWUOA [21]: Software for unconstrained optimization without derivatives developed by Powell in 2004.

[4] As we already mentioned, the Levenberg-Marquardt algorithm is essentially a modified Gauss-Newton algorithm where the modification can be thought of as a regularisation of $J^{\top}J$ with a parameter scaled identity matrix. Although the motivation is rather different, if one associates the parameter with a trust region radius it is essentially a trust-region Gauss-Newton method.

LMDIF and NEWUOA belong to two fundamentally different classes of algorithms that do not use explicit derivatives. One class of methods use finite difference to approximate the derivative whereas the other uses polynomial interpolations to build approximation models of the problem. By paying very careful attention to both the maintenance of "good" geometry of the interpolating sets and the updating of the approximation models, the latter methods are numerically very stable and they are not particularly sensitive to moderate levels of noise. By contrast, finite difference methods are typically extremely sensitive to noise since the underlying algorithms often rely on reasonably accurate derivative information.

Here, we also want to point out that NEWUOA is derivative free software that was initially developed for general unconstrained optimization. An extension to accommodate simple bounds has been development by Powell ([20]) and the software, BOBYQA has recently been made available to anyone who asks for it directly from Professor Powell. Consequently, although NEWUOA is applicable to minimizing least-squares objectives, it does not make use of the problem structure. The purpose of nevertheless using it as a comparison code is to indicate that, perhaps not surprisingly, the performance of this type of algorithm in the case of least-squares can be greatly improved by taking advantage of the problem's structure. On the other hand LMDIF is a base code that serves to indicate the general superiority of the approach of [21], but our purpose here is to demonstrate the advantages of exploiting the specific structure of the least-squares problem.

Our implementation of the DFLS algorithm is based on the same framework as that provided by NEWUOA. This is partly because the details in NEWUOA are carefully and intelligently chosen and partly because it makes the comparison between it and DFLS more meaningful. In our runs of DFLS, we chose $N_P = 2n + 1$ sampling points. We applied the same polynomial interpolation techniques as those implemented in NEWUOA, on each of the nonlinear functions f_i, $i = 1, \ldots, m$. The model m_i interpolates f_i, for all $i = 1, \ldots, m$, at the $2n + 1$ sampling points. Hence, the degree of m_i, $i = 1, \ldots, m$, should be at least 2. The remaining freedom in defining m_i is taken up by minimizing the Frobenius norm of the change to the Hessian of m_i as in [22]. Since the sampling points for all the functions f_i, $i = 1, \ldots, m$, are the same, the computational complexity at each iteration is comparable to that of NEWUOA, except that DFLS requires more memory. In fact, DFLS needs this to store the individual models $m_i(\mathbf{x})$, for $i = 1, \ldots, m$. This leads to $(m - 1)(1 + n + n(n + 1)/2)$, which is $O(mn^2)$ more storage requirements compared with NEWUOA. Essentially, this is the cost of exploiting the structure. In addition, $O(mn)$ auxiliary memory may be needed for the implementation. However, discounting the occasional origin shifts, the amount of work per iteration is of the order of $(N_P + n)^2 \approx (3n)^2$ (see [22] for details). In NEWUOA, Powell developed a very elegant scheme for selecting the sampling points and for updating the models accurately and efficiently. Numerical rounding error and stability issues, whose control are crucial for the success of any software, are addressed in [18, 22] and the references therein and the reader is encouraged to obtain more details there.

All three codes were written in Fortran and compiled with F77 on an IBM ThinkPad T40 laptop computer. In LMDIF, it is admissible for the variables to be scaled internally. All the other parameters are set to the default values. In NEWUOA, we set the number of interpolating points to the default number recommended by the code, namely $2n + 1$. In both NEWUOA and DFLS, the initial trust region radius is set to 1.0. Both algorithms will stop if the trust region radius is less than 10^{-8}.

In derivative-free optimization, users are often interested in both low-accuracy and high-accuracy solutions. Hence, these three codes were tested in the following two subsections for different purposes. In the first subsection we assume there are no constraints on the computational budget and essentially want to compare the long-term behavior of different solvers. In the second subsection, we are more interested in the short-term behavior of those solvers under a given computational budget. A limited budget is often the bottleneck in practice, especially when function evaluations are expensive.

5.1. Long-term Behavior Comparisons. In this subsection, we essentially assume there is no limit on the computational budget and allow the solvers to explore the test problems as much as they reasonably can. Consequently, in all three codes, we set the maximum number of function evaluations to be 10^5, which is rather large for derivative-free optimization. For this purpose, the solvers were tested using 4 different sets of test problems. All the results are listed in Tables 5.1 to 5.4. In all the tables, "n" means the number of variables and "m" means the number of nonlinear functions in (1.1). "NF" stands for the number of function evaluations performed by each code. We put "$*$" in the table, if the code converges to a different local optimum from the global optimum.

Our first set of test problems consist of 25 zero residual problems. The first 15 problems are the problems with equality constraints in the Hock and Schittkowski [16] test problem library. They are reformulated as least-square problems by taking the objective function minus its optimum value as the first nonlinear function and taking all the equality constraints as the other nonlinear functions in (1.1). The remaining problems were collected from a variety of places during the development of the DFLS codes. The stopping condition for all the codes was

$$\Phi(\mathbf{x}_k) \leq \max\{10^{-12}, 10^{-20}\Phi(\mathbf{x}_0)\}, \tag{5.1}$$

where $\mathbf{x}_0$ is the initial point, which is rather stringent. The numerical results are listed in Table 5.1. All the codes stop either because the stopping condition (5.1) was satisfied or the internal conditions in the code determined that no further improvement could be made, perhaps due to the numerical rounding errors, or the number of function evaluations reached its maximum allowable value 10^5. If the optimal stopping condition was not met, we list in the column F-Error the best cost function value finally returned by the code, along with the number of function evaluations. From Table 5.1, we can see for the zero residual case DFLS is very stable and normally uses between 1/5 to 1/10 of the number of function evaluations of NEWUOA. Generally the LMDIF code does not perform well, although in a few cases it performs as well as DFLS. In many cases LMDIF could not reach the stopping condition (5.1) Based on these small number of numerical experiments, we believe the approach taken by DFLS is both promising and robust as concerns its long-term behavior for zero residual least-square problems. Maybe a more challenging set of test problems would have been more unfavourable to DFLS. However, if one considers the underlying issues, we think this is unlikely.

Our second test set consists of 22 nonzero residual problems. The "HS" problems in the second test set are formulated as least-square problems by simply taking the objective function as the first nonlinear function and taking the equality constraints for the remaining nonlinear functions in (1.1). In the case that the objective optimum value for the original formulation is zero, the first nonlinear function is taken as the objective function plus 10.0 to ensure that the least-squares residual is nonzero.

The "TRIGSSQS" problems are extracted from [21] and the remaining problems are intrinsically nonzero residue problems. Again we ran all the codes until either the internal conditions in the code determined that no further improvement could be made, or the number of function evaluations reached its maximum allowable value of 10^5. Then, for each code we calculate

$$\text{F-ReErr} = \frac{\Phi_{end} - \Phi^*}{\Phi^*},$$

where Φ_{end} is the final cost function value returned by the particular code and Φ^* is the lowest one among all the final (non-zero) cost function values. We list F-ReErr and the number of function evaluations whenever F-ReErr $\geq 10^{-12}$. The numerical results are shown in Table 5.2. From Table 5.2, we can see for the nonzero residual case, DFLS still performs better than NEWUOA when comparing their long-term behaviors. The difference is not as large as that in the zero residual case, which is what one might expect since the structure to be exploited is less. On the other hand, with the nonzero residual results, as one could expect, we can see that LMDIF was unable to reach very high accuracy for most of the test problems.

In many real problems, the situation is exacerbated by the presence of noise. For this reason, we wanted to test how noise affects the long-term performance of each code. We first tested a perturbed version of the zero residual problem set by adding some random noise to the nonlinear functions in (1.1). More specifically, we let

$$F(\mathbf{x}) = F_{true}(\mathbf{x}) + 10^{-2}E,$$

where $E \in \mathbb{R}^m$ is a random vector with a normal distribution in $[-0.5, 0.5]$. For each problem, we ran the code 10 times until either the internal conditions terminated the iterations, or the number of function evaluations reached its maximum allowable value, 10^5. The numerical results are shown in Table 5.3. In Table 5.3, "NF" means the average number of function evaluations during the 10 runs. Here, we let F-Error be the best true cost function value, i.e. the lowest cost function value without noise, returned by the code during the 10 runs. Since the original problems without noise are zero residue problems, we list F-Error in its column if F-Error ≤ 1.0; otherwise, we list "F" in its column to indicate that the implemented algorithm failed on this problem. We can readily see that DFLS normally gives the best cost function value. **Note that the error is comparable to the noise level since the functions, f_i are squared**. We remark that NEWUOA is also relatively insensitive to the noise in its long-term behavior. In fact, NEWUOA performs closer to DFLS in this case. However, as one could anticipate, because of the use of difference approximations for derivatives, LMDIF is much more sensitive to noise than either NEWUOA or DFLS.

Secondly, we tested the effect of noise for each method in the case of the nonzero residual problems. In this instance we let

$$F(\mathbf{x}) = F_{true}(\mathbf{x}) + 10^{-2}E \circ |F(\mathbf{x})|.$$

Here, $E \in \mathbb{R}^m$ is again a random vector with normal distribution in $[-0.5, 0.5]$ and $|F(\mathbf{x})|$ is used to denote the vector in $\mathbb{R}^m$ with its components the absolute value of the corresponding component of $F(\mathbf{x})$. Here "$\circ$" stands for the componentwise product. For each problem, we again ran each code 10 times until either the internal conditions terminated the algorithm iterations, or the number of function evaluations reached its maximum allowable value of 10^5. Then, for each method we calculated

$$\text{F-ReErr} = \frac{\Phi_{best} - \Phi^*}{\Phi^*},$$

Problem Name	n	m	LMDIF	NEWUOA	DFLS
			NF/F-Error	NF/F-Error	NF/F-Error
HS26	3	2	890/6.22e-9	3706	378
HS27	3	2	281/2.40e-11	405	338
HS28	3	2	107	247	48
HS39	4	3	81	585	53
HS40	4	4	41	344	30
HS42	4	3	139/4.86e-12	655	48
HS46	5	3	1762/3.75e-8	7060	1076
HS47	5	4	2186/4.40e-11	2151	263
HS49	5	3	10^5/4.18e-7	2546	2395
HS56	7	5	266/1.55e-9	1851	168
HS60	3	2	123/2.29e-10	415	65
HS77	5	3	638/8.35e-5	1439	145
HS78	5	4	79	727	48
HS81	5	4	232/6.06e-7	2262	107
HS111	10	4	4434/8.77e-6	10^5/2.32e-8	2193/7.85e-12
CONN	20	20	667	639	113
CHROSEN	20	38	169	635	96
CHROSEN	80	158	649	3094	346
TRIGSSQS	10	10	67	490	81
TRIGSSQS	10	20	45	236	49
TRIGSSQS	20	20	170	2267	145
BUCKLEY21	6	13	113	2076	171
HEART	6	6	*	10^5/1.21e-2	863
MORE25	20	22	400	10359	198
RORE20	30	31	295	10^5/1.23e-8	431/8.00e-12

Table 5.1

Numerical Comparisons (zero residuals)

where Φ_{best} is the best true (i.e. with the added noise) cost function value returned by the code during the 10 runs and Φ^* is the optimal cost function value of the problem without noise. The numerical results are shown in Table 5.4. Here, "NF" again means the average number of function evaluations during the 10 runs. We list F-ReErr, along with the number of function evaluations if F-ReErr $\leq$ 1.0; otherwise, we list "F" in its column to indicate that the code is not successful for this problem. From Table 5.4, we can see DFLS gives the best final solutions in almost all cases, or otherwise the difference is marginal. NEWUOA also gives reasonable solutions, although DFLS is generally superior. LMDIF in this case, as is to be expected, is even more sensitive to the noise in its long-term performance. In fact, for many problems, LMDIF could not provide a good approximate to the true solution.

5.2. Short-term Behavior Comparisons. In practice, when the function evaluations are more expensive, a low-accuracy solution is often all that can be obtained whilst maintaining the user's computational budget. In addition, in many of these situations, the underlining codes for providing function values are often based on numerical simulations or discretizations of some continuous model. So, the provided function information may only make sense for a low-accuracy solution. Hence, as is

Problem Name	n	m	LMDIF NF/F-ReErr	NEWUOA NF/F-ReErr	DFLS NF/F-ReErr
HS26	3	2	90/1.97e-7	189	155
HS27	3	2	208/1.06e-3	248	108
HS28	3	2	1950	107	143
HS39	4	3	104/5.30e-8	245	100
HS40	4	4	54/8.91e-7	125	76
HS42	4	3	165/3.91e-6	84	59
HS46	5	3	1788/1.54e-5	600/1.99e-12	586
HS47	5	4	*	92	92
HS49	5	3	10^5/6.86e-5	832/3.51e-12	516
HS56	7	5	169/3.21e-5	234	160
HS60	3	2	139/3.93e-5	217	81
HS77	5	3	959/2.44e-3	655	162
HS78	5	4	226/7.76e-7	158	105
HS81	5	4	194/1.69e-2	949	214
HS111	10	4	90451/1.16e-6	10562	673
DENNIS	4	20	2034/2.38e-11	254	130
SPHRPTS	20	45	*	2263	1406
TRIGSSQS	10	10	230/7.10e-7	241	227
TRIGSSQS	10	20	215/4.70e-6	217	241
TRIGSSQS	20	20	378/6.33e-5	580	576
PENALTY1	20	21	219/5.87e-5	9855	433
PENALTY2	20	40	442/1.89e-10	1982	336

TABLE 5.2
Numerical Comparisons (nonzero residuals)

pointed out in the article [19], under these situations users are interested to know, at least statistically, the ratio of the obtained function value reduction as a function of the number of function evaluations. So, in this subsection, we apply the same procedure as that used in [19] to evaluate the short-term behavior of the different solvers and we use the following convergence condition:

$$\Phi(\mathbf{x}_0) - \Phi(\mathbf{x}_k) \geq (1-\tau)(\Phi(\mathbf{x}_0) - \Phi_L), \tag{5.2}$$

where $0 \leq \tau < 1$ is a tolerance and Φ_L is the smallest function value obtained by any solver within the same computational budget. Since our computational results in this section focus on demonstrating the short-term behavior of different solvers, we would like to investigate the behavior of the solvers within a limit of 50 (simplex) gradients. That is for each problem with dimension n_p , the maximum number of function evaluations allowed for each solver is $50(n_p+1)$. Then, given a test problem set, $\mathcal{P}$, the performance of the different solvers can be measured by the percentage of problems that can be solved (for a given a tolerance τ) within a certain number of (simplex) gradient evaluations. Now, by defining $t_{p,s}$ to be the number of function evaluations needed for solver s to satisfy (5.2) for a problem $p \in \mathcal{P}$ with dimension n_p, the percentage of problems solved as a function of a computational budget of

Problem Name	n	m	LMDIF	NEWUOA	DFLS
			NF/F-Error	NF/F-Error	NF/F-Error
HS26	3	2	52.2/4.22e-2	91.6/4.09e-3	87.4/1.80e-5
HS27	3	2	F	106.1/1.85e-2	94.3/2.48e-5
HS28	3	2	63.3/1.22e-1	93.9/9.32e-5	85.1/1.29e-5
HS39	4	3	83.6/2.29e-1	122.6/0.22	99.4/1.28e-6
HS40	4	4	43.1/3.11e-2	102.9/9.88e-5	95.4/2.32e-5
HS42	4	3	45.2/2.78e-1	115.4/3.65e-3	93.8/3.55e-6
HS46	5	3	80.8/1.77e-1	161.9/1.36e-2	133.8/5.65e-5
HS47	5	4	89.8/7.77e-1	171.9/6.15e-3	136.3/6.90e-5
HS49	5	3	F	F	192/6.05e-4
HS56	7	5	F	174.6/6.41e-4	168/2.64e-4
HS60	3	2	63.34.12e-3	81.1/1.04e-2	85.7/1.91e-5
HS77	5	3	95.1/5.53e-2	158/7.95e-2	149.4/1.95e-3
HS78	5	4	82.1/3.83e-3	117.4/1.46e-2	116.7/2.14e-5
HS81	5	4	83.4/3.61e-5	148.9/7.25e-2	115.4/1.34e-4
HS111	10	4	136.6/1.92e-2	248.7/8.01e-2	235.3/1.32e-4
CONN	20	20	F	625.9/2.95e-3	415.5/7.96e-5
CHROSEN	20	38	F	479.8/4.36e-3	385.4/1.14e-4
CHROSEN	80	158	F	2292.7/4.64e-1	1866.4/4.03e-4
TRIGSSQS	10	10	167.5/4.06e-4	347.3/3.22e-3	205/1.02e-4
TRIGSSQS	10	20	155.5/1.20e-4	254.1/3.14e-4	158/3.65e-5
TRIGSSQS	20	20	449.1/5.00e-3	816.5/2.1e-1	375.7/1.61e-4
BUCKLEY21	6	13	60.1/7.9e-1	159.8/1.5e-1	168.7/3.94e-5
HEART	6	6	F	F	196.6/4.2e-1
MORE25	20	22	F	F	450.2/2.28e-4
RORE20	30	31	F	961/2.3e-1	588.1/1.18e-4

Table 5.3

Numerical Comparisons (zero residuals, noise level 1.e-2)

(simplex) gradients can be expressed as a function

$$d_s(\alpha) = \frac{|p \in \mathcal{P} : t_{p,s} \leq \alpha(n_p + 1)|}{|\mathcal{P}|},$$

where α is a positive integer and $|\cdot|$ means the cardinality of the set (see [19]). By convention, the set $t_{p,s} = \infty$, if the convergence test (5.2) can not be satisfied after the pre-specified computational budget.

The test problem set $\mathcal{P}$ used in this section is again proposed in the paper [19] and the source code for these problems can be downloaded from Jorge Moré's Web page (http:// www.mcs.anl.gov/∼more/dfo). This benchmark test problem set comprises of 22 types of nonlinear least-square problems from the CUTEr [13] problem library. Because of the different combinations of starting points and number of variables for each problem, the benchmark set $\mathcal{P}$ has a total of 53 problems and each of the 22 types of nonlinear least-square problems is fairly represented. For more details on how this problem set was formulated, readers can refer to [19].

The data profiles in Figure 5.1 shows the performances of the different solvers for the original 53 problems in the test set, $\mathcal{P}$, for various sizes of the computational budget up to 50 (simplex) gradients and different levels of accuracy, $\tau = 10^{-1}, 10^{-3}, 10^{-5}$

Problem Name	n	m	LMDIF NF/F-Error	NEWUOA NF/F-Error	DFLS NF/F-Error
HS26	3	2	40.9/6.63e-1	84/1.81e-2	94.3/1.79e-3
HS27	3	2	F	98.2/6.29e-1	113.3/4.14e-4
HS28	3	2	38.8/5.21e-1	83/1.41e-2	92.1/2.20e-2
HS39	4	3	F	124.5/2.09e-2	116.7/3.22e-3
HS40	4	4	59.3/9.38e-3	107.5/1.11e-2	105.4/8.60e-4
HS42	4	3	F	94.4/1.48e-3	105.1/2.31e-3
HS46	5	3	49.2/1.92e-1	131.4/1.13e-2	132.5/3.30e-3
HS47	5	4	*	92/1.33e-1	92/5.20e-2
HS49	5	3	F	F	156.3/5.25e-1
HS56	7	5	F	185.2/1.20e-2	174.2/8.71e-3
HS60	3	2	71.9/3.74e-3	124.2/6.02e-2	102/1.57e-4
HS77	5	3	84.2/6.47e-1	F	157.9/4.81e-2
HS78	5	4	63.9/2.26e-2	141.6/1.51e-3	121.5/1.69e-3
HS81	5	4	84.9/1.05e-1	F	154/2.94e-3
HS111	10	4	F	238.6/2.76e-1	248.3/2.09e-2
DENNIS	4	20	F	164.8/8.59e-3	133.9/3.61e-4
SPHRPTS	20	45	F	562.5/8.18e-3	541.4/6.59e-3
TRIGSSQS	10	10	74.7/2.03e-1	241.1/5.38e-3	248.9/4.29e-3
TRIGSSQS	10	20	59.3/2.93e-1	249.4/3.05e-3	260.2/1.89e-3
TRIGSSQS	20	20	114.8/3.43e-1	468.5/7.28e-3	473.5/8.78e-3
PENALTY1	20	21	135.7/2.60e-1	590.7/1.26e-1	475.8/5.67e-3
PENALTY2	20	40	F	447.4/3.26e-3	417.3/3.50e-3

Table 5.4

Numerical Comparisons (nonzero residuals, relative noise 1.e-2)

and 10^{-7}. When the tolerance τ is relatively large, it seems the performances of these solvers are more alike. In this case, although there some differences when the computational budget is very small, say within the budget of 25 (simplex) gradients, all solvers seem to solve more than 90% problems within 50 (simplex) gradients (See Figures 5.1 (a) and (b)). However, when the tolerance becomes small, say $\tau = 10^{-5}$, it seems that DFLS and LMDIF outperform NEWUOA within the budget of 25 (simplex) gradients, but NEWUOA is able to catch up with LMDIF by 50 (simplex) gradients (See Figure 5.1 (c)). When the tolerance is even tighter, say $\tau = 10^{-7}$, it seems DFLS ranks first, LMDIF ranks second and NEWUOA ranks third (See Figure 5.1 (d)). This result is perhaps not surprising. Because NEWUOA assumes no knowledge of the problem structure it probably is unable to build a sufficiently accurate model within a very small computational budget, while both DFLS and LMDIF take advantages of the structure of the least-square problem and are thereby capable of constructing a more adequate model. In all the cases, DFLS seems to perform the best of these three solvers. Additional, perhaps very useful, information is provided by Figure 5.1 on how to allocate a computational budget. For example, DFLS seems be able to obtain the function value to the level $(1 - 10^{-7})\Phi_L$ for about 88% of the smooth problems in $\mathcal{P}$ within only about 22 (simplex) gradients, where ϕ_L is the smallest function value obtained by any solver within 50 (simplex) gradients.

As suggested in the paper [19], we also want to see the short-term behavior of different solvers on the second class of problems proposed in [19], which mimic sim-

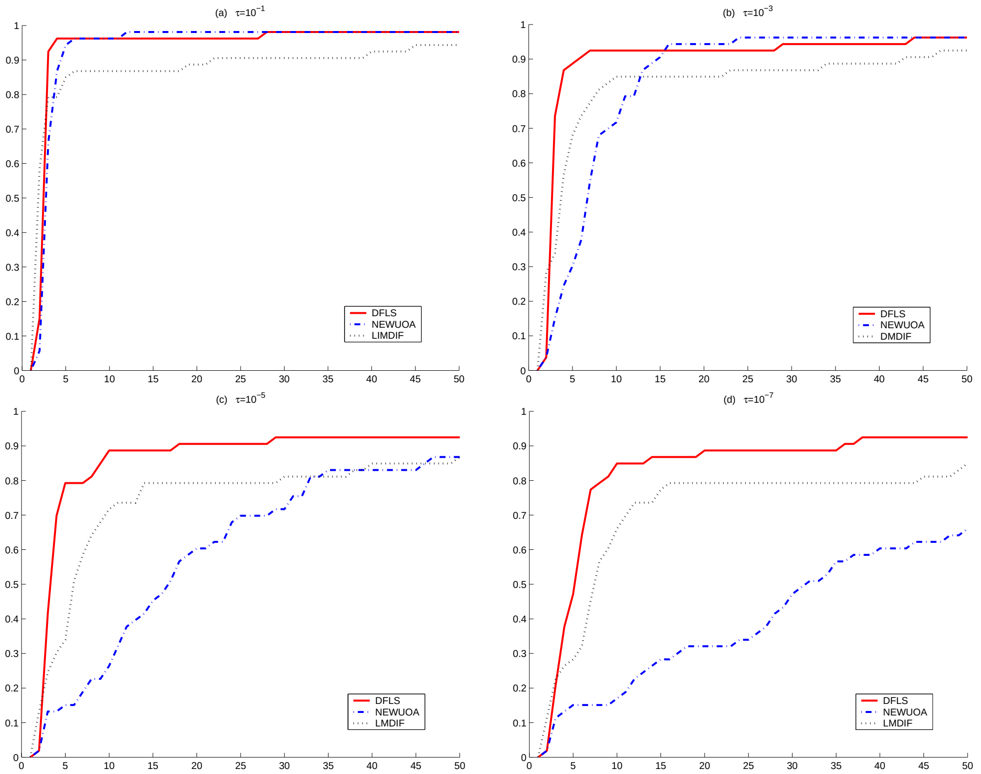

FIG. 5.1. *Data profiles of function $d_s(\alpha)$ for smooth problems: (a) $\tau = 10^{-1}$, (b) $\tau = 10^{-3}$, (c) $\tau = 10^{-5}$, (d) $\tau = 10^{-7}$*

ulations defined by some iterative process, for example, solving differential equations with parameters. For convenience, we repeat the following definition of this class of problems, denoted by $\mathcal{P}_\mathcal{N}$, in terms of our notation: Let

$$\begin{aligned}\Phi_\mathcal{N}(\mathbf{x}) &= (1 + \epsilon_\mathcal{N}\pi(\mathbf{x}))\Phi(\mathbf{x}) \\ &= \frac{1}{2}(1 + \epsilon_\mathcal{N}\pi(\mathbf{x}))\sum_{i=1}^{m} f_i^2(\mathbf{x}) \end{aligned} \tag{5.3}$$

where $0 < \epsilon_\mathcal{N} < 1$ is the relative noise level, $\Phi(\mathbf{x})$ is a problem in $\mathcal{P}$ and the non-stochastic noise function $\pi : \mathbb{R}^n \to [-1, 1]$ is defined in terms of the cubic Chebyshev polynomial T_3 by

$$\pi(\mathbf{x}) = T_3(\pi_0(\mathbf{x})), \qquad T_3(\alpha) = \alpha(4\alpha^2 - 3),$$

and

$$\pi_0(\mathbf{x}) = 0.9\sin(100\|\mathbf{x}\|_1)\cos(100\|x\|_\infty) + 0.1\cos(\|\mathbf{x}\|_2).$$

One advantage of using a non-stochastic noise function is that the computational results are reproducible. In total there are 53 problems in $\mathcal{P}_\mathcal{N}$. In addition, from the definition (5.3), the problem $\Phi_\mathcal{N}(\mathbf{x}) \in \mathcal{P}_\mathcal{N}$ can also be easily written in our format as

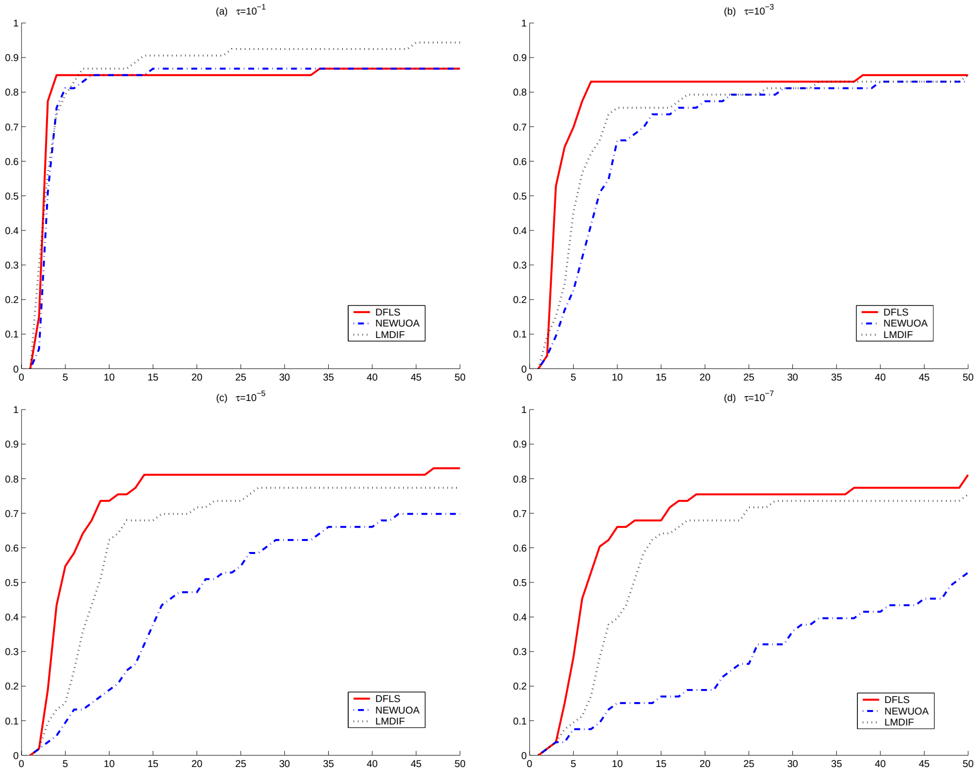

FIG. 5.2. *Data profiles of function $d_s(\alpha)$ for non-stochastic noisy problems: (a) $\tau = 10^{-1}$, (b) $\tau = 10^{-3}$, (c) $\tau = 10^{-5}$, (d) $\tau = 10^{-7}$*

the following least-square problem:

$$\Phi_{\mathcal{N}}(\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{m} \hat{f}_i^2(\mathbf{x}),$$

where $\hat{f}_i(\mathbf{x}) = \sqrt{1 + \epsilon_{\mathcal{N}}\pi(\mathbf{x})}f_i(\mathbf{x})$ for $i = 1, \ldots, m$.

As in [19], we set the noise level, $\epsilon_{\mathcal{N}} = 10^{-3}$, in our numerical experiments. The performances of different solvers for this second problem set, $\mathcal{P}_{\mathcal{N}}$, for all sizes of the computational budget up to 50 (simplex) gradients and different levels of accuracy are shown in Figure 5.2. Unsurprisingly, for each level of accuracy, the performances of all of the solvers declines for this non-stochastic, noisy problem set. The polynomial interpolation based methods DFLS and NEWUOA are more affected than the finite difference based method LMDIF compared with the their performances for the problem set $\mathcal{P}$ without noise, but mostly in the case of DFLS in the sense that its performance is closer to that of LMDIF. But a little surprisingly, for very low accuracy solutions, LMDIF now performs slightly better than DFLS and NEWUOA (See Fig 5.2 (a)). This is perhaps because for the short term behavior and very low accuracy, using finite difference to approximate the derivative is not that critical for early iterations. But as we require more and more accuracy, DFLS outperforms LMDIF if we have a small computational budget, say within the budget of 25 (simplex) gradients, but the gap between DFLS and LMDIF reduces by the time we reach 50 (simplex) gradients (See Fig 5.2 (b), (c) and (d)). However, the short-term behavior of NEWUOA remains behind LMDIF and DFLS, and the gap becomes larger as the tolerance becomes smaller. This phenomena is different from what we have observed in the stochastic noise case, in the previous subsection, when comparing the long-term behavior. In that case, we have essentially an unlimited budget and we are trying to obtain a solution to a very high accuracy and the stochastic noises finally affects very significantly LMDIF. On the other hand, DFLS and NEWUOA gradually build quadratic models based on the accumulated information from previous iterations. So, the noise is ultimately, to some extent at least, filtered or canceled. This may explain why DFLS and NEWUOA performs better than LMDIF when comparing the long-term behavior. Again, since NEWUOA does not use the structure of the least-square problem, its performance is worst in this case when comparing short-term behavior. DFLS consistently performs very well.

6. Conclusion. In this paper, we present a framework for a class of derivative-free algorithms, called DFLS, for minimizing least-square problems. Several global and local convergence issues have been addressed in the paper. It is worth noting that numerical results can vary considerably depending upon the presence or absence of noise and if one is content with rather approximate solutions or one requires more accurate ones. This is the case even to the extent that the algorithm of choice might change. For example in the present paper there are circumstances where LMDIF is preferable to NEWUOA, although overall it is clear that DFLS is preferable to both. More specifically, we have established global convergence and local quadratic convergence results of these algorithms under certain reasonable conditions. Preliminary numerical experiments have been carried out to compare both the short-term and the long-term behavior of DFLS with the other two benchmark algorithms, NEWUOA and LMDIF. For the long-term behavior, we have observed that the polynomial interpolation based methods, NEWUOA and DFLS, are more efficient and reliable when accurate solutions (for both problems with and without noise) are required. Consequently, for long-term, robust and accurate solutions, compared with finite difference based methods, the polynomial interpolation based methods are recommended. However, for the least-square minimization, without making use of the problem structure, the polynomial interpolation based method NEWUOA may not build accurate models given a small computing budget, and hence can not perform as well in the short-term compared with LMDIF, which does make full use of the problem structure. Therefore, especially for short-term efficiency, the methods should take advantages of the problem structure. On the other hand, DFLS are not only based on models built by polynomial interpolations, but also takes full advantages of the least-square problem structure. Hence, it is reasonable to believe DFLS will perform more efficiently and will be more robust in both the short-term and the long-term compared with methods that only implement one of the two features in the above discussion. Our numerical result support this conclusion.

REFERENCES

[1] C. Audet and J. E. Dennis Jr., *A pattern search flter methods for nonlinear programming without derivatives*, SIAM Journal on Optim., 14 (2004), pp. 980–1010.

[2] G. A. Gray and T. G. Kolda, Algorithm 856: APPSPACK 4.0: asynchronous parallel pattern search for derivative-free optimization, ACM Tran. Math. Software, 32 (2006), pp. 485–507.

[3] A. R. Conn and N. I. M. Gould and Ph. L. Toint, *Trust-Region Methods*, MPS-SIAM Series on Optimization, SIAM, Philadelphia, 2000.

[4] A. R. Conn and K. Scheinberg and L. N. Vicente, *Global Convergence of General Derivative-Free Trust-Region Algorithms to First and Second Order Critical Points* Math. Program., (2009) to appear.

[5] A. R. Conn and K. Scheinberg and L. N. Vicente, *Geometry of interpolation sets in derivative free optimization*, Math. Program., 111 (2008), pp. 141–172

[6] A. R. Conn and K. Scheinberg and L. N. Vicente, *Introduction to Derivative-Free Optimization*, MPS-SIAM Series on Optimization, SIAM, Philadelphia, 2009.

[7] A. R. Conn and K. Scheinberg and L. N. Vicente, *Geometry of sample sets in derivative free optimization: Polynomial regression and underdetermined interpolation*, IMA J. Numer. Anal., 28 (2008), pp. 721–748

[8] H. Dan and N. Yamashita and M. Fukushima, *Convergence properties of the inexact Levenberg-Marquardt method under local error bound*, Optimization Methods and Software, 17 (2002), pp. 605-626

[9] J. E. Dennis and J. J. Moré, *A characterization of superlinear convergence and its application to quasi-Newton Methods*, Math. Comp., 28 (1974), pp. 549–560

[10] J. E. Dennis and J. J. Moré, *Quasi-Newton methods, motivation and theory*, Siam Rev., 19 (1977), pp. 46–89

[11] J. Fan, *Convergence properties of a self-adaptive Levenberg-Marquardt algorithm under local error bound condition*, Comput. Optim. Appl., 34, (2006), pp. 47–62

[12] J. Fan and Y. Yuan, *On the quadratic convergence of the Levenberg-Marquardt method without nonsingularity assumption*, Computing, 74 (2005), pp. 23–39

[13] N. I. M. Gould, D. Orban and Ph. L. Toint, *CUTEr and SifDec: A constrained and unconstrained testing environment, revisited*, ACM Trans. Math. Software, 29, (2003), pp. 373–394.

[14] W. W. Hager and H. Zhang, *Self-adaptive inexact proximal point methods*, Comput. Optim. Appl., 39, (2008), pp. 161–181

[15] W. W. Hager and H. Zhang, *Asymptotic convergence analysis of a new class of proximal point methods*, SIAM J. Control and Optim., 46 (2007), pp. 1683–1704

[16] W. Hock and K. Schittkowski, *Test examples for nonlinear programming codes*, Lecture Notes in Eco. and Math. Systems 187, 1981

[17] M. Marazzi and J. Norcedal, *Wedge trust region methods for Derivative-free Optimization*, Math. Program., 91 (2002), pp. 289–305.

[18] J. J. Moré, *The Levenberg-Marquardt Algorithm, Implementation and Theory*, Numerical Analysis, G. A. Watson, eds., Lecture Notes in Mathematics 630, Springer-Verlag, 1977

[19] J. J. Moré and S. M. Wild, *Benchmarking Derivative-Free Optimization Algorithms*, Mathematics and Computer Science Division, Argonne National Laboratory, Preprint ANL/MCS-P1471-1207, April 2008.

[20] M. J. D. Powell , *Developments of NEWUOA for unconstrained minimization without derivatives*, IMA Journal of Numerical Analysis, 28 (2008), pp. 649–664,

[21] M. J. D. Powell , *The NEWUOA software for unconstrained optimization without derivatives*, DAMTP 2004

[22] M. J. D. Powell , *Least Frobenius norm updating of quadratic models that satisfy iterpolation conditions*, Math. Program., Ser. B, 100 (2004), pp. 183–215

[23] M. J. D. Powell , *On trust region methods for unconstrained minimization without derivatives*, Math. Program., 97 (2003), pp. 605–623

[24] M. J. D. Powell , *A new algorithm for unconstrained optimization*, In J. B. Rosen, O. L. Mangasarian and K. Ritter eds., Nonlinear Programming, (Academic Press, New York, 1970), PP. 31–36

[25] P. Tseng, *Error bounds and superlinear convergence analysis of some Newton-type methods in optimization*, In Nonlinear Optimization and Related Topics, G. D. Pillo and F. Giannessi, eds., Kluwer, 2000, pp. 445–462

[26] N. Yamashita and M. Fukushima, *The proximal point algorithm with genuine superlinear convergence for the monotone complementarity problem*, SIAM J. Optim., 11 (2000), pp. 364–379

[27] N. Yamashita and M. Fukushima, *On the rate of convergence of the Levenberg-Marquardt method*, Computing (Suppl.15), Springer, 2001, pp. 237–249