Unconstrained optimization of real functions in complex variables

Laurent Sorber
Marc Van Barel
Lieven De Lathauwer

Report TW 592, April 2011



Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

Unconstrained optimization of real functions in complex variables

Laurent Sorber
Marc Van Barel
Lieven De Lathauwer

Report TW 592, April 2011

Department of Computer Science, K.U.Leuven

Abstract

Nonlinear optimization problems in complex variables are frequently encountered in applied mathematics and engineering applications such as control theory, signal processing and electrical engineering. Optimization of these problems often requires a first- or second-order approximation of the objective function to generate a new step or descent direction. However, such methods cannot be applied to real functions in complex variables because they are necessarily nonanalytic in their argument, i.e., the Taylor series expansion in their argument alone does not exist. To overcome this problem, the objective function is often redefined as a function of the real and imaginary parts of its complex argument so that standard optimization methods can be applied. We show that real functions in complex variables do have a Taylor series expansion in complex variables, which we then use to generalize existing optimization methods for both general nonlinear optimization problems and nonlinear least squares problems. We then apply these methods to a number of case studies which show that complex Taylor expansions can lead to greater insight in the structure of the problem and that this structure can often be exploited to improve computational complexity and storage cost.

Keywords : unconstrained optimization, functions of complex variables, quasi-Newton, BFGS, L-BFGS, nonlinear least squares, Gauss–Newton, Levenberg–Marquardt, Wirtinger calculus.
MSC : Primary : 90-08, 90C06, 90C53, 65K05, Secondary : 90C90.

UNCONSTRAINED OPTIMIZATION OF REAL FUNCTIONS IN COMPLEX VARIABLES

LAURENT SORBER[†], MARC VAN BAREL[†], AND LIEVEN DE LATHAUWER[‡§]

Abstract. Nonlinear optimization problems in complex variables are frequently encountered in applied mathematics and engineering applications such as control theory, signal processing and electrical engineering. Optimization of these problems often requires a first- or second-order approximation of the objective function to generate a new step or descent direction. However, such methods cannot be applied to real functions in complex variables because they are necessarily nonanalytic in their argument, i.e., the Taylor series expansion in their argument alone does not exist. To overcome this problem, the objective function is often redefined as a function of the real and imaginary parts of its complex argument so that standard optimization methods can be applied. We show that real functions in complex variables do have a Taylor series expansion in complex variables, which we then use to generalize existing optimization methods for both general nonlinear optimization problems and nonlinear least squares problems. We then apply these methods to a number of case studies which show that complex Taylor expansions can lead to greater insight in the structure of the problem and that this structure can often be exploited to improve computational complexity and storage cost.

Key words. unconstrained optimization, functions of complex variables, quasi-Newton, BFGS, L-BFGS, nonlinear least squares, Gauss–Newton, Levenberg–Marquardt, Wirtinger calculus

AMS subject classifications. 90-08, 90C06, 90C53, 90C90, 65K05

1. Introduction.

In this article we focus on methods to solve unconstrained nonlinear optimization problems of the form

$$\min_{\boldsymbol{z}} f(\boldsymbol{z}, \overline{\boldsymbol{z}}) \tag{1.1}$$

where f is a real, smooth function in n complex variables $\boldsymbol{z}$ and their complex conjugates $\overline{\boldsymbol{z}}$. We will also consider unconstrained nonlinear least squares problems of the form

$$\min_{\boldsymbol{z}} \frac{1}{2}\|F(\boldsymbol{z}, \overline{\boldsymbol{z}})\|^2 \tag{1.2}$$

where F maps n complex variables $\boldsymbol{z}$ and their complex conjugates $\overline{\boldsymbol{z}}$ to m complex residuals $F(\boldsymbol{z}, \overline{\boldsymbol{z}})$. Many nonlinear optimization methods use a first- or second-order approximation of the objective function to generate a new step or a descent direction, where the approximation is either updated or recomputed every iteration. A problem that arises with real-valued functions f in complex variables $\boldsymbol{z}$ is that they are necessarily nonanalytic in $\boldsymbol{z}$. In other words, there exists no Taylor series in $\boldsymbol{z}$ of f at $\boldsymbol{z}_0$ so that the series converges to $f(\boldsymbol{z})$ in a neighbourhood of $\boldsymbol{z}_0$. A common workaround is to convert the optimization problem to the real domain by viewing f as a function of the real and imaginary parts of $\boldsymbol{z}$. However, by reformulating an optimization problem that is inherently complex to the real domain, it becomes easy to miss important insights about the structure of the problem which might otherwise be exploited. By making the dependence of the objective function on both $\boldsymbol{z}$ and $\overline{\boldsymbol{z}}$

[†]Department of Computer Science, Katholieke Universiteit Leuven, Celestijnenlaan 200A, B-3001 Leuven, Belgium ({Laurent.Sorber,Marc.VanBarel}@cs.kuleuven.be).

[‡]Group Science, Engineering and Technology, Katholieke Universiteit Leuven Campus Kortrijk, E. Sabbelaan 53, B-8500 Kortrijk, Belgium (Lieven.DeLathauwer@kuleuven-kortrijk.be).

[§]Department of Electrical Engineering (ESAT), Katholieke Universiteit Leuven, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium (Lieven.DeLathauwer@esat.kuleuven.be).

explicit as we have in (1.1) and (1.2), we will see that there are several ways to expand these functions into a complex Taylor series. The key to this observation lies in the fact that if a function is analytic in the space spanned by $\mathrm{Re}\{\boldsymbol{z}\}$ and $\mathrm{Im}\{\boldsymbol{z}\}$, it is also analytic in the space spanned by $\boldsymbol{z}$ and $\overline{\boldsymbol{z}}$. These expansions allow us to generalize existing real optimization methods to the complex domain, and importantly, depend on complex derivatives that are often described by more elegant expressions than their real counterparts.

The paper is organized as follows. In Section §2 we first review the notation, and give a short overview of Wirtinger calculus, which is the underlying framework for the complex derivatives used in this article. We then apply this framework to derive Taylor series expansions of functions in complex variables. In Section §3 we use these expansions to generalize nonlinear optimization methods such as L-BFGS and nonlinear conjugate gradient to functions in complex variables. In Section §4 we do the same for nonlinear least squares methods such as Gauss–Newton and Levenberg–Marquardt. In Section §5 we show the potential of these generalized optimization methods with two case studies. The first is the canonical polyadic decomposition, which is a tensor decomposition in rank-one terms. The second is the simulation of nonlinear circuits in the frequency domain. We conclude the paper in Section §6.

2. Wirtinger calculus.

2.1. Notation and preliminaries. Vectors are denoted by boldface letters and are lower case, e.g., $\boldsymbol{a}$. Matrices are denoted by capital letters, e.g., A. Higher-order tensors are denoted by Euler script letters, e.g., $\mathcal{A}$. The ith entry of a vector $\boldsymbol{a}$ is denoted by a_i, element (i,j) of a matrix A by a_{ij} and element (i,j,k) of a third-order tensor $\mathcal{A}$ by a_{ijk}. Indices typically range from one to their capital version, e.g., $i = 1,\ldots,I$. A colon is used to indicate all elements of a mode. Thus $\boldsymbol{a}_{:j}$ corresponds to the jth column of a matrix A, which we write compactly as $\boldsymbol{a}_j$. The nth element in a sequence is denoted by a superscript in parentheses, e.g., $A^{(n)}$ denotes the nth matrix in a sequence. The superscripts $\cdot^{\mathrm{T}}$, $\cdot^{\mathrm{H}}$, $\cdot^{-1}$ and $\cdot^{\dagger}$ are used for the transpose, Hermitian conjugate, matrix inversion and Moore–Penrose pseudo-inverse, respectively. The identity matrix of order n is denoted by $\mathbb{I}_{n\times n}$. The complex conjugate is denoted by an overline, e.g., $\overline{\boldsymbol{a}}^{\mathrm{T}}$ is equivalent to $\boldsymbol{a}^{\mathrm{H}}$. The two-norm and Frobenius norm are denoted by $\|\cdot\|$ and $\|\cdot\|_F$, respectively. We use parentheses to denote the concatenation of two or more vectors, e.g., $(\boldsymbol{a},\boldsymbol{b})$ is equivalent to $\begin{bmatrix}\boldsymbol{a}^{\mathrm{T}} & \boldsymbol{b}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$.

The calculus underlying the complex derivatives in this article goes back to H. Poincaré and was developed principally by W. Wirtinger. It is often called, especially in the German literature, the Wirtinger calculus [36]. In order to facilitate the transition from and to the real and complex numbers, we first define the vector spaces $\mathcal{R}$, $\mathcal{C}$ and $\mathcal{C}^*$ as $\{(\boldsymbol{x},\boldsymbol{y})\,|\boldsymbol{x}\in\mathbb{R}^n,\boldsymbol{y}\in\mathbb{R}^n\} = \mathbb{R}^{2n}$, $\{(\boldsymbol{z},\overline{\boldsymbol{z}})\,|\boldsymbol{z}\in\mathbb{C}^n\}\subset\mathbb{C}^{2n}$ and $\{(\overline{\boldsymbol{z}},\boldsymbol{z})\,|\boldsymbol{z}\in\mathbb{C}^n\}\subset\mathbb{C}^{2n}$, respectively. Elements of $\mathbb{C}^n$ have an equivalent representation in any of these spaces. Let $\boldsymbol{z}\in\mathbb{C}^n$, then we define $\overset{\mathcal{R}}{\boldsymbol{z}} \triangleq (\mathrm{Re}\{\boldsymbol{z}\},\mathrm{Im}\{\boldsymbol{z}\})\in\mathcal{R}$, $\overset{\mathcal{C}}{\boldsymbol{z}} \triangleq (\boldsymbol{z},\overline{\boldsymbol{z}})\in\mathcal{C}$ and $\overset{\mathcal{C}^*}{\boldsymbol{z}} \triangleq (\overline{\boldsymbol{z}},\boldsymbol{z}) = \overset{\mathcal{C}}{\boldsymbol{z}}\in\mathcal{C}$. Furthermore, the linear map

$$J \triangleq \begin{bmatrix}\mathbb{I}_{n\times n} & i\mathbb{I}_{n\times n}\\ \mathbb{I}_{n\times n} & -i\mathbb{I}_{n\times n}\end{bmatrix} \tag{2.1}$$

is an isomorphism from $\mathcal{R}$ to $\mathcal{C}$ and its inverse is given by $J^{-1} = \frac{1}{2}J^{\mathrm{H}}$. The *swap operator*

$$S \triangleq \begin{bmatrix}0 & \mathbb{I}_{n\times n}\\ \mathbb{I}_{n\times n} & 0\end{bmatrix} \tag{2.2}$$

is an isomorphism from $\mathcal{C}$ to the *dual space* $\mathcal{C}^*$. Its inverse is given by $S^{-1} = S^{\mathrm{T}} = S$.

DEFINITION 2.1. *Let* $\boldsymbol{z} \in \mathbb{C}^n$ *and let* $\boldsymbol{x} = \mathrm{Re}\{\boldsymbol{z}\}$ *and* $\boldsymbol{y} = \mathrm{Im}\{\boldsymbol{z}\}$. *The* cogradient operator $\frac{\partial}{\partial \boldsymbol{z}}$ *and* conjugate cogradient operator $\frac{\partial}{\partial \overline{\boldsymbol{z}}}$ *are defined as [2, 21, 26, 36]*

$$\frac{\partial}{\partial \boldsymbol{z}} \triangleq \frac{1}{2} \begin{bmatrix} \frac{\partial}{\partial x_1} - \frac{\partial}{\partial y_1} i \\ \vdots \\ \frac{\partial}{\partial x_n} - \frac{\partial}{\partial y_n} i \end{bmatrix}, \tag{2.3a}$$

$$\frac{\partial}{\partial \overline{\boldsymbol{z}}} \triangleq \frac{1}{2} \begin{bmatrix} \frac{\partial}{\partial x_1} + \frac{\partial}{\partial y_1} i \\ \vdots \\ \frac{\partial}{\partial x_n} + \frac{\partial}{\partial y_n} i \end{bmatrix}. \tag{2.3b}$$

The (conjugate) cogradient operator acts as a partial derivative with respect to $\boldsymbol{z}$ ($\overline{\boldsymbol{z}}$), treating $\overline{\boldsymbol{z}}$ ($\boldsymbol{z}$) as a constant. To see this, let $z \in \mathbb{C}$ and let $x = \mathrm{Re}\{z\}$ and $y = \mathrm{Im}\{z\}$ so that $x = \frac{1}{2}(z + \overline{z})$ and $y = \frac{i}{2}(\overline{z} - z)$. Then for a function $f : \mathbb{C} \to \mathbb{C}$ we have that $\frac{\partial f}{\partial z} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial z} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial z}$, which is equal to $\frac{1}{2}(\frac{\partial f}{\partial x} - \frac{\partial f}{\partial y} i)$ if $\frac{\partial \overline{z}}{\partial z}$ is set to zero. Using the cogradient operator, we note that the Cauchy–Riemann conditions for f to be analytic in z can be written compactly as $\frac{\partial f}{\partial \overline{z}} = 0$, i.e., f is function only of z. Analogously, f is analytic in $\overline{z}$ if and only if $\frac{\partial f}{\partial z} = 0$.

Although their definitions often allow the cogradients to be expressed elegantly in terms of $\boldsymbol{z}$ and $\overline{\boldsymbol{z}}$, neither contains enough information by itself to express the change in a function with respect to a change $\boldsymbol{z}$. This motivates the following definition of a complex gradient operator.

DEFINITION 2.2. *Let* $\boldsymbol{z} \in \mathbb{C}^n$. *We define the* complex gradient operator $\frac{\partial}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}}$ *as*

$$\frac{\partial}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}} \triangleq \left(\frac{\partial}{\partial \boldsymbol{z}}, \frac{\partial}{\partial \overline{\boldsymbol{z}}} \right). \tag{2.4}$$

The linear map (2.1) can be used to express the relation between the complex gradient $\frac{\partial}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}}$ and the real gradient $\frac{\partial}{\partial \overset{\mathcal{R}}{\boldsymbol{z}}}$. We have that

$$\frac{\partial}{\partial \overset{\mathcal{R}}{\boldsymbol{z}}} = J^{\mathrm{T}} \frac{\partial}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}}. \tag{2.5}$$

Similarly, the real Hessian $\frac{\partial^2}{\partial \overset{\mathcal{R}}{\boldsymbol{z}} \partial \overset{\mathcal{R}}{\boldsymbol{z}}^{\mathrm{T}}}$ can be transformed into several complex Hessians, two of which are

$$\frac{\partial^2}{\partial \overset{\mathcal{R}}{\boldsymbol{z}} \partial \overset{\mathcal{R}}{\boldsymbol{z}}^{\mathrm{T}}} = \frac{\partial}{\partial \overset{\mathcal{R}}{\boldsymbol{z}}} \left(\frac{\partial}{\partial \overset{\mathcal{R}}{\boldsymbol{z}}} \right)^{\mathrm{T}} = J^{\mathrm{T}} \frac{\partial}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}} \left(\frac{\partial}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}} J \right) = J^{\mathrm{T}} \frac{\partial^2}{\partial \overset{\mathcal{C}}{\boldsymbol{z}} \partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}} J, \tag{2.6a}$$

$$\frac{\partial^2}{\partial \overset{\mathcal{R}}{\boldsymbol{z}} \partial \overset{\mathcal{R}}{\boldsymbol{z}}^{\mathrm{T}}} = \frac{\partial}{\partial \overset{\mathcal{R}}{\boldsymbol{z}}} \left(\frac{\partial}{\partial \overset{\mathcal{R}}{\boldsymbol{z}}} \right)^{\mathrm{T}} = J^{\mathrm{H}} \frac{\partial}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}} \left(\frac{\partial}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}} J \right) = J^{\mathrm{H}} \frac{\partial^2}{\partial \overset{\mathcal{C}}{\boldsymbol{z}} \partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}} J. \tag{2.6b}$$

2.2. First-order complex Taylor series expansion. Consider the first-order real Taylor series expansion of a function $F : \mathcal{R} \to \mathbb{C}^m$,

$$m_F(\Delta \overset{\mathcal{R}}{\boldsymbol{z}}) = F(\overset{\mathcal{R}}{\boldsymbol{z}}) + \frac{\partial F(\overset{\mathcal{R}}{\boldsymbol{z}})}{\partial \overset{\mathcal{R}}{\boldsymbol{z}}^{\mathrm{T}}} \Delta \overset{\mathcal{R}}{\boldsymbol{z}}. \tag{2.7}$$

Because $\mathcal{R}$ and $\mathcal{C}$ are isomorphic, the function F can also be regarded as a function of $\overset{\mathcal{C}}{\boldsymbol{z}}$. Although it is generally not true that F is analytic in $\boldsymbol{z}$ and $\overline{\boldsymbol{z}}$ independently if

F is analytic in $\overset{\mathcal{R}}{\boldsymbol{z}}$, it does hold that F is analytic in $\boldsymbol{z}$ and $\overline{\boldsymbol{z}}$ as a whole. Using (2.1) and (2.5), the first-order complex Taylor series expansion of F can be expressed as

$$m_F(\Delta\overset{\mathcal{C}}{\boldsymbol{z}}) = F(\overset{\mathcal{C}}{\boldsymbol{z}}) + \frac{\partial F(\overset{\mathcal{C}}{\boldsymbol{z}})}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}}\Delta\overset{\mathcal{C}}{\boldsymbol{z}}. \tag{2.8}$$

The matrix $\frac{\partial F}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}} = \begin{bmatrix}\frac{\partial F}{\partial \boldsymbol{z}^{\mathrm{T}}} & \frac{\partial F}{\partial \overline{\boldsymbol{z}}^{\mathrm{T}}}\end{bmatrix}$ is obtained by applying the transpose of the complex gradient operator (2.5) component-wise to F. The matrices $\frac{\partial F}{\partial \boldsymbol{z}^{\mathrm{T}}}$ and $\frac{\partial F}{\partial \overline{\boldsymbol{z}}^{\mathrm{T}}}$ are called the *Jacobian* and *conjugate Jacobian*, respectively.

2.3. Second-order complex Taylor series expansion. Consider the second-order real Taylor series expansion of a function $f : \mathcal{R} \to \mathbb{C}$,

$$m_f(\Delta\overset{\mathcal{R}}{\boldsymbol{z}}) = f(\overset{\mathcal{R}}{\boldsymbol{z}}) + \Delta\overset{\mathcal{R}}{\boldsymbol{z}}^{\mathrm{T}}\frac{\partial f(\overset{\mathcal{R}}{\boldsymbol{z}})}{\partial \overset{\mathcal{R}}{\boldsymbol{z}}} + \frac{1}{2}\Delta\overset{\mathcal{R}}{\boldsymbol{z}}^{\mathrm{T}}\frac{\partial^2 f(\overset{\mathcal{R}}{\boldsymbol{z}})}{\partial \overset{\mathcal{R}}{\boldsymbol{z}}\partial \overset{\mathcal{R}}{\boldsymbol{z}}^{\mathrm{T}}}\Delta\overset{\mathcal{R}}{\boldsymbol{z}}. \tag{2.9}$$

Because $\mathcal{R}$ and $\mathcal{C}$ are isomorphic, the function f can also be regarded as a function of $\overset{\mathcal{C}}{\boldsymbol{z}}$. Using (2.1), (2.5) and (2.6), the second-order Taylor series expansion of $f(\overset{\mathcal{C}}{\boldsymbol{z}})$ can be expressed in the following two equivalent ways

$$m_f(\Delta\overset{\mathcal{C}}{\boldsymbol{z}}) = f(\overset{\mathcal{C}}{\boldsymbol{z}}) + \Delta\overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}\frac{\partial f(\overset{\mathcal{C}}{\boldsymbol{z}})}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}} + \frac{1}{2}\Delta\overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}\frac{\partial^2 f(\overset{\mathcal{C}}{\boldsymbol{z}})}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}\partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}}\Delta\overset{\mathcal{C}}{\boldsymbol{z}}, \tag{2.10a}$$

$$m_f(\Delta\overset{\mathcal{C}}{\boldsymbol{z}}) = f(\overset{\mathcal{C}}{\boldsymbol{z}}) + \Delta\overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}\frac{\partial f(\overset{\mathcal{C}}{\boldsymbol{z}})}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}} + \frac{1}{2}\Delta\overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{H}}\frac{\partial^2 f(\overset{\mathcal{C}}{\boldsymbol{z}})}{\partial \overset{\mathcal{C}}{\overline{\boldsymbol{z}}}\partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}}\Delta\overset{\mathcal{C}}{\boldsymbol{z}}. \tag{2.10b}$$

Often $f(\overset{\mathcal{R}}{\boldsymbol{z}})$ will have continuous second-order derivatives. Clairaut's theorem [18] states that the order of differentiation does not matter for functions for which this property holds, and hence that the real Hessian $\frac{\partial^2 f}{\partial \overset{\mathcal{R}}{\boldsymbol{z}}\partial \overset{\mathcal{R}}{\boldsymbol{z}}^{\mathrm{T}}}$ is symmetric. It then follows from (2.6) that $\frac{\partial^2 f}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}\partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}}$ is symmetric and $\frac{\partial^2 f}{\partial \overset{\mathcal{C}}{\overline{\boldsymbol{z}}}\partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}}$ is Hermitian.

Note that (2.10b) is not differentiable with respect to $\Delta\overset{\mathcal{C}}{\boldsymbol{z}}$ because of the dependency on $\Delta\overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{H}}$. In contrast, (2.10a) is differentiable with respect to $\Delta\overset{\mathcal{C}}{\boldsymbol{z}}$. To overcome this apparent difficulty, we note that the dependency on $\Delta\overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{H}}$ can be resolved by substituting $\Delta\overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{H}} = \Delta\overset{\mathcal{C}}{\overline{\boldsymbol{z}}}^{\mathrm{T}} = \Delta\overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}S$. The complex gradient of the second-order Taylor expansions (2.10) can now be computed as

$$\frac{\partial m_f(\Delta\overset{\mathcal{C}}{\boldsymbol{z}})}{\partial \Delta\overset{\mathcal{C}}{\boldsymbol{z}}} = \frac{\partial f(\overset{\mathcal{C}}{\boldsymbol{z}})}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}} + \frac{\partial^2 f(\overset{\mathcal{C}}{\boldsymbol{z}})}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}\partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}}\Delta\overset{\mathcal{C}}{\boldsymbol{z}} \text{ and} \tag{2.11a}$$

$$\frac{\partial m_f(\Delta\overset{\mathcal{C}}{\boldsymbol{z}})}{\partial \Delta\overset{\mathcal{C}}{\boldsymbol{z}}} = \frac{\partial f(\overset{\mathcal{C}}{\boldsymbol{z}})}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}} + \frac{1}{2}\left(S\frac{\partial^2 f(\overset{\mathcal{C}}{\boldsymbol{z}})}{\partial \overset{\mathcal{C}}{\overline{\boldsymbol{z}}}\partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}} + \left(\frac{\partial^2 f(\overset{\mathcal{C}}{\boldsymbol{z}})}{\partial \overset{\mathcal{C}}{\overline{\boldsymbol{z}}}\partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}}\right)^{\mathrm{T}} S\right)\Delta\overset{\mathcal{C}}{\boldsymbol{z}}, \tag{2.11b}$$

respectively.

Let us now assume f maps to the real numbers. This assumption has certain consequences for the structure of the gradient and Hessian in the Taylor series expansions (2.10). First, because of the identity $\frac{\partial \overline{f}}{\partial \boldsymbol{z}} = \overline{\frac{\partial f}{\partial \overline{\boldsymbol{z}}}}$ [26, 36], it follows that $\frac{\partial f}{\partial \boldsymbol{z}} = \overline{\frac{\partial f}{\partial \overline{\boldsymbol{z}}}}$ for real-valued f. Second, it is also not hard to show that $\frac{\partial^2 f}{\partial \boldsymbol{z}\partial \overline{\boldsymbol{z}}^{\mathrm{T}}} = \overline{\frac{\partial^2 f}{\partial \overline{\boldsymbol{z}}\partial \boldsymbol{z}^{\mathrm{T}}}}$ and that $\frac{\partial^2 f}{\partial \boldsymbol{z}\partial \boldsymbol{z}^{\mathrm{T}}} = \overline{\frac{\partial^2 f}{\partial \overline{\boldsymbol{z}}\partial \overline{\boldsymbol{z}}^{\mathrm{T}}}}$. Using these properties, we can simplify expression (2.11b) by premul-

tiplying with S, so that

$$S\frac{\partial m_f(\Delta \overset{c}{\boldsymbol{z}})}{\partial \Delta \overset{c}{\boldsymbol{z}}} = S\frac{\partial f(\overset{c}{\boldsymbol{z}})}{\partial \overset{c}{\boldsymbol{z}}} + \frac{1}{2}\left(SS\frac{\partial^2 f(\overset{c}{\boldsymbol{z}})}{\partial \overset{c}{\overline{\boldsymbol{z}}} \partial \overset{c}{\boldsymbol{z}}^{\mathrm{T}}} + S\left(\frac{\partial^2 f(\overset{c}{\boldsymbol{z}})}{\partial \overset{c}{\overline{\boldsymbol{z}}} \partial \overset{c}{\overline{\boldsymbol{z}}}^{\mathrm{T}}}\right)^{\mathrm{T}} S \right) \Delta \overset{c}{\boldsymbol{z}}$$

$$\overline{\frac{\partial m_f(\Delta \overset{c}{\boldsymbol{z}})}{\partial \Delta \overset{c}{\boldsymbol{z}}}} = \overline{\frac{\partial f(\overset{c}{\boldsymbol{z}})}{\partial \overset{c}{\boldsymbol{z}}}} + \frac{\partial^2 f(\overset{c}{\boldsymbol{z}})}{\partial \overset{c}{\overline{\boldsymbol{z}}} \partial \overset{c}{\boldsymbol{z}}^{\mathrm{T}}} \Delta \overset{c}{\boldsymbol{z}}. \tag{2.12}$$

3. Nonlinear optimization problems $\min_{\boldsymbol{z}} f(\boldsymbol{z}, \overline{\boldsymbol{z}})$. First we consider nonlinear optimization problems of the form (1.1). The space spanned by $\boldsymbol{z}$ and $\overline{\boldsymbol{z}}$ is equal to $\mathcal{C}$, which is equivalent to $\mathcal{R}$ under the linear map (2.1). It is then understood that the function f can also be regarded as a function that maps $\mathcal{C}$ or $\mathcal{R}$ to $\mathbb{R}$, so that both the second-order model (2.9) in $\overset{\mathcal{R}}{\boldsymbol{z}}$ and the two second-order models (2.10) in $\overset{c}{\boldsymbol{z}}$ are applicable.

3.1. The generalized (limited memory) BFGS method. In the generalized BFGS method, we use the quadratic model

$$m_k(\overset{c}{\boldsymbol{p}}) = f(\overset{c}{\boldsymbol{z}}_k) + \overset{c}{\boldsymbol{p}}^{\mathrm{T}} \frac{\partial f(\overset{c}{\boldsymbol{z}}_k)}{\partial \overset{c}{\boldsymbol{z}}} + \frac{1}{2} \overset{c}{\boldsymbol{p}}^{\mathrm{H}} B_k \overset{c}{\boldsymbol{p}} \tag{3.1}$$

of the objective function f at the current iterate $\overset{c}{\boldsymbol{z}}_k$, where B_k is a Hermitian positive definite matrix that is updated every iteration. Since f is real-valued, the minimizer of this convex quadratic model can be obtained by setting the conjugate complex gradient (2.12) equal to zero, and is given by

$$\overset{c}{\boldsymbol{p}}_k = -B_k^{-1} \overline{\frac{\partial f(\overset{c}{\boldsymbol{z}}_k)}{\partial \overset{c}{\boldsymbol{z}}}}. \tag{3.2}$$

In a line search framework, the next iterate is $\boldsymbol{z}_{k+1} = \boldsymbol{z}_k + \alpha_k \boldsymbol{p}_k$, where the real step length α_k is usually chosen to satisfy the (strong) Wolfe conditions [28, 42]. A reasonable requirement for the updated Hessian B_{k+1} is that the gradient of the model m_{k+1} matches the gradient of the objective function f in the last two iterates $\overset{c}{\boldsymbol{z}}_k$ and $\overset{c}{\boldsymbol{z}}_{k+1}$. The second condition is satisfied automatically by (3.1). The first condition can be written as

$$\overline{\frac{\partial m_{k+1}(-\alpha_k \overset{c}{\boldsymbol{p}}_k)}{\partial \overset{c}{\boldsymbol{p}}}} = \overline{\frac{\partial f(\overset{c}{\boldsymbol{z}}_{k+1})}{\partial \overset{c}{\boldsymbol{z}}}} - \alpha_k B_{k+1} \overset{c}{\boldsymbol{p}}_k = \overline{\frac{\partial f(\overset{c}{\boldsymbol{z}}_k)}{\partial \overset{c}{\boldsymbol{z}}}}.$$

Define $\overset{c}{\boldsymbol{y}}_k \triangleq \overline{\frac{\partial f(\overset{c}{\boldsymbol{z}}_{k+1})}{\partial \overset{c}{\boldsymbol{z}}}} - \overline{\frac{\partial f(\overset{c}{\boldsymbol{z}}_k)}{\partial \overset{c}{\boldsymbol{z}}}}$ and $\overset{c}{\boldsymbol{s}}_k \triangleq \overset{c}{\boldsymbol{z}}_{k+1} - \overset{c}{\boldsymbol{z}}_k$, then by rearranging we find that B_{k+1} should satisfy the *secant equation*

$$B_{k+1} \overset{c}{\boldsymbol{s}}_k = \overset{c}{\boldsymbol{y}}_k. \tag{3.3}$$

The BFGS update [3, 9, 13, 38] is chosen so that the inverse of the updated Hessian B_{k+1}^{-1} is, among all symmetric positive definite matrices satisfying the secant equation (3.3), in some sense closest to the current inverse Hessian approximation B_k^{-1}. In Lemma 3.2 and Theorem 3.3, we generalize Schnabel and Dennis' derivation of the BFGS update [37] to Hermitian Hessians. The proof requires the Broyden update [8], which is also generalized to Hermitian matrices in Theorem 3.1.

THEOREM 3.1. *Let $B \in \mathbb{C}^{n\times n}$, $\boldsymbol{y}, \boldsymbol{s} \in \mathbb{C}^n$ and $\boldsymbol{s}$ nonzero. Then the unique minimizer of*

$$\begin{aligned} &\min_{\hat{B}} \|B - \hat{B}\|_F \\ &s.t.\ \hat{B}\boldsymbol{s} = \boldsymbol{y} \end{aligned} \tag{3.4}$$

is given by the Broyden update

$$B^* = B + \frac{(\boldsymbol{y} - B\boldsymbol{s})\boldsymbol{s}^{\mathrm{H}}}{\boldsymbol{s}^{\mathrm{H}}\boldsymbol{s}}. \tag{3.5}$$

Proof. To show that B^* is a solution to (3.4), note that $B^*\boldsymbol{s} = \boldsymbol{y}$ and that

$$\|B - B^*\|_F = \left\|(B - \hat{B})\frac{\boldsymbol{s}\boldsymbol{s}^H}{\boldsymbol{s}^H\boldsymbol{s}}\right\|_F \leq \|B - \hat{B}\|_F.$$

That B^* is the unique solution follows from the fact that the mapping $f : \mathbb{C}^{n\times n} \to \mathbb{R}$ defined by $f(\hat{B}) = \|B - \hat{B}\|_F$ is strictly convex in $\mathbb{C}^{n\times n}$ and that the set of $\hat{B} \in \mathbb{C}^{n\times n}$ such that $\hat{B}\boldsymbol{s} = \boldsymbol{y}$ is convex. ∎

LEMMA 3.2. *Let $\boldsymbol{y}, \boldsymbol{s} \in \mathbb{C}^n$, $\boldsymbol{s}$ nonzero and let $Q(\boldsymbol{y}, \boldsymbol{s}) \triangleq \{B \in \mathbb{C}^{n\times n} | B\boldsymbol{s} = \boldsymbol{y}\}$. Then the* generalized quotient *$Q(\boldsymbol{y}, \boldsymbol{s})$ contains a Hermitian positive definite matrix if and only if for some nonzero $\boldsymbol{v} \in \mathbb{C}^n$ and nonsingular $J \in \mathbb{C}^{n\times n}$, $\boldsymbol{y} = J\boldsymbol{v}$ and $\boldsymbol{v} = J^{\mathrm{H}}\boldsymbol{s}$.*

Proof. If $\boldsymbol{v}$ and J exist then $\boldsymbol{y} = J\boldsymbol{v} = JJ^{\mathrm{H}}\boldsymbol{s}$ and JJ^{H} is a Hermitian positive definite matrix in $Q(\boldsymbol{y}, \boldsymbol{s})$. Now suppose B is a Hermitian positive definite matrix which satisfies $B\boldsymbol{s} = \boldsymbol{y}$. Let $B = LL^{\mathrm{H}}$ be the Cholesky factorization of B and set $J = L$ and $\boldsymbol{v} = L^{\mathrm{H}}\boldsymbol{s}$ to complete the proof. ∎

THEOREM 3.3. *Let $L \in \mathbb{C}^{n\times n}$ be nonsingular, $B = LL^{\mathrm{H}}$, $\boldsymbol{y}, \boldsymbol{s} \in \mathbb{C}^n$ and $\boldsymbol{s}$ nonzero. There is a Hermitian positive definite matrix $\hat{B} \in Q(\boldsymbol{y}, \boldsymbol{s})$ if and only if $\boldsymbol{y}^{\mathrm{H}}\boldsymbol{s} > 0$. If there is such a matrix, then the* generalized BFGS update *$B^* \triangleq \hat{L}\hat{L}^{\mathrm{H}}$ is one such, where*

$$\hat{L} = \left(\mathbb{I}_{n\times n} \pm \sqrt{\frac{\boldsymbol{y}^{\mathrm{H}}\boldsymbol{s}}{\boldsymbol{s}^{\mathrm{H}}B\boldsymbol{s}}}\frac{\boldsymbol{y}\boldsymbol{s}^{\mathrm{H}}}{\boldsymbol{y}^{\mathrm{H}}\boldsymbol{s}} - \frac{B\boldsymbol{s}\boldsymbol{s}^{\mathrm{H}}}{\boldsymbol{s}^{\mathrm{H}}B\boldsymbol{s}}\right)L, \tag{3.6}$$

so that

$$B^* = B - \frac{B\boldsymbol{s}\boldsymbol{s}^{\mathrm{H}}B}{\boldsymbol{s}^{\mathrm{H}}B\boldsymbol{s}} + \frac{\boldsymbol{y}\boldsymbol{y}^{\mathrm{H}}}{\boldsymbol{y}^{\mathrm{H}}\boldsymbol{s}} \tag{3.7}$$

and

$$B^{*-1} = \left(\mathbb{I}_{n\times n} - \frac{\boldsymbol{s}\boldsymbol{y}^{\mathrm{H}}}{\boldsymbol{y}^{\mathrm{H}}\boldsymbol{s}}\right)B^{-1}\left(\mathbb{I}_{n\times n} - \frac{\boldsymbol{y}\boldsymbol{s}^{\mathrm{H}}}{\boldsymbol{y}^{\mathrm{H}}\boldsymbol{s}}\right) + \frac{\boldsymbol{s}\boldsymbol{s}^{\mathrm{H}}}{\boldsymbol{y}^{\mathrm{H}}\boldsymbol{s}}. \tag{3.8}$$

Proof. Let $\hat{B}$ be a Hermitian positive definite matrix in $Q(\boldsymbol{y}, \boldsymbol{s})$, then $\boldsymbol{s}^{\mathrm{H}}\boldsymbol{y} = \boldsymbol{s}^{\mathrm{H}}\hat{B}\boldsymbol{s} > 0$ is a necessary condition for the update to exist. The nearest matrix $\hat{L}$ in Frobenius norm to L that satisfies $\hat{L}\boldsymbol{v} = \boldsymbol{y}$ is given by the Broyden update (3.5)

$$\hat{L} = L + \frac{(\boldsymbol{y} - L\boldsymbol{v})\boldsymbol{v}^{\mathrm{H}}}{\boldsymbol{v}^{\mathrm{H}}\boldsymbol{v}}.$$

If we can find a $\boldsymbol{v} \in \mathbb{C}^n$ such that $\boldsymbol{v} = \hat{L}^{\mathrm{H}}\boldsymbol{s}$, then by Lemma 3.2 we know that $\hat{L}\hat{L}^{\mathrm{H}}$ is in $Q(\boldsymbol{y}, \boldsymbol{s})$. The condition

$$\boldsymbol{v} = \hat{L}^{\mathrm{H}}\boldsymbol{s} = L^{\mathrm{H}}\boldsymbol{s} + \frac{(\boldsymbol{y}^{\mathrm{H}}\boldsymbol{s} - \boldsymbol{v}^{\mathrm{H}}L^{\mathrm{H}}\boldsymbol{s})}{\boldsymbol{v}^{\mathrm{H}}\boldsymbol{v}}\boldsymbol{v}$$

implies that $\boldsymbol{v} = \alpha L^{\mathrm{H}}\boldsymbol{s}$ for some scalar α. Plugging back in, we find

$$\alpha = 1 + \frac{\alpha\boldsymbol{y}^{\mathrm{H}}\boldsymbol{s} - |\alpha|^2\boldsymbol{s}^{\mathrm{H}}B\boldsymbol{s}}{|\alpha|^2\boldsymbol{s}^{\mathrm{H}}B\boldsymbol{s}}$$

or

$$|\alpha|^2 = \frac{\boldsymbol{y}^{\mathrm{H}}\boldsymbol{s}}{\boldsymbol{s}^{\mathrm{H}}B\boldsymbol{s}},$$

which shows that $\boldsymbol{y}^{\mathrm{H}}\boldsymbol{s} > 0$ is a sufficient condition for the update to exist. Filling $\boldsymbol{v} = \alpha L^{\mathrm{H}}\boldsymbol{s}$ back in the Broyden update results in (3.6). It is then a matter of verifying (3.7-3.8) by forming the product $\hat{L}\hat{L}^{\mathrm{H}}$ and applying the Sherman-Morrison formula [39, 40], respectively. □

Theorem 3.3 holds for any $\boldsymbol{s}$ and $\boldsymbol{y}$ in $\mathbb{C}^{2n}$ if all dimensions are scaled appropriately, and hence also for the subset $\mathcal{C} \subset \mathbb{C}^{2n}$. At every step of the generalized BFGS method, the inverse Hessian can then be updated by (3.8), so that

$$B_{k+1}^{-1} = \left(\mathbb{I}_{2n\times 2n} - \frac{\overset{\mathcal{C}}{\boldsymbol{s}}_k\overset{\mathcal{C}}{\boldsymbol{y}}_k^{\mathrm{H}}}{\overset{\mathcal{C}}{\boldsymbol{y}}_k^{\mathrm{H}}\overset{\mathcal{C}}{\boldsymbol{s}}_k}\right) B_k^{-1} \left(\mathbb{I}_{2n\times 2n} - \frac{\overset{\mathcal{C}}{\boldsymbol{y}}_k\overset{\mathcal{C}}{\boldsymbol{s}}_k^{\mathrm{H}}}{\overset{\mathcal{C}}{\boldsymbol{y}}_k^{\mathrm{H}}\overset{\mathcal{C}}{\boldsymbol{s}}_k}\right) + \frac{\overset{\mathcal{C}}{\boldsymbol{s}}_k\overset{\mathcal{C}}{\boldsymbol{s}}_k^{\mathrm{H}}}{\overset{\mathcal{C}}{\boldsymbol{y}}_k^{\mathrm{H}}\overset{\mathcal{C}}{\boldsymbol{s}}_k}.$$

If the number of variables n is large, the cost of storing and manipulating the inverse Hessian B_{k+1}^{-1} can become prohibitive. The limited-memory BFGS method [12, 22, 27] circumvents this problem by storing the inverse Hessian implicitly as a set of m vector pairs $\{\overset{\mathcal{C}}{\boldsymbol{s}}_i, \overset{\mathcal{C}}{\boldsymbol{y}}_i\}$, $i = k-m, \ldots, k-1$. In fact, it suffices to store $\{\boldsymbol{s}_i, \boldsymbol{y}_i\}$, since the second half of any vector in $\mathcal{C}$ is just the complex conjugate of its first half.

Suppose we have a real function of complex variables $f : \mathcal{C} \to \mathbb{R}$ that we are interested in minimizing for $\boldsymbol{z} \in \mathbb{C}^n$ as well as for $\boldsymbol{x} \in \mathbb{R}^n$. The quasi-Newton step (3.2) depends on the complex gradient. Because the conjugate cogradient is just the complex conjugate of the cogradient for real-valued f, we need only compute one of the cogradients, say the conjugate cogradient $\frac{\partial f(\boldsymbol{x}_k)}{\partial \overline{\boldsymbol{z}}}$. For $\boldsymbol{x} \in \mathbb{R}^n$ we need the real gradient $\frac{\partial f(\boldsymbol{x}_k)}{\partial \boldsymbol{x}}$, which can also be expressed as $\frac{\partial f(\boldsymbol{x}_k)}{\partial \boldsymbol{z}} + \frac{\partial f(\boldsymbol{x}_k)}{\partial \overline{\boldsymbol{z}}} = 2\frac{\partial f(\boldsymbol{x}_k)}{\partial \overline{\boldsymbol{z}}}$. Therefore, by constructing complex optimization algorithms in a way that only requires evaluating objective function values $f_k \triangleq f(\overset{\mathcal{C}}{\boldsymbol{z}}_k)$ and scaled conjugate cogradients $\boldsymbol{g}_k \triangleq 2\frac{\partial f(\overset{\mathcal{C}}{\boldsymbol{z}}_k)}{\partial \overline{\boldsymbol{z}}}$, the function can be minimized over $\mathbb{C}^n$ as well as $\mathbb{R}^n$ without a separate implementation of the real gradient $\frac{\partial f(\boldsymbol{x}_k)}{\partial \boldsymbol{x}}$.

In Algorithm 3.1 a generalized limited-memory BFGS two-loop recursion for computing the quasi-Newton step (3.2) is presented. Its computational and storage cost are equal to that of the limited-memory BFGS method applied in the real domain. However, the generalized method requires the objective function's gradient in a form that may be more intuitive to the user, and can furthermore minimize the same objective function over $\mathbb{R}^n$ without requiring the real gradient to be treated as a separate case.

The quasi-Newton step $\boldsymbol{p}_k^{\mathrm{QN}}$ computed by Algorithm 3.1 can be used in either a line search or trust-region framework. In the former, the next iterate is $\boldsymbol{z}_{k+1} =$

Input: $\boldsymbol{g}_k \triangleq 2\overline{\frac{\partial f(\overset{c}{\boldsymbol{z}}_k)}{\partial \boldsymbol{z}}} = 2\frac{\partial f(\overset{c}{\boldsymbol{z}}_k)}{\partial \overline{\boldsymbol{z}}}$,
$\boldsymbol{s}_i \triangleq \boldsymbol{z}_{i+1} - \boldsymbol{z}_i$,
$\boldsymbol{y}_i \triangleq \boldsymbol{g}_{i+1} - \boldsymbol{g}_i$ and
$\rho_i \triangleq \mathrm{Re}\{\boldsymbol{y}_i^{\mathrm{H}}\boldsymbol{s}_i\}^{-1}$ for $i = k-m, \ldots, k-1$

Output: $\boldsymbol{p}$ where $\overset{c}{\boldsymbol{p}} = -B_k^{-1}\overline{\frac{\partial f(\overset{c}{\boldsymbol{z}}_k)}{\partial \overset{c}{\boldsymbol{z}}}}$

$\boldsymbol{p} \leftarrow -\boldsymbol{g}_k$
for $i = k-1, k-2, \ldots, k-m$ **do**
  $\alpha_i \leftarrow \rho_i \mathrm{Re}\{\boldsymbol{s}_i^{\mathrm{H}}\boldsymbol{p}\}$
  $\boldsymbol{p} \leftarrow \boldsymbol{p} - \alpha_i \boldsymbol{y}_i$
end
$\boldsymbol{p} \leftarrow \frac{1}{2}B_{k-m}^{-1}\boldsymbol{p}$ (e.g., $\frac{1}{2}B_{k-m}^{-1} = \frac{\mathrm{Re}\{\boldsymbol{y}_{k-1}^{\mathrm{H}}\boldsymbol{s}_{k-1}\}}{\boldsymbol{y}_{k-1}^{\mathrm{H}}\boldsymbol{y}_{k-1}}\mathbb{I}_{n\times n}$ [12])
for $i = k-m, k-m+1, \ldots, k-1$ **do**
  $\beta \leftarrow \rho_i \mathrm{Re}\{\boldsymbol{y}_i^{\mathrm{H}}\boldsymbol{p}\}$
  $\boldsymbol{p} \leftarrow \boldsymbol{p} + (\alpha_i - \beta)\boldsymbol{s}_i$
end

ALGORITHM 3.1. *Generalized L-BFGS two-loop recursion.*

$\boldsymbol{z}_k + \alpha_k \boldsymbol{p}_k^{\mathrm{QN}}$. The step length α_k is usually chosen to loosely minimize the one-dimensional optimization problem $\min_{\alpha_k} f(\overset{c}{\boldsymbol{z}}_k + \alpha_k \overset{c}{\boldsymbol{p}}_k)$ so that the *Wolfe conditions* [28, 42]

$$f(\overset{c}{\boldsymbol{z}}_k + \alpha_k \overset{c}{\boldsymbol{p}}_k) \leq f(\overset{c}{\boldsymbol{z}}_k) + c_1 \alpha_k \overset{c\mathrm{T}}{\boldsymbol{p}_k} \frac{\partial f(\overset{c}{\boldsymbol{z}}_k)}{\partial \overset{c}{\boldsymbol{z}}} \text{ and} \tag{3.9a}$$

$$\overset{c\mathrm{T}}{\boldsymbol{p}_k} \frac{\partial f(\overset{c}{\boldsymbol{z}}_k + \alpha_k \overset{c}{\boldsymbol{p}}_k)}{\partial \overset{c}{\boldsymbol{z}}} \geq c_2 \overset{c\mathrm{T}}{\boldsymbol{p}_k} \frac{\partial f(\overset{c}{\boldsymbol{z}}_k)}{\partial \overset{c}{\boldsymbol{z}}}, \tag{3.9b}$$

where $0 < c_1 < c_2 < 1$, are satisfied. Inequalities (3.9a) and (3.9b) are known as the *sufficient decrease* and *curvature condition*, respectively. The former ensures the objective function is sufficiently smaller at the next iterate, and the latter ensures convergence of the gradient to zero. Furthermore, the curvature condition is a sufficient condition for the BFGS update to exist since

$$\overset{c\mathrm{H}}{\boldsymbol{s}_k} \overset{c}{\boldsymbol{y}_k} = \alpha_k \overset{c\mathrm{H}}{\boldsymbol{p}_k} \left(\overline{\frac{\partial f(\overset{c}{\boldsymbol{z}}_k + \alpha_k \overset{c}{\boldsymbol{p}}_k)}{\partial \overset{c}{\boldsymbol{z}}}} - \overline{\frac{\partial f(\overset{c}{\boldsymbol{z}}_k)}{\partial \overset{c}{\boldsymbol{z}}}} \right) \geq (c_2 - 1)\alpha_k \overset{c\mathrm{T}}{\boldsymbol{p}_k} \frac{\partial f(\overset{c}{\boldsymbol{z}}_k)}{\partial \overset{c}{\boldsymbol{z}}} > 0.$$

A step length may satisfy the Wolfe conditions without being particularly close to a minimizer of $f(\overset{c}{\boldsymbol{z}}_k + \alpha_k \overset{c}{\boldsymbol{p}}_k)$. In the strong Wolfe conditions, the curvature condition is replaced by

$$\left| \overset{c\mathrm{T}}{\boldsymbol{p}_k} \frac{\partial f(\overset{c}{\boldsymbol{z}}_k + \alpha_k \overset{c}{\boldsymbol{p}}_k)}{\partial \overset{c}{\boldsymbol{z}}} \right| \leq c_2 \left| \overset{c\mathrm{T}}{\boldsymbol{p}_k} \frac{\partial f(\overset{c}{\boldsymbol{z}}_k)}{\partial \overset{c}{\boldsymbol{z}}} \right|$$

so that points far from stationary points are excluded. Line search algorithms are an integral part of quasi-Newton methods, but can be difficult to implement. There are several good software implementations available in the public domain, such as Moré and Thuente [23] and Hager and Zhang [14], which require little effort to adapt to

minimization of functions in complex variables since they often only require function evaluations and derivative information along a given descent direction.

In a trust-region framework, a region around the current iterate $\boldsymbol{z}_k$ is defined in which the model m_k is trusted to be an adequate representation of the objective function. The next iterate $\boldsymbol{z}_{k+1}$ is then chosen to be the approximate minimizer of the model in this region. In effect, the direction and length of the step is chosen simultaneously. The trust-region radius Δ_k is updated every iteration based on the trustworthiness ρ_k of the model, which is defined as the ratio of the actual reduction $f(\overset{\mathcal{C}}{\boldsymbol{z}}_k) - f(\overset{\mathcal{C}}{\boldsymbol{z}}_k + \overset{\mathcal{C}}{\boldsymbol{p}}_k)$ and the predicted reduction $m_k(\overset{\mathcal{C}}{\boldsymbol{z}}_k) - m_k(\overset{\mathcal{C}}{\boldsymbol{z}}_k + \overset{\mathcal{C}}{\boldsymbol{p}}_k)$.

There exist several strategies to approximately solve the trust-region subproblem

$$\begin{aligned} &\min_{\boldsymbol{p}} m_k(\overset{\mathcal{C}}{\boldsymbol{p}}) \\ &\text{s.t. } \|\boldsymbol{p}\| \leq \Delta_k \end{aligned} \tag{3.10}$$

The quasi-Newton step $\boldsymbol{p}_k^{\mathrm{QN}}$ minimizes (3.10) when $\|\boldsymbol{p}_k^{\mathrm{QN}}\| \leq \Delta_k$. If the trust-region radius is small compared to the quasi-Newton step, the quadratic term in m_k has little effect and the solution to the trust-region subproblem can be approximated by $\boldsymbol{p}^* = -\Delta_k \frac{\boldsymbol{g}_k}{\|\boldsymbol{g}_k\|}$. The dogleg method [31, 32], double-dogleg method [7] and two-dimensional subspace minimization [4] all attempt to approximately minimize (3.10) by restricting $\boldsymbol{p}$ to (a subset of) the two-dimensional subspace spanned by the steepest descent direction $-\boldsymbol{g}_k$ and the quasi-Newton step $\boldsymbol{p}_k^{\mathrm{QN}}$ when the model Hessian B_k is positive definite. Two-dimensional subspace minimization can also be adapted for indefinite B_k, though in that case it requires an estimate of the most negative eigenvalue of this matrix. For a comprehensive treatment of trust-region methods, see [6].

3.2. The generalized nonlinear conjugate gradient method. In the nonlinear conjugate gradient method, search directions $\overset{\mathcal{C}}{\boldsymbol{p}}_k$ are generated by the recurrence relation

$$\overset{\mathcal{C}}{\boldsymbol{p}}_{k+1} = -\overset{\mathcal{C}}{\boldsymbol{g}}_{k+1} + \beta_{k+1} \overset{\mathcal{C}}{\boldsymbol{p}}_k, \tag{3.11}$$

where $\overset{\mathcal{C}}{\boldsymbol{g}}_{k+1}$ is now defined[1] as $\frac{\partial f(\overset{\mathcal{C}}{\boldsymbol{z}}_{k+1})}{\partial \overset{\mathcal{C}}{\overline{\boldsymbol{z}}}}$ and β_{k+1} is the conjugate gradient update parameter. Different choices for the scalar β_{k+1} correspond to different conjugate gradient methods. The generalized Hestenes–Stiefel [17] update parameter $\beta_{k+1}^{\mathrm{HS}}$ can be derived from a special case of the L-BFGS search direction, where $m = 1$, $B_k^{-1} = \mathbb{I}_{2n \times 2n}$ and an exact line search, for which $\boldsymbol{g}_{k+1}^{\mathrm{H}} \boldsymbol{p}_k = 0$ for all k, is assumed. We then obtain $\overset{\mathcal{C}}{\boldsymbol{p}}_{k+1} = -B_{k+1}^{-1} \overset{\mathcal{C}}{\boldsymbol{g}}_{k+1} = -\overset{\mathcal{C}}{\boldsymbol{g}}_{k+1} + \beta_{k+1}^{\mathrm{HS}} \overset{\mathcal{C}}{\boldsymbol{p}}_k$, where

$$\beta_{k+1}^{\mathrm{HS}} = \frac{\mathrm{Re}\{(\boldsymbol{g}_{k+1} - \boldsymbol{g}_k)^{\mathrm{H}} \boldsymbol{g}_{k+1}\}}{\mathrm{Re}\{(\boldsymbol{g}_{k+1} - \boldsymbol{g}_k)^{\mathrm{H}} \boldsymbol{p}_k\}}. \tag{3.12a}$$

[1] In Section §3.1 it was argued that $\boldsymbol{g}_k = 2\frac{\partial f(\overset{\mathcal{C}}{\boldsymbol{z}}_{k+1})}{\partial \overline{\boldsymbol{z}}}$ is a more practical choice for a computer implementation. Using the latter definition throughout this section is also possible, although the generated steps would be twice as large since it is then implicitly assumed that $B_k^{-1} = 2\mathbb{I}_{2n \times 2n}$. One way to take this extra scaling factor into account is by scaling the initial line search step length appropriately. However, as it is inherent to the conjugate gradient method that the search directions it generates are often poorly scaled, the extra factor two can safely be ignored depending on the strategy chosen for the initial step length [28].

When $\boldsymbol{g}_{k+1}^{\mathrm{H}}\boldsymbol{p}_k = 0$, (3.12a) reduces to the Polak–Ribière update parameter [30]

$$\beta_{k+1}^{\mathrm{PR}} = \frac{\mathrm{Re}\{(\boldsymbol{g}_{k+1} - \boldsymbol{g}_k)^{\mathrm{H}}\boldsymbol{g}_{k+1}\}}{\boldsymbol{g}_k^{\mathrm{H}}\boldsymbol{g}_k}. \tag{3.12b}$$

Further, if f is quadratic, $\boldsymbol{g}_{k+1}^{\mathrm{H}}\boldsymbol{g}_k = 0$, and we find the Fletcher–Reeves update parameter [10]

$$\beta_{k+1}^{\mathrm{FR}} = \frac{\boldsymbol{g}_{k+1}^{\mathrm{H}}\boldsymbol{g}_{k+1}}{\boldsymbol{g}_k^{\mathrm{H}}\boldsymbol{g}_k}. \tag{3.12c}$$

Powell [33] showed that the Fletcher–Reeves method is susceptible to jamming. That is, the algorithm could take many short steps without making significant progress to the minimum. The Hestenes–Stiefel and Polak–Ribière methods, which share the common numerator $\mathrm{Re}\{(\boldsymbol{g}_{k+1} - \boldsymbol{g}_k)^{\mathrm{H}}\boldsymbol{g}_{k+1}\}$, possess a built-in restart feature that addresses the jamming problem [15]. When the step $\boldsymbol{z}_{k+1} - \boldsymbol{z}_k$ is small, the factor $\boldsymbol{g}_{k+1} - \boldsymbol{g}_k$ tends to zero. Hence, β_{k+1} becomes small and the new search direction $\boldsymbol{p}_{k+1}$ is essentially the steepest descent direction $-\boldsymbol{g}_{k+1}$.

Gilbert and Nocedal [11] proved that the modified Polak–Ribière method [34], where $\beta_{k+1}^{\mathrm{PR}+} = \max\{\beta_{k+1}^{\mathrm{PR}}, 0\}$, is globally convergent even when an inexact line search is used. Similarly, it can also be shown [15] that the modified Hestenes–Stiefel method, where $\beta_{k+1}^{\mathrm{HS}+} = \max\{\beta_{k+1}^{\mathrm{HS}}, 0\}$, is also globally convergent when using an inexact line search.

4. Nonlinear least squares problems $\min_{\boldsymbol{z}} \|F(\boldsymbol{z}, \overline{\boldsymbol{z}})\|^2$. Now we consider the special case where the objective is a nonlinear least squares problem of the form (1.2). The space spanned by $\boldsymbol{z}$ and $\overline{\boldsymbol{z}}$ is equal to $\mathcal{C}$, which is equivalent to $\mathcal{R}$ under the linear map (2.1). It is then understood that the function F can also be regarded as a function that maps $\mathcal{C}$ or $\mathcal{R}$ to $\mathbb{C}^m$, so that both the first-order model (2.7) in $\overset{\mathcal{R}}{\boldsymbol{z}}$ and the first-order model (2.8) in $\overset{\mathcal{C}}{\boldsymbol{z}}$ are applicable.

4.1. The generalized Gauss-Newton & Levenberg-Marquardt methods. The generalized Gauss–Newton and Levenberg–Marquardt methods use the first-order model

$$m_k^F(\overset{\mathcal{C}}{\boldsymbol{p}}) = F(\overset{\mathcal{C}}{\boldsymbol{z}}_k) + \frac{\partial F(\overset{\mathcal{C}}{\boldsymbol{z}}_k)}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}}\overset{\mathcal{C}}{\boldsymbol{p}} \tag{4.1}$$

to approximate F at the current iterate $\overset{\mathcal{C}}{\boldsymbol{z}}_k$. The optimization problem $\min_{\boldsymbol{z}} f(\boldsymbol{z}, \overline{\boldsymbol{z}}) \triangleq \frac{1}{2}\|F(\boldsymbol{z}, \overline{\boldsymbol{z}})\|^2$ can then be approximated by

$$m_k^f(\overset{\mathcal{C}}{\boldsymbol{p}}) = \frac{1}{2}\|m_k^F(\overset{\mathcal{C}}{\boldsymbol{p}})\|^2 + \frac{\lambda_k}{2}\|\boldsymbol{p}\|^2, \tag{4.2}$$

where λ_k is the Levenberg–Marquardt regularization parameter which influences both the length and direction of the step $\boldsymbol{p}$ that minimizes m_k^f. In the Gauss–Newton method, $\lambda_k = 0$ for all k, and a trust region framework can instead be used to control the length and direction of the step. Let $\boldsymbol{F}_k = F(\overset{\mathcal{C}}{\boldsymbol{z}}_k)$ and let $J_k = \frac{\partial F(\overset{\mathcal{C}}{\boldsymbol{z}}_k)}{\partial \overset{\mathcal{C}}{\boldsymbol{z}}^{\mathrm{T}}}$. By substituting (4.1) in (4.2), we find

$$m_k^f(\overset{\mathcal{C}}{\boldsymbol{p}}) = \frac{1}{2}\|\boldsymbol{F}_k\|^2 + \frac{1}{2}\overset{\mathcal{C}}{\boldsymbol{p}}^{\mathrm{T}}\begin{bmatrix} J_k S \\ \overline{J}_k \end{bmatrix}^{\mathrm{H}}\begin{bmatrix} \boldsymbol{F}_k \\ \overline{\boldsymbol{F}}_k \end{bmatrix} + \frac{1}{2}\overset{\mathcal{C}}{\boldsymbol{p}}^{\mathrm{H}}\left(J_k^{\mathrm{H}}J_k + \frac{\lambda_k}{2}\mathbb{I}_{2n\times 2n}\right)\overset{\mathcal{C}}{\boldsymbol{p}}. \tag{4.3}$$

Comparing (4.3) to (2.10b) reveals that the complex gradient of f is given by $\frac{1}{2}(SJ_k^{\mathrm{H}}\boldsymbol{F}_k + J_k^{\mathrm{T}}\overline{\boldsymbol{F}}_k)$ and its Hessian is approximated by $J_k^{\mathrm{H}}J_k$. The conjugate complex gradient of m_k^f is given by

$$\begin{aligned}\frac{\partial m_k^f(\overset{\mathcal{C}}{\boldsymbol{p}})}{\partial \overset{\mathcal{C}}{\boldsymbol{p}}} &= \frac{1}{2}\left(SJ_k^{\mathrm{T}}\overline{\boldsymbol{F}}_k + J_k^{\mathrm{H}}\boldsymbol{F}_k\right) + \frac{1}{2}\left(J_k^{\mathrm{H}}J_kS + SJ_k^{\mathrm{T}}\overline{J}_k + \lambda_k S\right)\overset{\mathcal{C}}{\boldsymbol{p}} \\ &= \frac{1}{2}\begin{bmatrix} J_k \\ \overline{J}_k S\end{bmatrix}^{\mathrm{H}}\begin{bmatrix}\boldsymbol{F}_k \\ \overline{\boldsymbol{F}}_k\end{bmatrix} + \frac{1}{2}\left(\begin{bmatrix} J_k \\ \overline{J}_k S\end{bmatrix}^{\mathrm{H}}\begin{bmatrix} J_k \\ \overline{J}_k S\end{bmatrix} + \lambda_k \mathbb{I}_{2n\times 2n}\right)\overset{\mathcal{C}}{\boldsymbol{p}}.\end{aligned}$$

Since f is real-valued, its conjugate complex gradient must be zero when its complex gradient is zero. The minimizer of the convex quadratic model (4.2) is then

$$\overset{\mathcal{C}}{\boldsymbol{p}}_k^* = -\begin{bmatrix} J_k \\ \overline{J}_k S \\ \sqrt{\lambda_k}\mathbb{I}_{2n\times 2n}\end{bmatrix}^{\dagger}\begin{bmatrix}\boldsymbol{F}_k \\ \overline{\boldsymbol{F}}_k \\ 0_{2n\times 1}\end{bmatrix}. \tag{4.4}$$

Let $A_k = \frac{\partial F(\overset{\mathcal{C}}{\boldsymbol{z}}_k)}{\partial \boldsymbol{z}^{\mathrm{T}}}$ and let $B_k = \frac{\partial F(\overset{\mathcal{C}}{\boldsymbol{z}}_k)}{\partial \overline{\boldsymbol{z}}^{\mathrm{T}}}$. A more efficient way to compute the minimizer (4.4) is

$$\overset{\mathcal{R}}{\boldsymbol{p}}_k^* = -\begin{bmatrix} \mathrm{Re}\{A_k\} + \mathrm{Re}\{B_k\} & \mathrm{Im}\{B_k\} - \mathrm{Im}\{A_k\} \\ \mathrm{Im}\{A_k\} + \mathrm{Im}\{B_k\} & \mathrm{Re}\{A_k\} - \mathrm{Re}\{B_k\} \\ \hline \sqrt{\lambda_k}\mathbb{I}_{2n\times 2n} & \end{bmatrix}^{\dagger}\begin{bmatrix}\mathrm{Re}\{\boldsymbol{F}_k\} \\ \mathrm{Im}\{\boldsymbol{F}_k\} \\ 0_{2n\times 1}\end{bmatrix}. \tag{4.5}$$

There are a number of cases in which additional structure in F can be exploited to simplify (4.5) further:

(i) F is analytic in $\boldsymbol{z}$, then $B_k \equiv 0_{m\times n}$ and $J_k \equiv A_k$:

$$\overset{\mathcal{R}}{\boldsymbol{p}}_k^* = -\begin{bmatrix} \mathrm{Re}\{J_k\} & -\mathrm{Im}\{J_k\} \\ \mathrm{Im}\{J_k\} & \mathrm{Re}\{J_k\} \\ \hline \sqrt{\lambda_k}\mathbb{I}_{2n\times 2n} & \end{bmatrix}^{\dagger}\begin{bmatrix}\mathrm{Re}\{\boldsymbol{F}_k\} \\ \mathrm{Im}\{\boldsymbol{F}_k\} \\ 0_{2n\times 1}\end{bmatrix} \text{ or equivalently}$$

$$\boldsymbol{p}_k^* = -\begin{bmatrix} J_k \\ \sqrt{\lambda_k}\mathbb{I}_{n\times n}\end{bmatrix}^{\dagger}\begin{bmatrix}\boldsymbol{F}_k \\ 0_{n\times 1}\end{bmatrix}. \tag{4.6}$$

(ii) $F : \mathbb{C}^n \to \mathbb{R}^m$, then $B_k \equiv \overline{A}_k$:

$$\overset{\mathcal{R}}{\boldsymbol{p}}_k^* = -\begin{bmatrix} 2\mathrm{Re}\{A_k\} & -2\mathrm{Im}\{A_k\} \\ \hline \sqrt{\lambda_k}\mathbb{I}_{2n\times 2n} & \end{bmatrix}^{\dagger}\begin{bmatrix}\boldsymbol{F}_k \\ 0_{2n\times 1}\end{bmatrix}. \tag{4.7}$$

(iii) $F : \mathbb{R}^n \to \mathbb{C}^m$, then $J_k \equiv \frac{\partial F(\boldsymbol{x}_k)}{\partial \boldsymbol{x}^{\mathrm{T}}}$:

$$\boldsymbol{p}_k^* = -\begin{bmatrix} J_k \\ \sqrt{\lambda_k}\mathbb{I}_{n\times n}\end{bmatrix}^{\dagger}\begin{bmatrix}\boldsymbol{F}_k \\ 0_{n\times 1}\end{bmatrix}. \tag{4.8}$$

The least squares system (4.5) is equivalent to the system we would have arrived at had we applied the real Gauss–Newton or Levenberg–Marquardt, which is the de facto course of action for complex nonlinear least squares problems. However, expression (4.5) offers an alternative way of computing the real Jacobian using the complex Jacobian and conjugate Jacobian, which are often easier to derive and lead to more elegant expressions than is the case for the real Jacobian, as we demonstrate in the second case study. Moreover, we also show that the real Jacobian matrix-vector product can be expressed in terms of matrix-vector products with the complex Jacobians. In this way, conjugate gradient type methods that implicitly solve the normal equations can take advantage of any structure present in the complex Jacobians.

5. Case studies.

5.1. The canonical polyadic decomposition. A tensor may be considered as an element of the tensor product of a set of vector spaces, although for this discussion their representation as a multiway array, given a choice of bases, suffices. Before we introduce the canonical polyadic decomposition, we review a few necessary definitions.

DEFINITION 5.1. *The* inner product $\langle \mathcal{T}, \mathcal{U} \rangle$ *of two tensors* $\mathcal{T}, \mathcal{U} \in \mathbb{C}^{I_1 \times \cdots \times I_N}$ *is defined as*

$$\langle \mathcal{T}, \mathcal{U} \rangle = \sum_{i_1=1}^{I_1} \cdots \sum_{i_N=1}^{I_N} \bar{t}_{i_1 \cdots i_N} u_{i_1 \cdots i_N}. \tag{5.1}$$

DEFINITION 5.2. *The outer product* $\mathcal{T} \circ \mathcal{U}$ *of a tensor* $\mathcal{T} \in \mathbb{C}^{I_1 \times \cdots \times I_P}$ *and a tensor* $\mathcal{U} \in \mathbb{C}^{J_1 \times \cdots \times J_Q}$ *is the tensor defined by*

$$(\mathcal{T} \circ \mathcal{U})_{i_1 \cdots i_P j_1 \cdots j_Q} = t_{i_1 \cdots i_P} u_{j_1 \cdots j_Q}. \tag{5.2}$$

DEFINITION 5.3. *An* N*th order* $\mathcal{T}$ *is* rank-one *if it is equal to the* outer product *of* N *nonzero vectors* $\boldsymbol{a}^{(n)} \in \mathbb{C}^{I_n}$*, i.e.,* $\mathcal{T} = \boldsymbol{a}^{(1)} \circ \cdots \circ \boldsymbol{a}^{(N)}$.

The *canonical polyadic decomposition* (CPD) [5, 16] approximates an Nth order tensor by a sum of R rank-one tensors, i.e.

$$\mathcal{T} \approx \sum_{r=1}^{R} \boldsymbol{a}_r^{(1)} \circ \cdots \circ \boldsymbol{a}_r^{(N)}. \tag{5.3}$$

The rank of a tensor, denoted by $\operatorname{rank}(\mathcal{T})$, is defined as the minimal number of rank-one terms such that (5.3) is exact. Such a decomposition is called a rank decomposition. For any CPD, the order of the different rank-one terms may be arbitrarily permuted, and the components of the rank-one terms may be arbitrarily scaled as long as their outer product remains the same. These properties are referred to as the permutation and scaling indeterminacy, respectively. When a rank decomposition is subject only to these trivial indeterminacies, it is called essentially unique. An interesting property of higher-order tensors is that their rank decompositions are often essentially unique, which is not the case for matrices. Next, we introduce a few operators which will allow us to formulate the CPD as an optimization problem.

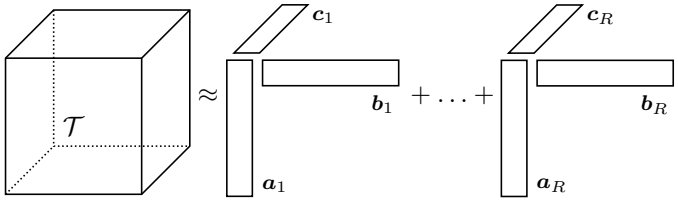


FIG. 5.1. *CPD of a third-order tensor.*

DEFINITION 5.4. *In a* mode-n matricization, flattening *or* unfolding $T_{(n)}$ *of an* N*th order tensor* $\mathcal{T} \in \mathbb{C}^{I_1 \times \cdots \times I_N}$*, tensor element with indices* $(i_1, \ldots, i_N)$ *is mapped*

to matrix element (i_n, j) *such that*

$$j = 1 + \sum_{\substack{k=1 \\ k \neq n}}^{N} (i_k - 1) J_k \text{ with } J_k = \begin{cases} 1 & \text{for } \{k = 1, k = 2 \wedge n = 1\} \\ \prod_{\substack{m=1 \\ m \neq n}}^{k-1} I_m & \text{otherwise} \end{cases}. \quad (5.4)$$

DEFINITION 5.5. *The* Kronecker product *of two matrices* $A \in \mathbb{C}^{I \times J}$ *and* $B \in \mathbb{C}^{K \times L}$ *is defined as*

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1J}B \\ \vdots & \ddots & \vdots \\ a_{I1}B & \cdots & a_{IJ}B \end{bmatrix}. \quad (5.5)$$

DEFINITION 5.6. *The* Khatri-Rao product [20] *of two matrices* $A \in \mathbb{C}^{I \times K}$ *and* $B \in \mathbb{C}^{J \times K}$ *is defined as*

$$A \odot B = \begin{bmatrix} \boldsymbol{a}_1 \otimes \boldsymbol{b}_1 & \cdots & \boldsymbol{a}_K \otimes \boldsymbol{b}_K \end{bmatrix}. \quad (5.6)$$

DEFINITION 5.7. *The* Hadamard product *of two matrices* $A \in \mathbb{C}^{I \times J}$ *and* $B \in \mathbb{C}^{I \times J}$ *is defined as*

$$A * B = \begin{bmatrix} a_{11}b_{11} & \cdots & a_{1J}b_{1J} \\ \vdots & \ddots & \vdots \\ a_{I1}b_{I1} & \cdots & a_{IJ}b_{IJ} \end{bmatrix}.$$

Let $A^{(n)} = \begin{bmatrix} \boldsymbol{a}_1^{(n)} & \cdots & \boldsymbol{a}_R^{(n)} \end{bmatrix}$ be the factor matrix corresponding to the nth mode. The rank-R CPD of an Nth order tensor $\mathcal{T}$ can be written in terms of its mode-n unfoldings as

$$T_{(n)} \approx A^{(n)}(A^{(N)} \odot \cdots \odot A^{(n+1)} \odot A^{(n-1)} \odot \cdots \odot A^{(1)})^{\mathrm{T}}, \; n = 1, \ldots, N. \quad (5.7)$$

The CPD can now be cast as the nonlinear least squares problem

$$\min_{A^{(1)}, \cdots, A^{(N)}} f_{\mathrm{CPD}} \triangleq \frac{1}{2} \| F_{\mathrm{CPD}} \|_F^2, \quad (5.8)$$

where $F_{\mathrm{CPD}} = T_{(n)} - A^{(n)}(A^{(N)} \odot \cdots \odot A^{(n+1)} \odot A^{(n-1)} \odot \cdots \odot A^{(1)})^{\mathrm{T}}$. There are as many residuals as there are elements in $\mathcal{T}$ and these residuals can be arranged in N different ways, depending on which mode-n unfolding is chosen. First we look for an expression for the complex gradient of f_{CPD}. Let $V^{\{n\}} = A^{(N)} \odot \cdots \odot A^{(n+1)} \odot A^{(n-1)} \odot \cdots \odot A^{(1)}$. The objective function f_{CPD} can be written as the sum $\frac{1}{2}(f_{\mathrm{CPD}}^{(1)} - f_{\mathrm{CPD}}^{(2)} - \overline{f}_{\mathrm{CPD}}^{(2)} + f_{\mathrm{CPD}}^{(3)})$, where

$$f_{\mathrm{CPD}}^{(1)} = \| A^{(n)} {V^{\{n\}}}^{\mathrm{T}} \|_F^2, \quad (5.9a)$$

$$f_{\mathrm{CPD}}^{(2)} = \langle T_{(n)}, A^{(n)} {V^{\{n\}}}^{\mathrm{T}} \rangle \text{ and} \quad (5.9b)$$

$$f_{\mathrm{CPD}}^{(3)} = \| T_{(n)} \|_F^2. \quad (5.9c)$$

The partial derivative of $f_{\text{CPD}}^{(1)}$ with respect to the factor matrix $A^{(n)}$ is given by

$$\begin{aligned}\frac{\partial f_{\text{CPD}}^{(1)}}{\partial A^{(n)}} &= \frac{\partial}{\partial A^{(n)}} \|A^{(n)} V^{\{n\}\mathrm{T}}\|_F^2 \\ &= \frac{\partial}{\partial A^{(n)}} \sum_i \left(A^{(n)} \boldsymbol{v}_{i:}^{\{n\}\mathrm{T}}\right)^{\mathrm{H}} \left(A^{(n)} \boldsymbol{v}_{i:}^{\{n\}\mathrm{T}}\right) \\ &= \sum_i \overline{A}^{(n)} \boldsymbol{v}_{i:}^{\{n\}\mathrm{H}} \boldsymbol{v}_{i:}^{\{n\}} \\ &= \overline{A}^{(n)} V^{\{n\}\mathrm{H}} V^{\{n\}}.\end{aligned}$$

Similarly, it can be shown that $\frac{\partial f_{\text{CPD}}^{(2)}}{\partial A^{(n)}} = \overline{T}_{(n)} V^{\{n\}}$, $\frac{\partial \overline{f}_{\text{CPD}}^{(2)}}{\partial A^{(n)}} = 0$ and $\frac{\partial f_{\text{CPD}}^{(3)}}{\partial A^{(n)}} = 0$. Let

$$W^Q = \underset{\substack{n=1 \\ n \notin S}}{\overset{N}{*}} A^{(n)\mathrm{H}} A^{(n)}$$

be the Hadamard product of all matrices $A^{(n)\mathrm{H}} A^{(n)}$, $n \in \{1, \ldots, N\} \backslash Q$. Then a corollary of Proposition 5.8 is that $V^{\{n\}\mathrm{H}} V^{\{n\}}$ can be evaluated more efficiently as $W^{\{n\}}$ without explicitly forming any Khatri-Rao products [41].

PROPOSITION 5.8. *Let $A^{(n)} \in \mathbb{C}^{I_n \times J}$, $n = 1, \ldots, N$, let $V = A^{(p_1)} \odot \cdots \odot A^{(p_N)}$ where $\boldsymbol{p}$ is any permutation of $\{1, \ldots, N\}$ and let $W = \overset{N}{\underset{n=1}{*}} A^{(n)\mathrm{H}} A^{(n)}$ be the Hadamard product of all matrices $A^{(n)\mathrm{H}} A^{(n)}$. Then $V^{\mathrm{H}} V = W$.*

Proof.

$$\begin{aligned}(V^{\mathrm{H}} V)_{nm} &= (\boldsymbol{a}_n^{(p_1)} \otimes \cdots \otimes \boldsymbol{a}_n^{(p_N)})^H (\boldsymbol{a}_m^{(p_1)} \otimes \cdots \otimes \boldsymbol{a}_m^{(p_N)}) \\ &= \langle \boldsymbol{a}_n^{(1)} \circ \cdots \circ \boldsymbol{a}_n^{(N)}, \boldsymbol{a}_m^{(1)} \circ \cdots \circ \boldsymbol{a}_m^{(N)} \rangle \\ &= (\boldsymbol{a}_n^{(1)H} \boldsymbol{a}_m^{(1)}) \cdots (\boldsymbol{a}_n^{(N)H} \boldsymbol{a}_m^{(N)}) = w_{nm}.\end{aligned}$$

□

Let $\boldsymbol{z} = \left(\text{vec}(A^{(1)}), \ldots, \text{vec}(A^{(N)})\right) \in \mathbb{C}^{R \sum_n^N I_n}$, where $\text{vec}(\cdot)$ denotes the column-wise vectorization of a matrix. The cogradient of f_{CPD} completely determines the complex gradient of f_{CPD} and is given by

$$\frac{\partial f_{\text{CPD}}}{\partial \boldsymbol{z}} = \frac{1}{2} \begin{bmatrix} \text{vec}\left(\overline{A}^{(1)} W^{\{1\}} - \overline{T}_{(1)} V^{\{1\}}\right) \\ \vdots \\ \text{vec}\left(\overline{A}^{(N)} W^{\{N\}} - \overline{T}_{(N)} V^{\{N\}}\right) \end{bmatrix}. \tag{5.10}$$

As outlined in Section §3, it suffices to implement the function f_{CPD} and its associated scaled conjugate cogradient $g_{\text{CPD}} \triangleq 2\overline{\frac{\partial f_{\text{CPD}}}{\partial \boldsymbol{z}}}$ to be able to apply any of the generalized gradient based optimization methods presented in that section. Although we did not explicitly derive an expression for the real gradient, it is not hard to imagine that this derivation and resulting expression would be relatively more complicated compared to that of the complex gradient.

To apply either the generalized Gauss–Newton or Levenberg–Marquardt method, we define J as the Jacobian $\frac{\partial \text{vec}(F_{\text{CPD}})}{\partial \boldsymbol{z}^{\mathrm{T}}}$ of the nonlinear least squares problem (5.8).

Since the conjugate Jacobian is identically equal to zero, F_{CPD} is analytic in $\boldsymbol{z}$ and we may use (4.6) to compute the Levenberg–Marquardt step $\boldsymbol{p}_k^{\text{LM}}$. However, the number of rows in J is equal to $\prod_n^N I_n$ and is potentially very large compared to the number of columns $R\sum_n^N I_n$. Solving the normal equations $J_k^{\text{H}} J_k \boldsymbol{p}_k = -J_k^{\text{H}} \text{vec}(F_{\text{CPD}}(\boldsymbol{z}_k)) = -g_{\text{CPD}}(\boldsymbol{z}_k)$ is one way to reduce the computational complexity, although at the cost of squaring the condition number of the system. It can be shown that $J^{\text{H}}J$ is an $N \times N$ block matrix, in which the (n,m)-th block is the $R \times R$ block matrix

$$\left(J^{\text{H}}J\right)_{nm} = \begin{cases} W^{\{n\}} \otimes \mathbb{I}_{I_n \times I_n} & \text{for } n = m \\ \begin{bmatrix} w_{1,1}^{\{n,m\}} \boldsymbol{a}_1^{(n)} \boldsymbol{a}_1^{(m)\text{H}} & \cdots & w_{1,R}^{\{n,m\}} \boldsymbol{a}_R^{(n)} \boldsymbol{a}_1^{(m)\text{H}} \\ \vdots & \ddots & \vdots \\ w_{R,1}^{\{n,m\}} \boldsymbol{a}_1^{(n)} \boldsymbol{a}_R^{(m)\text{H}} & \cdots & w_{R,R}^{\{n,m\}} \boldsymbol{a}_R^{(n)} \boldsymbol{a}_R^{(m)\text{H}} \end{bmatrix} & \text{for } n \neq m \end{cases}. \quad (5.11)$$

Inverting $J^{\text{H}}J$ costs $O((R\sum_{n=1}^N I_n)^3)$ flop unless its structure is taken into account. A straightforward way to do so is by solving the normal equations iteratively with a preconditioned conjugate gradient method. Let $Y \in \mathbb{C}^{I_m \times R}$ and let $\boldsymbol{y} = \text{vec}(Y)$, then it can be shown that

$$\left(J^{\text{H}}J\right)_{nm} \boldsymbol{y} = \begin{cases} \text{vec}\left(Y\overline{W}^{\{n\}}\right) & \text{for } n = m \\ \text{vec}\left(A^{(n)}\left(\overline{W}^{\{n,m\}} * \left(Y^{\text{T}}\overline{A}^{(m)}\right)\right)\right) & \text{for } n \neq m \end{cases}. \quad (5.12)$$

The matrix-vector product $J^{\text{H}}J\boldsymbol{z}$ costs only $O(NR^2\sum_{n=1}^N I_n)$ flop using (5.12), which means that inverting $J^H J$ can be done in at most $O(NR^3(\sum_{n=1}^N I_n)^2)$ flop. When $I_n \equiv I$, this is a reduction from $O(N^3R^3I^3)$ flop to $O(N^3R^3I^2)$ flop. In practice an approximate solution can often be obtained in fewer than $R\sum_{n=1}^N I_n$ iterations with the help of the block-diagonal preconditioner P, wherein the nth diagonal block is defined as $W^{\{n\}} \otimes \mathbb{I}_{I_n \times I_n}$. From (5.12) we see that inverting P requires solving N square Hermitian linear systems of order R, for a total cost of $O(NR^3)$ flop.

5.2. Simulation of nonlinear circuits in the frequency domain. Simulation in the frequency domain avoids many of the problems experienced when trying to use traditional time domain simulators such as SPICE [24, 35] to find the steady-state behavior of analog, radio frequency and microwave circuits. In principle, the periodic response can always be found by integrating the differential equations that describe the system until the transient responses vanish. However, this approach can be prohibitively expensive whenever the transients are governed by large time constants.

Applying Kirchoff's current law in the frequency domain to a nonlinear circuit of $N+1$ nodes yields N parametrized nonlinear equations

$$I(\omega, V(\omega), \overline{V(\omega)}) = 0_{N\times 1}, \quad (5.13)$$

where $I(\omega, V(\omega), \overline{V(\omega)})$ describes the total current through each node as a function of the angular frequency ω and the node voltages. One of the nodes is selected as a reference node. The node voltages are referenced with respect to the reference node. $V(\omega) \in \mathbb{C}^N$. Voltage sources can be brought into account using a Norton equivalent circuit. If the circuit were linear, (5.13) could be written as the parametrized linear system $Y(\omega)V(\omega) + I_s(\omega) = 0$, where $Y(\omega) \in \mathbb{C}^{N\times N}$ is a node admittance matrix and

$I_s(\omega) \in \mathbb{C}^N$ represents the current sources from and to each node. In the harmonic balance method [25], the node voltages and currents are approximated by a truncated Fourier series of H harmonics. The time domain voltage $v_n(t)$ of the nth node is represented by the complex coefficients $V_h^{(n)}$, $h = 0, \ldots, H-1$, in the frequency domain, i.e.

$$v_n(t) \approx V_0^{(n)} + \sum_{h=1}^{H-1} \left(V_h^{(n)} e^{j2\pi f_0 ht} + \overline{V}_h^{(n)} e^{-j2\pi f_0 ht} \right), \text{ where} \tag{5.14a}$$

$$V_h^{(n)} = f_0 \int_0^{\frac{1}{f_0}} v_n(t) e^{-j2\pi f_0 ht} \mathrm{d}t. \tag{5.14b}$$

Analogously, the time domain current $i_n(t, v(t))$ through the nth node is a function of all node voltages $v(t) \triangleq \{v_1(t), \ldots, v_N(t)\}$ and is represented by the complex coefficients $I_h^{(n)}$, $h = 0, \ldots, H-1$, in the frequency domain.

Let $\boldsymbol{V}^{(n)} = (V_0^{(n)}, \ldots, V_{H-1}^{(n)})$ and let $\boldsymbol{I}^{(n)} = (I_0^{(n)}, \ldots, I_{H-1}^{(n)})$. The steady-state behavior of a nonlinear circuit can then be approximated as the solution of the nonlinear least squares problem

$$\min_{\boldsymbol{V}^{(1)}, \cdots, \boldsymbol{V}^{(N)}} f_{\mathrm{HBM}} \triangleq \frac{1}{2} \|F_{\mathrm{HBM}}\|^2, \tag{5.15}$$

where $F_{\mathrm{HBM}} = (\boldsymbol{I}^{(1)}, \ldots, \boldsymbol{I}^{(N)})$. At this point the optimization problem (5.15) is usually converted to the real domain so that real optimization methods can be applied. However, the derivatives of F_{HBM} with respect to the real and imaginary parts of $V_h^{(n)}$ obscure the structure in the real Jacobian. In comparison, the expressions for the complex (conjugate) Jacobian required by the generalized Gauss–Newton and Levenberg–Marquardt methods are much more elegant. As we will see, their structure is easy to recognize and in this case also easy to exploit. To apply one of the latter methods, we need expressions for the Jacobian and conjugate Jacobian of F_{HBM}. The derivative of the hth harmonic of the current through the nth node $I_h^{(n)}$ with respect to the kth harmonic of the mth node voltage $V_k^{(m)}$ is

$$\begin{aligned} \frac{\partial I_h^{(n)}}{\partial V_k^{(m)}} &= f_0 \int_0^{\frac{1}{f_0}} \frac{\partial i_n(t, v(t))}{\partial v_m(t)} \frac{\partial v_m(t)}{\partial V_k^{(m)}} e^{-j2\pi f_0 ht} \mathrm{d}t \\ &= f_0 \int_0^{\frac{1}{f_0}} \frac{\partial i_n(t, v(t))}{\partial v_m(t)} e^{-j2\pi f_0 (h-k)t} \mathrm{d}t \\ &\triangleq G_{h-k}^{(n,m)}, \end{aligned} \tag{5.16}$$

which is just the $(h-k)$-th Fourier series coefficient of the derivative $\frac{\partial i_n(t,v(t))}{\partial v_m(t)}$. Similarly, it can be shown that $\frac{\partial I_h^{(n)}}{\partial \overline{V}_k^{(m)}} = G_{h+k}^{(n,m)}$. Further, because $\frac{\partial i_n(t,v(t))}{\partial v_m(t)}$ is real, $G_h^{(n,m)} = \overline{G}_h^{(n,m)}$ for any integer h. Let $\boldsymbol{z} = (\boldsymbol{V}^{(1)}, \ldots, \boldsymbol{V}^{(N)})$, then we have that the

(n,m)-th block of the Jacobian $\frac{\partial F_{\text{HBM}}}{\boldsymbol{z}^{\text{T}}}$ is the Hermitian Toeplitz matrix

$$\left(\frac{\partial F_{\text{HBM}}}{\boldsymbol{z}^{\text{T}}}\right)_{nm} = \frac{\partial \boldsymbol{I}^{(n)}}{\partial \boldsymbol{V}^{(m)\,\text{T}}} = \begin{bmatrix} G_0^{(n,m)} & \overline{G}_1^{(n,m)} & \overline{G}_2^{(n,m)} & \cdots & \overline{G}_{H-1}^{(n,m)} \\ G_1^{(n,m)} & G_0^{(n,m)} & \overline{G}_1^{(n,m)} & \cdots & \overline{G}_{H-2}^{(n,m)} \\ G_2^{(n,m)} & G_1^{(n,m)} & G_0^{(n,m)} & \cdots & \overline{G}_{H-3}^{(n,m)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ G_{H-1}^{(n,m)} & G_{H-2}^{(n,m)} & G_{H-3}^{(n,m)} & \cdots & G_0^{(n,m)} \end{bmatrix}, \tag{5.17}$$

and the (n,m)-th block of the conjugate Jacobian $\frac{\partial F_{\text{HBM}}}{\overline{\boldsymbol{z}}^{\text{T}}}$ is the Hankel matrix

$$\left(\frac{\partial F_{\text{HBM}}}{\overline{\boldsymbol{z}}^{\text{T}}}\right)_{nm} = \frac{\partial \boldsymbol{I}^{(n)}}{\partial \overline{\boldsymbol{V}}^{(m)\,\text{T}}} = \begin{bmatrix} G_0^{(n,m)} & G_1^{(n,m)} & G_2^{(n,m)} & \cdots & G_{H-1}^{(n,m)} \\ G_1^{(n,m)} & G_2^{(n,m)} & G_3^{(n,m)} & \cdots & G_H^{(n,m)} \\ G_2^{(n,m)} & G_3^{(n,m)} & G_4^{(n,m)} & \cdots & G_{H+1}^{(n,m)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ G_{H-1}^{(n,m)} & G_H^{(n,m)} & G_{H+1}^{(n,m)} & \cdots & G_{2H-2}^{(n,m)} \end{bmatrix}. \tag{5.18}$$

For each such block, the coefficients $G_h^{(n,m)}$, $h = 0, \ldots, 2H-2$, can be computed in $O(H \log H)$ flop using the fast Fourier transform. The cost of computing the Jacobians is then $O(N^2 H \log H)$ flop, and can be stored compactly using $O(N^2 H)$ complex numbers. It is well known that the product of a Toeplitz or Hankel matrix of order H with a vector can be done in $O(H \log H)$ operations [1, 19]. We can exploit this fact when solving the Gauss–Newton or Levenberg–Marquardt step (4.5) iteratively. Let A_k and B_k be defined as in (4.5) and let

$$C_k = \left[\begin{array}{cc} \operatorname{Re}\{A_k\} + \operatorname{Re}\{B_k\} & \operatorname{Im}\{B_k\} - \operatorname{Im}\{A_k\} \\ \operatorname{Im}\{A_k\} + \operatorname{Im}\{B_k\} & \operatorname{Re}\{A_k\} - \operatorname{Re}\{B_k\} \\ \hline \multicolumn{2}{c}{\sqrt{\lambda_k}\mathbb{I}_{2n\times 2n}} \end{array}\right]$$

be the real Jacobian, then it is not hard to show that

$$C_k \overset{\mathbb{R}}{\boldsymbol{s}} = \begin{bmatrix} \operatorname{Re}\{A_k \boldsymbol{s}\} + \operatorname{Re}\{B_k \overline{\boldsymbol{s}}\} \\ \operatorname{Im}\{A_k \boldsymbol{s}\} + \operatorname{Im}\{B_k \overline{\boldsymbol{s}}\} \\ \sqrt{\lambda_k}\operatorname{Re}\{\boldsymbol{s}\} \\ \sqrt{\lambda_k}\operatorname{Im}\{\boldsymbol{s}\} \end{bmatrix} \text{ and} \tag{5.19a}$$

$$C_k^{\text{T}} \begin{bmatrix} \overset{\mathbb{R}}{\boldsymbol{t}} \\ \overset{\mathbb{R}}{\boldsymbol{u}} \end{bmatrix} = \begin{bmatrix} \operatorname{Re}\{A_k^{\text{H}} \boldsymbol{t}\} + \operatorname{Re}\{B_k^{\text{H}} \boldsymbol{t}\} + \sqrt{\lambda_k}\operatorname{Re}\{\boldsymbol{u}\} \\ \operatorname{Im}\{A_k^{\text{H}} \boldsymbol{t}\} - \operatorname{Im}\{B_k^{\text{H}} \boldsymbol{t}\} + \sqrt{\lambda_k}\operatorname{Im}\{\boldsymbol{u}\} \end{bmatrix}. \tag{5.19b}$$

The real matrix-vector products (5.19) depend only on the complex block-Toeplitz and block-Hankel matrix-vector products $A_k \boldsymbol{s}$, $B_k \overline{\boldsymbol{s}}$, $A_k^{\text{H}} \boldsymbol{t}$ and $B_k^{\text{H}} \boldsymbol{t}$, each of which can be computed in $O(N^2 H \log H)$ flop. Iterative methods that implicitly solve the normal equations, such as CGLS or LSQR [17, 29], can then be applied to the real system (4.5) using the transformations (5.19). The total cost of computing the Gauss–Newton or Levenberg–Marquardt step is then at most $O(N^3 H^2 \log H)$ flop, with a memory cost of $O(N^2 H)$ complex numbers. If the Levenberg–Marquardt algorithm were applied in the real domain, the derivation of the first-order derivatives would not only be more involved, it would also be harder to notice and exploit their underlying structure. In a naive implementation, computing the step would then cost $O(N^3 H^3)$ flop and $O(N^2 H^2)$ memory. In comparison, the the generalized nonlinear least squares method allows relatively many more harmonics to be used to represent the steady-state solution.

6. Conclusion. In this paper we showed that functions of complex variables have complex Taylor series expansions. The complex derivatives underlying these expansions avoid the structural overhead of dealing with derivatives of the real and imaginary parts separately. We used this approach to generalize the L-BFGS and nonlinear conjugate gradient methods to functions of complex variables. We showed that these methods need only depend on the objective function and its scaled conjugate cogradient. The latter conveniently coincides with the real gradient when optimizing over the real domain. The computational and storage cost of these generalized methods are equal to those of their real counterparts applied on the optimization problem formulated in the real domain. We also generalized Gauss–Newton and Levenberg–Marquardt methods for solving nonlinear least squares problems. We showed that the real Jacobian arising in the equivalent nonlinear least squares problem formulated in the real domain, is in fact composed of a combination of two complex Jacobians. In the case studies we demonstrated that the complex gradients and complex Jacobians are not only often easier to derive than their real counterparts, but are also described by more compact expressions. Additionally, we showed that the real Jacobian matrix-vector product can be expressed in terms of complex Jacobian matrix-vector products. This allows the structure of the complex Jacobians to be exploited when applying iterative methods for solving the normal equations associated with the Gauss–Newton or Levenberg–Marquardt step. The algorithms discussed in this article, including several line search and trust region variants, have been implemented and tested in MATLAB and are available upon request.

Acknowledgments. Laurent Sorber is supported by a Flanders Institute of Science and Technology (IWT) doctoral scholarship. Marc Van Barel's research is supported by (1) the Research Council K.U.Leuven: (a) project OT/10/038 (Multi-parameter model order reduction and its applications), (b) CoE EF/05/006 Optimization in Engineering (OPTEC), and by (2) the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office, Belgian Network DYSCO (Dynamical Systems, Control, and Optimization). Lieven De Lathauwer's research is supported by (1) the Research Council K.U.Leuven: (a) GOA-MaNet, (b) CoE EF/05/006 Optimization in Engineering (OPTEC), (c) CIF1, (d) STRT1/08/023, (2) F.W.O.: (a) project G.0427.10N, (b) Research Communities ICCoS, ANMMM and MLDM, (3) the Belgian Federal Science Policy Office: IUAP P6/04 (DYSCO, "Dynamical systems, control and optimization", 20072011), and by (4) EU: ERNSI. The scientific responsibility rests with its authors.

REFERENCES

[1] Z. Bai, D. Day, J. W. Demmel, J. J. Dongarra, M. Gu, A. Ruhe, and H. A. van der Vorst, *Templates for linear algebra problems*, in Computer Science Today, J. van Leeuwen, ed., vol. 1000 of Lecture Notes in Computer Science, Springer Berlin/Heidelberg, 1995, pp. 115–140.

[2] D. H. Brandwood, *A complex gradient operator and its application in adaptive array theory*, IEEE Proc. H, 130 (2008), pp. 11–16.

[3] C. G. Broyden, *The convergence of a class of double-rank minimization algorithms: 2. The new algorithm*, J. Inst. Math. Appl., 6 (1970), pp. 222–231.

[4] R. H. Byrd, R. B. Schnabel, and G. A. Schultz, *Approximate solution of the trust region problem by minimization over two-dimensional subspaces*, Math. Program., 40 (1988), pp. 247–263.

[5] J. D. Carroll and J.-J. Chang, *Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition*, Psychometrika, 35 (1970), pp. 283–319.

[6] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Trust-region methods*, MPS-SIAM Series on Optimization, SIAM and MPS, 2000.
[7] J. E. Dennis, Jr. and H. H. W. Mei, *Two new unconstrained optimization algorithms which use function and gradient values*, J. Opt. Th. Appl., 28 (1979), pp. 453–482.
[8] J. E. Dennis, Jr. and J. J. Moré, *Quasi-newton methods, motivation and theory*, SIAM Rev., 19 (1977), pp. 46–89.
[9] R. Fletcher, *A new approach to variable metric algorithms*, Comp. J., 13 (1970), pp. 317–322.
[10] R. Fletcher and C. M. Reeves, *Function minimization by conjugate gradients*, Comp. J., 7 (1964), pp. 149–154.
[11] J. C. Gilbert and S. Abbasbandy, *Global convergence properties of conjugate gradient methods for optimization*, SIAM J. Optim., 2 (1992), pp. 21–42.
[12] J. C. Gilbert and C. Lemaréchal, *Some numerical experiments with variable-storage quasi-newton algorithms*, Math. Program., 45 (1989), pp. 407–435.
[13] D. Goldfarb, *A family of variable-metric methods derived by variational means*, Math. Comp., 24 (1970), pp. 23–26.
[14] W. W. Hager and H. Zhang, *A new conjugate gradient method with guaranteed descent and an efficient line search*, SIAM J. Optim., 16 (2006), pp. 170–192.
[15] ———, *A survey of nonlinear conjugate gradient methods*, Pac. J. Optim., 2 (2006), pp. 35–58.
[16] R. A. Harshman, *Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis*, UCLA Working Papers in Phonetics, 16 (1970), pp. 84–84.
[17] M. R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Stand., 49 (1952), pp. 409–436.
[18] R. C. James, *Advanced calculus*, Wadsworth, 1966.
[19] T. Kailath and A. H. Sayed, eds., *Fast reliable algorithms for matrices with structure*, SIAM, Philadelphia, PA, USA, May 1999.
[20] C. G. Khatri and C. R. Rao, *Solutions to some functional equations and their applications to characterization of probability distributions*, Sankhya: Indian J. of Stat., 30 (1968), pp. 167–180.
[21] K. Kreutz-Delgado, *The complex gradient operator and the CR-calculus*, ArXiv e-prints, (2009).
[22] D. C. Liu and J. Nocedal, *On the limited memory BFGS method for large scale optimization*, Math. Program., 45 (1989), pp. 503–528.
[23] J. J. Moré and D. J. Thuente, *Line search algorithms with guaranteed sufficient decrease*, ACM Trans. Math. Softw., 20 (1994), pp. 286–307.
[24] L. W. Nagel, *SPICE2: A computer program to simulate semiconductor circuits*, memorandum ERL-M520, electronics research labora, University of California, Berkeley, 1975.
[25] M. S. Nakhla and J. Vlach, *A piecewise harmonic balance technique for determination of periodic response of nonlinear systems*, IEEE Trans. Circuits and Systems, 23 (1976), pp. 85–91.
[26] Z. Nehari, *Introduction to complex analysis*, Allyn & Bacon, Inc, 1961.
[27] J. Nocedal, *Updating quasi-Newton matrices with limited storage*, Math. Comp., 35 (1980), pp. 773–782.
[28] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer Series in Operations Research, Springer, second ed., 2006.
[29] C. C. Paige and M. A. Saunders, *LSQR: An algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Software, 8 (1982), pp. 43–71.
[30] E. Polak and G. Ribière, *Note sur la convergence de methods de directions conjugres*, Revue Française Informat. Recherche Operationnelle, 16 (1969), pp. 35–43.
[31] M. J. D. Powell, *A hybrid method for nonlinear equations*, in Numerical methods for nonlinear algebraic equations, P. Robinowitz, ed., Gordon and Breach Science, London, 1970, pp. 87–144.
[32] ———, *A new algorithm for unconstrained optimization*, in Nonlinear programming, J. B. Rosen, O. L. Mangasarian, and K. Ritter, eds., Academic Press, New York, 1970, pp. 31–66.
[33] ———, *Restart procedures for the conjugate gradient method*, Math. Program., 12 (1977), pp. 241–254.
[34] ———, *Nonconvex minimization calculations and the conjugate gradient method*, Numerical Analysis, Springer-Verlag, Berlin, 1984.
[35] T. L. Quarles, *Analysis of performance and convergence issues for circuit simulation*, memorandum ERL-M89/42, electronics research labo, University of California, Berkeley, 1989.
[36] R. Remmert, *Theory of Complex Functions*, Springer-Verlag, 1991.

[37] R. B. Schnabel and J. E. Dennis, Jr., *A new derivation of symmetric positive definite secant updates*, tech. report, Department of Computer Science, Colorado University, Aug. 1980.
[38] D. F. Shanno, *Conditioning of quasi-Newton methods for function minimization*, Math. Comp., 24 (1970), pp. 647–656.
[39] J. Sherman and W. J. Morrison, *Adjustment of an inverse matrix corresponding to changes in the elements of a given column or a given row of the original matrix*, Ann. of Math. Stat., 20 (1949), pp. 621–621.
[40] ———, *Adjustment of an inverse matrix corresponding to a change in one element of a given matrix*, Ann. of Math. Stat., 21 (1950), pp. 124–127.
[41] G. Tomasi, *Practical and computational aspects in chemometric data analysis*, PhD thesis, The Royal Veterinary and Agricultural University, May 2006.
[42] P. Wolfe, *Convergence conditions for ascent methods*, SIAM Rev., 11 (1969), pp. 226–235.