Numerical Differentiation

using the Complex-Step and Backward Difference Approximations

Tamas Kis | tamas.a.kis@outlook.com

Tamas Kis
https://tamaskis.github.io

Contents

List of Algorithms

PART I

Complex-Step Approximation

1

The Complex-Step Approximation

1.1 Definition

Consider a scalar-valued function $f : \mathbb{R} \to \mathbb{R}$. The complex-step approximation to its derivative with respect to x, evaluated at $x = x_0$, is defined as [11, 16]

$$\boxed{\left.\frac{df}{dx}\right|_{x=x_0} \approx \frac{\operatorname{Im}\left[f(x_0 + ih)\right]}{h}} \tag{1.1}$$

where h is the step size.

1.2 Implementation

As noted in Cleve Moler's blog post on the topic [13], the complex-step approximation converges to within double precision at a step size of about $h \approx \sqrt{\varepsilon}$ (due to the $\mathcal{O}(h^2)$ convergence), where ε represents double precision. Therefore, the implementation consists of just two steps: initializing $h = \sqrt{\varepsilon}$, and then approximating the derivative using Eq. (1.1). This procedure is formalized as Algorithm 1 below.

Algorithm 1: `iderivative_s`

Derivative of a univariate, scalar-valued function using the complex-step approximation.

Given:
- $f(x)$ - univariate, scalar-valued function ($f : \mathbb{R} \to \mathbb{R}$)
- $x_0 \in \mathbb{R}$ - point at which to differentiate
- $h \in \mathbb{R}$ - *(OPTIONAL)* step size

Procedure:
1. Initialize the step size if not input.

$$h = \sqrt{\varepsilon}$$

2. Evaluate the derivative of $f(x)$ at x_0 using the complex-step approximation.

$$\left.\frac{df}{dx}\right|_{x=x_0} \approx \frac{\operatorname{Im}\left[f(x_0 + ih)\right]}{h}$$

Return:

- $\left.\dfrac{df}{dx}\right|_{x=x_0} \in \mathbb{R}$ - derivative of $f(x)$ evaluated at $x = x_0$

Note: We precede all algorithm names with "`i`" to indicate that the algorithm is using the complex-step derivative approximation, where i is the imaginary unit.

1.3 Limitations

1.3.1 Higher-Order Derivatives

The complex-step approximation can be extended to second derivatives as

$$\left.\frac{df}{dx}\right|_{x=x_0} \approx \frac{2\left(f(x_0) - \operatorname{Re}\left[f(x_0 + ih)\right]\right)}{h^2}$$

Unfortunately, unlike in the first derivative case, the second derivative approximation can introduce errors if the step size is made too small. Therefore, it can be tricky to implement for general nonlinear functions and we will use a different approach when evaluating second derivatives in Section 2.6.

Additionally, we cannot use nested calls on a complex-step differentiation algorithm to obtain higher-order derivatives. Consider trying to approximate a second derivative by nesting one complex-step approximation within another:

$$\left.\frac{df}{dx}\right|_{x=x_0} \approx \left.\frac{d}{dx}\right|_{x=x_0} \left[\frac{\operatorname{Im}\left[f(x_0 + ih)\right]}{h}\right] \approx \frac{\operatorname{Im}\left[\overbrace{\frac{\operatorname{Im}\left[f(x_0 + 2ih)\right]}{h}}^{\text{this term has no imaginary part}}\right]}{h}$$

Since the term in the bracket has no imaginary part, we would simply get

$$\left.\frac{df}{dx}\right|_{x=x_0} = 0$$

which is incorrect.

1.3.2 Complexification

There are some special cases of functions where the complex-step approximation will not work directly; for example, trying to differentiate functions using MATLAB's `atan2` or `abs` would result in errors. The *Numerical Differentiation Toolbox* for MATLAB includes complexified versions of these functions (`iatan2` and `iabs`), but a more exhaustive implementation has been programmed (in Fortran) and can be found in [3]. However, for MATLAB specifically, most differentiable functions are already suitable for use with the complex-step approximation [12].

1.3.3 Functions That Yield Incorrect Results

The following functions were tested in `iderivative_test` in the *Numerical Differentiation Toolbox* for MATLAB and yield incorrect results:

- $\operatorname{arccsc}(x)$ for $x < -1$
- $\operatorname{arcsec}(x)$ for $x < -1$
- $\operatorname{arccoth}(x)$ for $0 < x < 1$
- $\operatorname{arctanh}(x)$ for $x > 1$
- $\operatorname{arcsech}(x)$ for $-1 < x < 0$
- $\operatorname{arccoth}(x)$ for $-1 < x < 0$
- $\operatorname{arccosh}(x)$ for $x < -1$
- $\operatorname{arctanh}(x)$ for $x < -1$

Other functions that may yield errors due to lack of complexification are `max` and `min`; “complexified” versions of these can be found in [3] (as discussed in Section 1.3.2), but are *not* included in the *Numerical Differentiation Toolbox* for MATLAB.

2

Multivariate Differentiation Using the Complex-Step Approximation

2.1 Derivative of a Univariate, Vector-Valued Function

Consider a univariate, vector-valued function $\mathbf{f} : \mathbb{R} \to \mathbb{R}^m$. To approximate its derivative at a point $x = x_0$, we can begin by simply differentiating each component of $\mathbf{f}$ with respect to x and evaluating the result at $x = x_0$.

$$\left.\frac{d\mathbf{f}}{dx}\right|_{x=x_0} = \begin{bmatrix} \left.\dfrac{df_1}{dx}\right|_{x=x_0} \\ \vdots \\ \left.\dfrac{df_m}{dx}\right|_{x=x_0} \end{bmatrix}$$

Applying the complex-step approximation to evaluate at $x = x_0$,

$$\left.\frac{d\mathbf{f}}{dx}\right|_{x=x_0} \approx \begin{bmatrix} \dfrac{\operatorname{Im}\left[f_1(x_0 + ih)\right]}{h} \\ \vdots \\ \dfrac{\operatorname{Im}\left[f_m(x_0 + ih)\right]}{h} \end{bmatrix}$$

Therefore, in a more compact form, we can simply write

$$\boxed{\left.\frac{d\mathbf{f}}{dx}\right|_{x=x_0} \approx \frac{\operatorname{Im}\left[\mathbf{f}(x_0 + ih)\right]}{h}} \tag{2.1}$$

For completeness, we include Algorithm 2. However, it should be noted that Algorithm 2 is nearly identical to Algorithm 1, except it is generalized to a vector-valued function $\mathbf{f}$.

Algorithm 2: `iderivative`

Derivative of a univariate, vector-valued function using the complex-step approximation.

Given:

- $\mathbf{f}(x)$ - univariate, vector-valued function ($\mathbf{f}:\mathbb{R}\to\mathbb{R}^m$)
- $x_0\in\mathbb{R}$ - point at which to differentiate
- $h\in\mathbb{R}$ - *(OPTIONAL)* step size

Procedure:

1. Initialize the step size if not input.

$$h=\sqrt{\varepsilon}$$

2. Evaluate the derivative of $\mathbf{f}(x)$ at $x=x_0$ using the complex-step approximation.

$$\left.\frac{d\mathbf{f}}{dx}\right|_{x=x_0}\approx\frac{\operatorname{Im}\left[\mathbf{f}(x_0+ih)\right]}{h}$$

Return:

- $\left.\frac{d\mathbf{f}}{dx}\right|_{x=x_0}\in\mathbb{R}^m$ - derivative of $\mathbf{f}(x)$ evaluated at x_0

Note:

- This function requires 1 evaluation of $\mathbf{f}(x)$.

2.2 Partial Derivative of a Multivariate, Scalar or Vector-Valued Function

Consider a multivariate, scalar-valued function $f:\mathbb{R}^n\to\mathbb{R}$ where $\mathbf{x}\in\mathbb{R}^n$ is the independent variable. The complex-step approximation for the partial derivative of $f(\mathbf{x})$ with respect to x_j, evaluated at $\mathbf{x}=\mathbf{x}_0$, is

$$\left.\frac{\partial f}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_0}\approx\frac{\operatorname{Im}\left[f\left((x_{0,1},x_{0,2},...,x_{0,j}+ih,...,x_{0,n})^T\right)\right]}{h}\tag{2.2}$$

If we define

$$\boxed{\boldsymbol{\chi}_j=\begin{bmatrix}0\\ \vdots\\ ih\\ \vdots\\ 0\end{bmatrix}\leftarrow j^{\text{th}}\text{ element}}\tag{2.3}$$

then we can write the partial derivative in a more compact form as

$$\boxed{\left.\frac{\partial f}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_0}\approx\frac{\operatorname{Im}\left[f(\mathbf{x}_0+\boldsymbol{\chi}_j)\right]}{h}}\tag{2.4}$$

From Section 2.1, we know it is trivial to extend this to the vector-valued case ($\mathbf{f}:\mathbb{R}^n\to\mathbb{R}^m$):

$$\boxed{\left.\frac{\partial \mathbf{f}}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_0}\approx\frac{\operatorname{Im}\left[\mathbf{f}(\mathbf{x}_0+\boldsymbol{\chi}_j)\right]}{h}}\tag{2.5}$$

In its implementation, we do not use $\boldsymbol{\chi}_j$ (although this term will be useful in later sections). Instead, within the function, we redefine $\mathbf{x}_0$ as

$$\mathbf{x}_0 \leftarrow \mathbf{x}_0 + \boldsymbol{\chi}_j$$

and then simply perform

$$\left.\frac{\partial \mathbf{f}}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_0} \approx \frac{\operatorname{Im}\left[\mathbf{f}(\mathbf{x}_0)\right]}{h}$$

Algorithm 3: `ipartial`

Partial derivative of a multivariate, scalar or vector-valued function using the complex-step approximation.

Given:

- $\mathbf{f}(\mathbf{x})$ - multivariate, vector-valued function ($\mathbf{f}:\mathbb{R}^n \to \mathbb{R}^m$)
- $\mathbf{x}_0 \in \mathbb{R}^n$ - point at which to differentiate
- $j \in \mathbb{Z}$ - index of element of $\mathbf{x}$ to differentiate with respect to
- $h \in \mathbb{R}$ - *(OPTIONAL)* step size

Procedure:

1. Initialize the step size if not input.

$$h = \sqrt{\varepsilon}$$

2. Redefine $\mathbf{x}_0$ by updating its j^{th} element.

$$x_{0,j} = x_{0,j} + ih$$

3. Evaluate the partial derivative of $\mathbf{f}(\mathbf{x})$ with respect to x_j at $\mathbf{x} = \mathbf{x}_0$ using the complex-step approximation.

$$\left.\frac{\partial \mathbf{f}}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_0} \approx \frac{\operatorname{Im}\left[\mathbf{f}(\mathbf{x}_0)\right]}{h}$$

Return:

- $\left.\dfrac{\partial \mathbf{f}}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_0} \in \mathbb{R}^m$ - partial derivative of $\mathbf{f}(\mathbf{x})$ with respect to x_j, evaluated at $\mathbf{x} = \mathbf{x}_0$

Note:

- This algorithm can be used for a scalar-valued function by just inputting a function where $m = 1$ (i.e. $f:\mathbb{R}^n \to \mathbb{R}$ instead of $\mathbf{f}:\mathbb{R}^n \to \mathbb{R}^m$).
- This function requires 1 evaluation of $\mathbf{f}(\mathbf{x})$.

2.3 Gradient

Consider a multivariate, scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$. The gradient of f with respect to $\mathbf{x} \in \mathbb{R}^n$, evaluated at $\mathbf{x} = \mathbf{x}_0$, is defined as [6]

$$\boxed{\mathbf{g}(\mathbf{x}_0) = \nabla f(\mathbf{x}_0) = \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_0} = \begin{bmatrix} \left.\dfrac{\partial f}{\partial x_1}\right|_{\mathbf{x}=\mathbf{x}_0} \\ \vdots \\ \left.\dfrac{\partial f}{\partial x_n}\right|_{\mathbf{x}=\mathbf{x}_0} \end{bmatrix}} \tag{2.6}$$

To evaluate the gradient, we need to approximate all of the partial derivatives. From Eq. (2.4), we know the j^{th} partial derivative can be approximated as

$$\frac{\partial f}{\partial x_j} \approx \frac{\operatorname{Im}\left[f(\mathbf{x} + \boldsymbol{\chi}_j)\right]}{h}$$

where

$$\boldsymbol{\chi}_j = \begin{bmatrix} 0 \\ \vdots \\ ih \\ \vdots \\ 0 \end{bmatrix} \leftarrow j^{\text{th}} \text{ element}$$

To make the computational implementation a bit neater and more efficient, we introduce the complex-step matrix, $\mathbf{X}$, defined as

$$\boxed{\mathbf{X} = ih\mathbf{I}_{n\times n} = \begin{bmatrix} ih & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & ih \end{bmatrix} = \begin{bmatrix} | & & | \\ \boldsymbol{\chi}_1 & \dots & \boldsymbol{\chi}_n \\ | & & | \end{bmatrix}} \tag{2.7}$$

As evident from Eq. (2.7), $\boldsymbol{\chi}_j$ is simply the j^{th} column of $\mathbf{X}$. Now, we can develop the algorithm to approximate the gradient.

Algorithm 4: `igradient`

Gradient of a multivariate, scalar-valued function using the complex-step approximation.

Given:

- $f(\mathbf{x})$ - multivariate, scalar-valued function ($f : \mathbb{R}^n \to \mathbb{R}$)
- $\mathbf{x}_0 \in \mathbb{R}^n$ - point at which to evaluate the gradient
- $h \in \mathbb{R}$ - *(OPTIONAL)* step size

Procedure:

1. Initialize the step size if not input.

   $$h = \sqrt{\varepsilon}$$

2. Determine n from the fact that $\mathbf{x}_0 \in \mathbb{R}^n$.
3. Preallocate a vector $\mathbf{g} \in \mathbb{R}^n$ to store the gradient.
4. Define the complex-step matrix.

   $$\mathbf{X} = ih\mathbf{I}_{n\times n}$$

5. Evaluate the gradient.

for $j = 1$ **to** n

$$g_j = \frac{\mathrm{Im}\left[f(\mathbf{x}_0 + \boldsymbol{\chi}_j)\right]}{h} \quad \text{(where } \boldsymbol{\chi}_j \text{ is the } j^{\text{th}} \text{ column of } \mathbf{X}\text{)}$$

end

Return:

- $\mathbf{g} \in \mathbb{R}^n$ - gradient of f evaluated at $\mathbf{x} = \mathbf{x}_0$

Note:

- This function requires n evaluations of $f(\mathbf{x})$.

2.4 Directional Derivative

Consider a multivariate, scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$. The directional derivative of f at $\mathbf{x} = \mathbf{x}_0$ in the direction of $\mathbf{v} \in \mathbb{R}^n$ is [4]

$$\boxed{D_{\mathbf{v}} f(\mathbf{x}_0) = \mathbf{v} \cdot \nabla f(\mathbf{x}_0)} \tag{2.8}$$

Therefore, we can simply add a single additional line to Algorithm 4 to develop Algorithm 5 to calculate the directional derivative.

Algorithm 5: `idirectional`

Directional derivative of a multivariate, scalar-valued function using the complex-step approximation.

Given:

- $f(\mathbf{x})$ - multivariate, scalar-valued function ($f : \mathbb{R}^n \to \mathbb{R}$)
- $\mathbf{x}_0 \in \mathbb{R}^n$ - point at which to evaluate the directional derivative
- $\mathbf{v} \in \mathbb{R}^n$ - vector defining direction of differentiation
- $h \in \mathbb{R}$ - *(OPTIONAL)* step size

Procedure:

1. Initialize the step size if not input.

$$h = \sqrt{\varepsilon}$$

2. Determine n from the fact that $\mathbf{x}_0 \in \mathbb{R}^n$.
3. Preallocate a vector $\mathbf{g} \in \mathbb{R}^n$ to store the gradient.
4. Define the complex-step matrix.

$$\mathbf{X} = ih\mathbf{I}_{n\times n}$$

5. Evaluate the gradient.

for $j = 1$ **to** n

$$g_j = \frac{\mathrm{Im}\left[f(\mathbf{x}_0 + \boldsymbol{\chi}_j)\right]}{h} \quad \text{(where } \boldsymbol{\chi}_j \text{ is the } j^{\text{th}} \text{ column of } \mathbf{X}\text{)}$$

end

6. Evaluate the directional derivative.

$$D_{\mathbf{v}} = \mathbf{v} \cdot \mathbf{g}$$

Return:

- $D_{\mathbf{v}} \in \mathbb{R}$ - directional derivative of f evaluated at $\mathbf{x} = \mathbf{x}_0$ in the direction of $\mathbf{v}$

Note:

- This function requires n evaluations of $f(\mathbf{x})$.

2.5 Jacobian Matrix

The Jacobian matrix of a multivariate, vector-valued function $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ with respect to $\mathbf{x} \in \mathbb{R}^n$, evaluated at $\mathbf{x} = \mathbf{x}_0$, is defined as

$$\boxed{\mathbf{J}(\mathbf{x}_0) = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_0} = \begin{bmatrix} \left.\frac{\partial f_1}{\partial x_1}\right|_{\mathbf{x}=\mathbf{x}_0} & \cdots & \left.\frac{\partial f_1}{\partial x_n}\right|_{\mathbf{x}=\mathbf{x}_0} \\ \vdots & \ddots & \vdots \\ \left.\frac{\partial f_m}{\partial x_1}\right|_{\mathbf{x}=\mathbf{x}_0} & \cdots & \left.\frac{\partial f_m}{\partial x_n}\right|_{\mathbf{x}=\mathbf{x}_0} \end{bmatrix}} \tag{2.9}$$

It is helpful to rewrite Eq. (2.9) in terms of its column vectors as [10]

$$\boxed{\mathbf{J}(\mathbf{x}_0) = \begin{bmatrix} \left.\frac{\partial \mathbf{f}}{\partial x_1}\right|_{\mathbf{x}=\mathbf{x}_0} & \cdots & \left.\frac{\partial \mathbf{f}}{\partial x_n}\right|_{\mathbf{x}=\mathbf{x}_0} \end{bmatrix}} \tag{2.10}$$

From Eq. (2.5), we know

$$\left.\frac{\partial \mathbf{f}}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_0} \approx \frac{\operatorname{Im}\left[\mathbf{f}(\mathbf{x}_0 + \boldsymbol{\chi}_j)\right]}{h}$$

where

$$\boldsymbol{\chi}_j = \begin{bmatrix} 0 \\ \vdots \\ ih \\ \vdots \\ 0 \end{bmatrix} \leftarrow j^{\text{th}} \text{ element}$$

Utilizing Eq. (2.10), we only need to make a small adjustment to Algorithm 4 to develop Algorithm 6 for evaluating the Jacobian using the complex-step approximation.

Algorithm 6: `ijacobian`

Jacobian matrix of a multivariate, vector-valued function using the complex-step approximation.

Given:

- $\mathbf{f}(\mathbf{x})$ - multivariate, vector-valued function ($\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$)
- $\mathbf{x}_0 \in \mathbb{R}^n$ - point at which to evaluate the Jacobian matrix
- $h \in \mathbb{R}$ - *(OPTIONAL)* step size

Procedure:

1. Initialize the step size if not input.

$$h = \sqrt{\varepsilon}$$

2. Determine m from the fact that $\mathbf{f}(\mathbf{x}_0) \in \mathbb{R}^m$.
3. Determine n from the fact that $\mathbf{x}_0 \in \mathbb{R}^n$.
4. Preallocate a matrix $\mathbf{J} \in \mathbb{R}^{m \times n}$ to store the Jacobian.
5. Define the complex-step matrix.

$$\mathbf{X} = ih\mathbf{I}_{n \times n}$$

6. Evaluate the Jacobian matrix (where $\mathbf{j}_j$ is the j^{th} column vector of $\mathbf{J}$).

for $j = 1$ **to** n

$$\mathbf{j}_j = \frac{\operatorname{Im}\left[f(\mathbf{x}_0 + \boldsymbol{\chi}_j)\right]}{h} \quad \text{(where } \boldsymbol{\chi}_j \text{ is the } j^{\text{th}} \text{ column of } \mathbf{X}\text{)}$$

end

Return:

- $\mathbf{J} \in \mathbb{R}^{m \times n}$ - Jacobian matrix of $\mathbf{f}$ evaluated at $\mathbf{x} = \mathbf{x}_0$

Note:

- This function requires n evaluations of $\mathbf{f}(\mathbf{x})$.

2.6 Hessian Matrix

The Hessian matrix of a multivariate, scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$ with respect to $\mathbf{x} \in \mathbb{R}^n$, evaluated at $\mathbf{x} = \mathbf{x}_0$, is defined as

$$\boxed{\mathbf{H}(\mathbf{x}_0) = \begin{bmatrix} \left.\dfrac{\partial^2 f}{\partial x_1^2}\right|_{\mathbf{x}=\mathbf{x}_0} & \left.\dfrac{\partial^2 f}{\partial x_1 \partial x_2}\right|_{\mathbf{x}=\mathbf{x}_0} & \cdots & \left.\dfrac{\partial^2 f}{\partial x_1 \partial x_n}\right|_{\mathbf{x}=\mathbf{x}_0} \\ \left.\dfrac{\partial^2 f}{\partial x_2 \partial x_1}\right|_{\mathbf{x}=\mathbf{x}_0} & \left.\dfrac{\partial^2 f}{\partial x_2^2}\right|_{\mathbf{x}=\mathbf{x}_0} & \cdots & \left.\dfrac{\partial^2 f}{\partial x_2 \partial x_n}\right|_{\mathbf{x}=\mathbf{x}_0} \\ \vdots & \vdots & \ddots & \vdots \\ \left.\dfrac{\partial^2 f}{\partial x_n \partial x_1}\right|_{\mathbf{x}=\mathbf{x}_0} & \left.\dfrac{\partial^2 f}{\partial x_n \partial x_2}\right|_{\mathbf{x}=\mathbf{x}_0} & \cdots & \left.\dfrac{\partial^2 f}{\partial x_n^2}\right|_{\mathbf{x}=\mathbf{x}_0} \end{bmatrix}} \tag{2.11}$$

In a more compact form, we can write

$$\boxed{\left[\mathbf{H}(\mathbf{x}_0)\right]_{j,k} = \frac{\partial^2 f}{\partial x_j \partial x_k}} \tag{2.12}$$

From Schwarz's theorem, we know

$$\frac{\partial^2 f}{\partial x_j \partial x_k} = \frac{\partial^2 f}{\partial x_k \partial x_j}$$

which implies that the Hessian matrix is symmetric and satisfies the property [8]

$$\boxed{\left[\mathbf{H}(\mathbf{x}_0)\right]_{j,k} = \left[\mathbf{H}(\mathbf{x}_0)\right]_{k,j}} \tag{2.13}$$

As mentioned in Section 1.3.1, we will not be using a true complex-step approximation for the second derivative in this section. Instead, we will evaluate the first derivative using a complex-step approximation, and the second derivative using a central difference approximation. Beginning by rewriting Eq. (2.12) in a slightly different form,

$$[\mathbf{H}(\mathbf{x}_0)]_{j,k} = \left.\frac{\partial}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_0} \left(\left.\frac{\partial f}{\partial x_k}\right|_{\mathbf{x}=\mathbf{x}_0} \right)$$

Replacing the derivative in the parentheses with its complex-step approximation (from Eq. (2.4)), we can write

$$[\mathbf{H}(\mathbf{x}_0)]_{j,k} \approx \left.\frac{\partial}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_0} \left(\frac{\operatorname{Im}\left[f(\mathbf{x}_0 + \boldsymbol{\chi}_k)\right]}{h} \right) \tag{2.14}$$

where

$$\boldsymbol{\chi}_k = \begin{bmatrix} 0 \\ \vdots \\ ih \\ \vdots \\ 0 \end{bmatrix} \leftarrow k^{\text{th}} \text{ element}$$

The central difference approximation to a derivative is [17]

$$\left.\frac{df}{dx}\right|_{x=x_0} \approx \frac{f(x_0+h) - f(x_0-h)}{2h}$$

For a multivariate function, this can be written as

$$\left.\frac{\partial f}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_0} \approx \frac{f(\mathbf{x}_0 + \mathbf{u}_j) - f(\mathbf{x}_0 - \mathbf{u}_j)}{2h} \tag{2.15}$$

where

$$\boxed{\mathbf{u}_j = \begin{bmatrix} 0 \\ \vdots \\ h \\ \vdots \\ 0 \end{bmatrix} \leftarrow j^{\text{th}} \text{ element}} \tag{2.16}$$

Applying the approximation from Eq. (2.15) to Eq. (2.14),

$$\begin{aligned}
[\mathbf{H}(\mathbf{x}_0)]_{j,k} &\approx \frac{1}{2h}\left[\left(\frac{\operatorname{Im}\left[f(\mathbf{x}_0 + \boldsymbol{\chi}_k) + \mathbf{u}_j\right]}{h} \right) - \left(\frac{\operatorname{Im}\left[f(\mathbf{x}_0 + \boldsymbol{\chi}_k - \mathbf{u}_j)\right]}{h} \right) \right] \\
&\approx \frac{\operatorname{Im}\left[f(\mathbf{x}_0 + \boldsymbol{\chi}_k + \mathbf{u}_j)\right] - \operatorname{Im}\left[f(\mathbf{x}_0 + \boldsymbol{\chi}_k - \mathbf{u}_j)\right]}{2h^2} \\
&\approx \frac{\operatorname{Im}\left[f(\mathbf{x}_0 + \boldsymbol{\chi}_k + \mathbf{u}_j) - f(\mathbf{x}_0 + \boldsymbol{\chi}_k - \mathbf{u}_j)\right]}{2h^2}
\end{aligned}$$

Finally, let's define

$$\boxed{\mathbf{x}_{r,k} = \mathbf{x}_0 + \boldsymbol{\chi}_k} \tag{2.17}$$

$$\boxed{\mathbf{y}_{j,k}^{+} = \mathbf{x}_{r,k} + \mathbf{u}_j} \tag{2.18}$$

$$\boxed{\mathbf{y}_{j,k}^{-} = \mathbf{x}_{r,k} - \mathbf{u}_j} \tag{2.19}$$

Then we arrive at the result

$$\boxed{[\mathbf{H}(\mathbf{x}_0)]_{j,k} = [\mathbf{H}(\mathbf{x}_0)]_{k,j} \approx \frac{\operatorname{Im}\left[f\left(\mathbf{y}_{j,k}^{+}\right) - f\left(\mathbf{y}_{j,k}^{-}\right)\right]}{2h^2}} \tag{2.20}$$

Similar to the complex-step matrix $\mathbf{X}$, we define the real-step matrix $\mathbf{U}$ as

$$\boxed{\mathbf{U} = h\mathbf{I}_{n\times n} = \begin{bmatrix} h & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & h \end{bmatrix} = \begin{bmatrix} | & & | \\ \mathbf{u}_1 & \dots & \mathbf{u}_n \\ | & & | \end{bmatrix}} \tag{2.21}$$

With these definitions, we can develop Algorithm 7 below[1]. Since the Hessian matrix is symmetric, we only have to evaluate the derivatives in the upper triangle of the matrix (shown in red below):

$$\begin{bmatrix} H_{1,1} & H_{1,2} & \dots & H_{1,n} \\ H_{2,1} & H_{2,2} & \dots & H_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ H_{n,1} & H_{n,2} & \dots & H_{n,n} \end{bmatrix}$$

Algorithm 7: `ihessian`

Hessian matrix of a multivariate, scalar-valued function using the complex-step and central difference approximations.

Given:

- $f(\mathbf{x})$ - multivariate, scalar-valued function ($f:\mathbb{R}^n \to \mathbb{R}$)
- $\mathbf{x}_0 \in \mathbb{R}^n$ - point at which to evaluate the Hessian matrix
- $h \in \mathbb{R}$ - *(OPTIONAL)* step size

Procedure:

1. Initialize the step size if not input.

$$h = \sqrt{\varepsilon}$$

2. Determine n from the fact that $\mathbf{x}_0 \in \mathbb{R}^n$.
3. Preallocate a matrix $\mathbf{H} \in \mathbb{R}^{n\times n}$ to store the Hessian.
4. Define the complex-step and real-step matrices.

$$\mathbf{X} = ih\mathbf{I}_{n\times n}$$
$$\mathbf{U} = h\mathbf{I}_{n\times n}$$

5. Loop over each independent variable.

for $k = 1$ **to** n

[1] This algorithm was inspired by [2] and [7].

(a) Define the reference point with the complex increment in the k^{th} independent variable.

$$\mathbf{x}_r = \mathbf{x}_0 + \boldsymbol{\chi}_k$$

(b) Loop through the upper triangular elements.

for $j = k$ **to** n

i. Define the reference points with the real increments in the j^{th} independent variable.

$$\mathbf{y}^+ = \mathbf{x}_r + \mathbf{u}_j$$
$$\mathbf{y}^- = \mathbf{x}_r - \mathbf{u}_j$$

ii. Evaluate the $(k,j)^{\text{th}}$ element of the Hessian.

$$H_{k,j} \approx \frac{\operatorname{Im}\left[f(\mathbf{y}^+) - f(\mathbf{y}^-)\right]}{2h^2}$$

iii. Evaluate the $(j,k)^{\text{th}}$ element of the Hessian using symmetry.

$$H_{j,k} = H_{k,j}$$

end

end

Return:

- $\mathbf{H} \in \mathbb{R}^{n \times n}$ - Hessian matrix of f evaluated at $\mathbf{x} = \mathbf{x}_0$

Note:

- $\boldsymbol{\chi}_k$ is the k^{th} column of $\mathbf{X}$ and $\mathbf{u}_j$ is the j^{th} column of $\mathbf{U}$.
- This function requires $n(n+1)$ evaluations of $f(\mathbf{x})$ (the upper triangular matrix entries (including the diagonal) consist of $n(n+1)/2$ entries [9], and each entry requires 2 evaluations of $f(\mathbf{x})$).

PART II

Backward Difference Approximation

3

The Backward-Difference Approximation

3.1 Definition

Consider a univariate function $f(x)$. Eq. (3.1) computes the backward difference approximation of its derivative at $x = x_i$ [5, 17].

$$\left.\frac{df}{dx}\right|_{x=x_i} \approx \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}} \tag{3.1}$$

The finite difference stencil for this approximation is shown in Fig. 3.1.

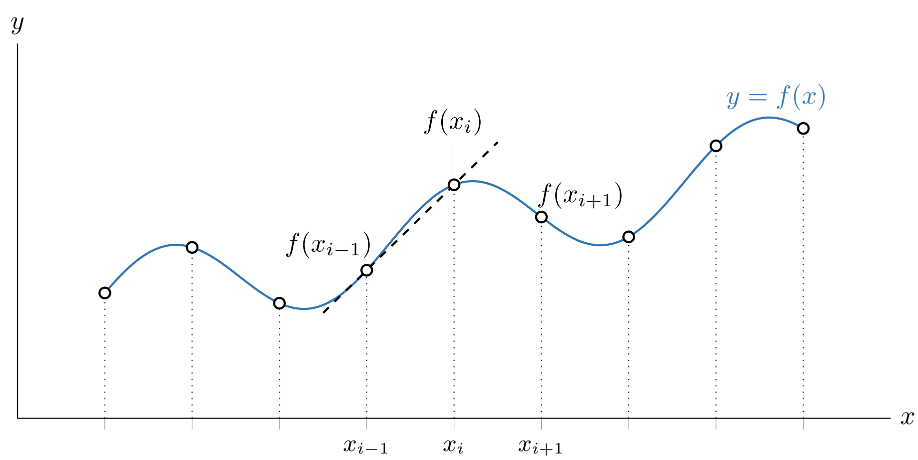

Figure 3.1: Backward difference approximation.

Consider the case where we are given two points, x_a and x_b, and would like to approximate the derivative of $f(x)$ at x_b. Using the backward difference approximation, we find

$$\boxed{\left.\frac{df}{dx}\right|_{x=x_b} \approx \frac{f(x_b) - f(x_a)}{x_b - x_a}} \tag{3.2}$$

3.2 Implementation

Algorithm 8: `derivative2_s`

Derivative of a univariate, scalar-valued function using the backward difference approximation (given two points).

Given:

- $f(x)$ - univariate, scalar-valued function ($f:\mathbb{R}\to\mathbb{R}$)
- $x_a \in \mathbb{R}$ - auxiliary point
- $x_b \in \mathbb{R}$ - point at which to differentiate

Procedure:

$$\left.\frac{df}{dx}\right|_{x=x_b} \approx \frac{f(x_b)-f(x_a)}{x_b-x_a}$$

Return:

- $\left.\frac{df}{dx}\right|_{x=x_b} \in \mathbb{R}$ - derivative of $f(x)$ evaluated at $x=x_b$

Note: We succeed all algorithm names with "2" to indicate that the algorithm is using the backward difference derivative approximation, which requires **two** points.

3.3 Points Used for the Approximation

The backward difference approximation requires **two** points to evaluate the derivative. In Section 3.1, we approximate the derivative of a univariate, scalar-valued function $f(x)$ at $x=x_b$ using the points $(x_a, f(x_a))$ and $(x_b, f(x_b))$. In Chapter 4, we will be extending the backward difference approximation for multivariate differentiation. Consequently, we need to generalize these points as the vectors $\mathbf{x}_a$ and $\mathbf{x}_b$ in the vector space $\mathbb{R}^n$. Additionally, we will introduce another point (the partial point) that is defined using $\mathbf{x}_a$ and $\mathbf{x}_b$ and is essential to approximating partial derivatives.

$$\begin{aligned}
\mathbf{x}_a &\in \mathbb{R}^n && \text{(auxiliary point)} \\
\mathbf{x}_b &\in \mathbb{R}^n && \text{(evaluation point)} \\
\mathbf{x}_j &\in \mathbb{R}^n && \text{(partial point)}
\end{aligned}$$

We refer to $\mathbf{x}_a$ as the **auxiliary point**, since it is purely there to aid us in approximating the derivative. The point $\mathbf{x}_b$ is referred to as the **evaluation point** since it is where we actually wish to evaluate the derivative.

The elements of $\mathbf{x}_a$ and $\mathbf{x}_b$ are denoted using the following convention:

$$\boxed{\mathbf{x}_a = \begin{bmatrix} x_{a,1} \\ x_{a,2} \\ \vdots \\ x_{a,j} \\ \vdots \\ x_{a,n} \end{bmatrix}, \qquad \mathbf{x}_b = \begin{bmatrix} x_{b,1} \\ x_{b,2} \\ \vdots \\ x_{b,j} \\ \vdots \\ x_{b,n} \end{bmatrix}} \tag{3.3}$$

We refer to the third point, $\mathbf{x}_j$, as the **partial point**, since it aids us in approximating partial derivatives (which are essential in building up other multivariate derivatives such as gradients and Jacobians). Essentially, $\mathbf{x}_j$ is formed by replacing the j^{th} element of $\mathbf{x}_b$ with the j^{th} element of $\mathbf{x}_a$.

$$\boxed{\mathbf{x}_j = \begin{bmatrix} x_{b,1} \\ x_{b,2} \\ \vdots \\ x_{b,j-1} \\ x_{a,j} \\ x_{b,j+1} \\ \vdots \\ x_{b,n} \end{bmatrix}} \tag{3.4}$$

3.4 Applications

The backward difference approximation is particularly useful for root-finding and optimization algorithms that update the location, $\mathbf{x}$, of either the root or minimum of the function, using multivariate derivatives. For example, given an initial guess x_0 for the root of $f(x)$, Newton's method uses $f'(x)$ to iteratively update the root location until $f(x_0) \approx 0$ (to within some tolerance). Specifically, the estimate of the root location is updated as [1, 14]

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

If we don't know the derivative $f'(x)$, then we can use the backward difference approximation between the points x_{i-1} and x_i to approximate $f'(x)$:

$$f'(x_i) \approx \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}}$$

Using this approximation for $f'(x)$ results in the secant method [1, 15]. For other numerical root finding and optimization methods, we can use the algorithms developed in Chapter 4 to approximate derivatives and multivariate derivatives (such as gradients and Jacobians) wherever needed.

However, it is important to note that the algorithms in Chapter 2 (based on the complex-step approximation) can also be used to approximate these derivatives. In fact, the derivatives obtained by the algorithms in Chapter 2 will be much more accurate, and the algorithms themselves are typically faster as well[1].

[1] The main computational cost is usually incurred through evaluations of the function f or $\mathbf{f}$, and comparing the algorithms in Chapters 2 and 4, we see that the algorithms relying on the backward difference approximation require exactly twice as many function evaluations.

4

Multivariate Differentiation Using the Backward Difference Approximation

4.1 Derivative of a Univariate, Vector-Valued Function

Consider a univariate, vector-valued function $\mathbf{f} : \mathbb{R} \to \mathbb{R}^m$. To approximate its derivative at a point $x = x_b$, we can begin by simply differentiating each component of $\mathbf{f}$ with respect to x and evaluting the result at $x = x_b$.

$$\left.\frac{d\mathbf{f}}{dx}\right|_{x=x_b} = \begin{bmatrix} \left.\dfrac{df_1}{dx}\right|_{x=x_b} \\ \vdots \\ \left.\dfrac{df_m}{dx}\right|_{x=x_b} \end{bmatrix}$$

Applying the backward difference approximation,

$$\left.\frac{d\mathbf{f}}{dx}\right|_{x=x_b} \approx \begin{bmatrix} \dfrac{f_1(x_b) - f_1(x_a)}{x_b - x_a} \\ \vdots \\ \dfrac{f_m(x_b) - f_m(x_a)}{x_b - x_a} \end{bmatrix}$$

In a more compact form, we can simply write

$$\boxed{\left.\frac{d\mathbf{f}}{dx}\right|_{x=x_b} \approx \frac{\mathbf{f}(x_b) - \mathbf{f}(x_a)}{x_b - x_a}} \tag{4.1}$$

For completeness, we include Algorithm 9. However, it should be noted that Algorithm 9 is nearly identical to Algorithm 8, except it is generalized to a vector-valued function $\mathbf{f}$.

Algorithm 9: `derivative2`
Derivative of a univariate, vector-valued function using the backward difference approximation (given two points).

Given:
- $\mathbf{f}(x)$ - univariate, vector-valued function ($\mathbf{f}:\mathbb{R}\to\mathbb{R}^m$)
- $x_a\in\mathbb{R}$ - auxiliary point
- $x_b\in\mathbb{R}$ - point at which to differentiate

Procedure:

$$\left.\frac{d\mathbf{f}}{dx}\right|_{x=x_b}\approx\frac{\mathbf{f}(x_b)-\mathbf{f}(x_a)}{x_b-x_a}$$

Return:
- $\left.\frac{d\mathbf{f}}{dx}\right|_{x=x_b}\in\mathbb{R}^m$ - derivative of $\mathbf{f}(x)$ evaluated at x_b

Note:
- This function requires 2 evaluations of $\mathbf{f}(x)$.

4.2 Partial Derivative of a Multivariate, Scalar or Vector-Valued Function

Let's begin by considering a bivariate, scalar-valued function $f(x,y)$. The backward approximation for the *partial* derivatives of f with respect to x and y, respectively, are

$$\begin{aligned}\left.\frac{\partial f}{\partial x}\right|_{(x,y)=(x_i,y_i)}&\approx\frac{f(x_i,y_i)-f(x_{i-1},y_i)}{x_i-x_{i-1}}\\ \left.\frac{\partial f}{\partial y}\right|_{(x,y)=(x_i,y_i)}&\approx\frac{f(x_i,y_i)-f(x_i,y_{i-1})}{y_i-y_{i-1}}\end{aligned}\tag{4.2}$$

Next, let's generalize to n dimensions, where $f=f(\mathbf{x})$ and $\mathbf{x}=(x_1,...,x_n)^T\in\mathbb{R}^n$. As a first step, let's consider the case where $\mathbf{x}=(x_1,x_2)^T$. Finding the partial derivative of f with respect to x_1 is simply an exercise of rewriting Eq. (4.2) with the substitutions $x=x_1$ and $y=x_2$.

$$\left.\frac{\partial f}{\partial x_1}\right|_{(x_1,x_2)=(x_{1,i},x_{2,i})}\approx\frac{f(x_{i,1},x_{i,2})-f(x_{i-1,1},x_{i,2})}{x_{i,1}-x_{i-1,1}}$$

Using the above expression, it is easier to see how this can be extended to n variables. Taking the partial derivative of $f(x_1,...,x_n)$ with respect to x_j (where $1\le j\le n$),

$$\left.\frac{\partial f}{\partial x_j}\right|_{(x_1,...,x_n)=(x_{1,i},...,x_{n,i})}\approx\frac{f(x_{i,1},...,x_{i,j-1},x_{i,j},x_{i,j+1},...,x_{i,n})-f(x_{i,1},...,x_{i,j-1},x_{i-1,j},x_{i,j+1},...,x_{i,n})}{x_{i,j}-x_{i-1,j}}$$

In vector form,

$$\left.\frac{\partial f}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_i} \approx \frac{f\left(\begin{bmatrix} x_{i,1} \\ \vdots \\ x_{i,j-1} \\ x_{i,j} \\ x_{i,j+1} \\ \vdots \\ x_{i,n} \end{bmatrix}\right) - f\left(\begin{bmatrix} x_{i,1} \\ \vdots \\ x_{i,j-1} \\ x_{i-1,j} \\ x_{i,j+1} \\ \vdots \\ x_{i,n} \end{bmatrix}\right)}{x_{i,j} - x_{i-1,j}} \tag{4.3}$$

Now, let's restrict ourselves to the case where we have two points, $\mathbf{x}_a$ and $\mathbf{x}_b$, as defined previously by Eq. (3.3):

$$\mathbf{x}_a = \begin{bmatrix} x_{a,1} \\ x_{a,2} \\ \vdots \\ x_{a,j} \\ \vdots \\ x_{a,n} \end{bmatrix}, \qquad \mathbf{x}_b = \begin{bmatrix} x_{b,1} \\ x_{b,2} \\ \vdots \\ x_{b,j} \\ \vdots \\ x_{b,n} \end{bmatrix}$$

We wish to evaluate the partial derivative of $\mathbf{f}$ with respect to x_j at $\mathbf{x}_b$. Then, from Eq. (4.3), we have

$$\left.\frac{\partial f(\mathbf{x})}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_b} \approx \frac{f\left(\begin{bmatrix} x_{b,1} \\ \vdots \\ x_{b,j-1} \\ x_{b,j} \\ x_{b,j+1} \\ \vdots \\ x_{b,n} \end{bmatrix}\right) - f\left(\begin{bmatrix} x_{b,1} \\ \vdots \\ x_{b,j-1} \\ x_{a,j} \\ x_{b,j+1} \\ \vdots \\ x_{b,n} \end{bmatrix}\right)}{x_{b,j} - x_{a,j}}$$

Recalling the definitions of $\mathbf{x}_b$ and $\mathbf{x}_j$ from Eqs. (3.3) and (3.4), respectively, the above equation can be written simply as

$$\left.\frac{\partial f(\mathbf{x})}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_b} \approx \frac{f\left(\mathbf{x}_b\right) - f\left(\mathbf{x}_j\right)}{x_{b,j} - x_{a,j}} \tag{4.4}$$

From Section 4.1, we know it is trivial to extend this to the vector-valued case ($\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$):

$$\boxed{\left.\frac{\partial \mathbf{f}}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_b} \approx \frac{\mathbf{f}(\mathbf{x}_b) - \mathbf{f}(\mathbf{x}_j)}{x_{b,j} - x_{a,j}}} \tag{4.5}$$

Algorithm 10: `partial2`

Partial derivative of a multivariate, scalar or vector-valued function using the backward difference approximation (given two points).

Given:

- $\mathbf{f}(\mathbf{x})$ - multivariate, vector-valued function ($\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$)
- $\mathbf{x}_a \in \mathbb{R}^n$ - auxiliary point
- $\mathbf{x}_b \in \mathbb{R}^n$ - point at which to differentiate
- $j \in \mathbb{Z}$ - index of element of $\mathbf{x}$ to differentiate with respect to

Procedure:

1. Define the partial point, $\mathbf{x}_j$.

$$\mathbf{x}_j = \begin{bmatrix} x_{b,1} \\ x_{b,2} \\ \vdots \\ x_{b,j-1} \\ x_{a,j} \\ x_{b,j+1} \\ \vdots \\ x_{b,n} \end{bmatrix}$$

2. Evaluate the partial derivative of $\mathbf{f}(\mathbf{x})$ with respect to x_j at $\mathbf{x} = \mathbf{x}_b$ using the backward difference approximation.

$$\left.\frac{\partial \mathbf{f}}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_b} \approx \frac{\mathbf{f}(\mathbf{x}_b) - \mathbf{f}(\mathbf{x}_j)}{x_{b,j} - x_{a,j}}$$

Return:

- $\left.\dfrac{\partial \mathbf{f}}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_b} \in \mathbb{R}^m$ - partial derivative of $\mathbf{f}(\mathbf{x})$ with respect to x_j, evaluated at $\mathbf{x} = \mathbf{x}_b$

Note:

- This algorithm can be used for a scalar-valued function by just inputting a function where $m = 1$ (i.e. $f : \mathbb{R}^n \to \mathbb{R}$ instead of $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$).
- This function requires 2 evaluations of $\mathbf{f}(\mathbf{x})$.

4.3 Gradient

Consider a multivariate, scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$. The gradient of f with respect to $\mathbf{x} \in \mathbb{R}^n$, evaluated at $\mathbf{x} = \mathbf{x}_b$, is defined as [6]

$$\boxed{\mathbf{g}(\mathbf{x}_b) = \nabla f(\mathbf{x}_b) = \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_b} = \begin{bmatrix} \left.\dfrac{\partial f}{\partial x_1}\right|_{\mathbf{x}=\mathbf{x}_b} \\ \vdots \\ \left.\dfrac{\partial f}{\partial x_n}\right|_{\mathbf{x}=\mathbf{x}_b} \end{bmatrix}} \tag{4.6}$$

To evaluate the gradient, we need to approximate all of the partial derivatives. From Eq. (4.4), we know the j^{th} partial derivative can be approximated as

$$\left.\frac{\partial f(\mathbf{x})}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_b} \approx \frac{f(\mathbf{x}_b) - f(\mathbf{x}_j)}{x_{b,j} - x_{a,j}}$$

Now, we can develop the algorithm to approximate the gradient.

Algorithm 11: `gradient2`

Gradient of a multivariate, scalar-valued function using the backward difference approximation (given two points).

Given:

- $f(\mathbf{x})$ - multivariate, scalar-valued function ($f : \mathbb{R}^n \to \mathbb{R}$)
- $\mathbf{x}_a \in \mathbb{R}^n$ - auxiliary point
- $\mathbf{x}_b \in \mathbb{R}^n$ - point at which to evaluate the gradient

Procedure:

1. Determine n from the fact that $\mathbf{x}_a \in \mathbb{R}^n$.
2. Preallocate a vector $\mathbf{g} \in \mathbb{R}^n$ to store the gradient.
3. Evaluate the gradient.

for $j = 1$ **to** n

(a) Define the partial point, $\mathbf{x}_j$.

$$\mathbf{x}_j = \begin{bmatrix} x_{b,1} \\ x_{b,2} \\ \vdots \\ x_{b,j-1} \\ x_{a,j} \\ x_{b,j+1} \\ \vdots \\ x_{b,n} \end{bmatrix}$$

(b) Evaluate the partial derivative of $f(\mathbf{x})$ with respect to x_j at $\mathbf{x}_b$ and store it in the gradient vector, $\mathbf{g}$.

$$g_j = \frac{\mathbf{f}(\mathbf{x}_b) - \mathbf{f}(\mathbf{x}_j)}{x_{b,j} - x_{a,j}}$$

end

Return:

- $\mathbf{g} \in \mathbb{R}^n$ - gradient of f evaluated at $\mathbf{x} = \mathbf{x}_b$

Note:

- This function requires $2n$ evaluations of $f(\mathbf{x})$.

4.4 Directional Derivative

Consider a multivariate, scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$. The directional derivative of f at $\mathbf{x} = \mathbf{x}_b$ in the direction of $\mathbf{v} \in \mathbb{R}^n$ is [4]

$$\boxed{D_{\mathbf{v}} f(\mathbf{x}_b) = \mathbf{v} \cdot \nabla f(\mathbf{x}_b)} \tag{4.7}$$

Therefore, we can simply add a single additional line to Algorithm 11 to develop Algorithm 12 to calculate the directional derivative.

Algorithm 12: `directional2`

Directional derivative of a multivariate, scalar-valued function using the backward difference approximation (given two points).

Given:

- $f(\mathbf{x})$ - multivariate, scalar-valued function ($f : \mathbb{R}^n \to \mathbb{R}$)
- $\mathbf{x}_a \in \mathbb{R}^n$ - auxiliary point
- $\mathbf{x}_b \in \mathbb{R}^n$ - point at which to evaluate the directional derivative
- $\mathbf{v} \in \mathbb{R}^n$ - vector defining direction of differentiation

Procedure:

1. Determine n from the fact that $\mathbf{x}_a \in \mathbb{R}^n$.
2. Preallocate a vector $\mathbf{g} \in \mathbb{R}^n$ to store the gradient.
3. Evaluate the gradient.

for $j = 1$ **to** n

(a) Define the partial point, $\mathbf{x}_j$.

$$\mathbf{x}_j = \begin{bmatrix} x_{b,1} \\ x_{b,2} \\ \vdots \\ x_{b,j-1} \\ x_{a,j} \\ x_{b,j+1} \\ \vdots \\ x_{b,n} \end{bmatrix}$$

(b) Evaluate the partial derivative of $f(\mathbf{x})$ with respect to x_j at $\mathbf{x}_b$ and store it in the gradient vector, $\mathbf{g}$.

$$g_j = \frac{\mathbf{f}(\mathbf{x}_b) - \mathbf{f}(\mathbf{x}_j)}{x_{b,j} - x_{a,j}}$$

end

4. Evaluate the directional derivative.

$$D_{\mathbf{v}} = \mathbf{v} \cdot \mathbf{g}$$

Return:

- $D_{\mathbf{v}} \in \mathbb{R}$ - directional derivative of f evaluated at $\mathbf{x} = \mathbf{x}_b$ in the direction of $\mathbf{v}$

Note:

- This function requires $2n$ evaluations of $f(\mathbf{x})$.

4.5 Jacobian Matrix

The Jacobian matrix of a multivariate, vector-valued function $\mathbf{f}:\mathbb{R}^n\to\mathbb{R}^m$ with respect to $\mathbf{x}\in\mathbb{R}^n$, evaluated at $\mathbf{x}=\mathbf{x}_b$, is defined as

$$\boxed{\mathbf{J}(\mathbf{x}_b)=\left.\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_b}=\begin{bmatrix}\left.\frac{\partial f_1}{\partial x_1}\right|_{\mathbf{x}=\mathbf{x}_b} & \cdots & \left.\frac{\partial f_1}{\partial x_n}\right|_{\mathbf{x}=\mathbf{x}_b}\\ \vdots & \ddots & \vdots\\ \left.\frac{\partial f_m}{\partial x_1}\right|_{\mathbf{x}=\mathbf{x}_b} & \cdots & \left.\frac{\partial f_m}{\partial x_n}\right|_{\mathbf{x}=\mathbf{x}_b}\end{bmatrix}} \tag{4.8}$$

It is helpful to rewrite Eq. (4.8) in terms of its column vectors as [10]

$$\boxed{\mathbf{J}(\mathbf{x}_b)=\begin{bmatrix}\left.\frac{\partial\mathbf{f}}{\partial x_1}\right|_{\mathbf{x}=\mathbf{x}_b} & \cdots & \left.\frac{\partial\mathbf{f}}{\partial x_n}\right|_{\mathbf{x}=\mathbf{x}_b}\end{bmatrix}} \tag{4.9}$$

From Eq. (4.5), we know

$$\left.\frac{\partial\mathbf{f}}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_b}\approx\frac{\mathbf{f}(\mathbf{x}_b)-\mathbf{f}(\mathbf{x}_j)}{x_{b,j}-x_{a,j}}$$

Utilizing Eq. (4.9), we only need to make a small adjustment to Algorithm 11 to develop Algorithm 13 for evaluating the Jacobian using the backward difference approximation.

Algorithm 13: `jacobian2`
Jacobian matrix of a multivariate, vector-valued function using the backward difference approximation (given two points).

Given:

- $\mathbf{f}(\mathbf{x})$ - multivariate, vector-valued function ($\mathbf{f}:\mathbb{R}^n\to\mathbb{R}^m$)
- $\mathbf{x}_a\in\mathbb{R}^n$ - auxiliary point
- $\mathbf{x}_b\in\mathbb{R}^n$ - point at which to evaluate the Jacobian matrix

Procedure:

1. Determine m from the fact that $\mathbf{f}(\mathbf{x}_0)\in\mathbb{R}^m$.
2. Determine n from the fact that $\mathbf{x}_a\in\mathbb{R}^n$.
3. Preallocate a matrix $\mathbf{J}\in\mathbb{R}^{m\times n}$ to store the Jacobian.
4. Evaluate the Jacobian matrix.

for $j=1$ **to** n

(a) Define the partial point, $\mathbf{x}_j$.

$$\mathbf{x}_j = \begin{bmatrix} x_{b,1} \\ x_{b,2} \\ \vdots \\ x_{b,j-1} \\ x_{a,j} \\ x_{b,j+1} \\ \vdots \\ x_{b,n} \end{bmatrix}$$

(b) Evaluate the partial derivative of $\mathbf{f}(\mathbf{x})$ with respect to x_j at $\mathbf{x}_b$ and store it in $\mathbf{j}_j$ (the j^{th} column vector of $\mathbf{J}$).

$$\mathbf{j}_j = \frac{\mathbf{f}(\mathbf{x}_b) - \mathbf{f}(\mathbf{x}_j)}{x_{b,j} - x_{a,j}}$$

end

Return:

- $\mathbf{J} \in \mathbb{R}^{m \times n}$ - Jacobian matrix of $\mathbf{f}$ evaluated at $\mathbf{x} = \mathbf{x}_b$

Note:

- This function requires $2n$ evaluations of $\mathbf{f}(\mathbf{x})$.

References

[1] Richard L. Burden and J. Douglas Faires. "Newton's Method and Its Extensions". In: *Numerical Analysis*. 9th ed. Boston, MA: Brooks/Cole, Cengage Learning, 2011. Chap. 2.3, pp. 67–78.

[2] Yi Cao. *Complex step Hessian*. MATLAB Central File Exchange. Accessed: August 3, 2021. URL: `https://www.mathworks.com/matlabcentral/fileexchange/18177-complex-step-hessian?s_tid=srchtitle`.

[3] *complexify.f90*. MDO Lab. Accessed: August 3, 2021. URL: `https://mdolab.engin.umich.edu/misc/files/complexify.f90`.

[4] *Directional derivative*. Wikipedia. Accessed: August 3, 2021. URL: `https://en.wikipedia.org/wiki/Directional_derivative`.

[5] *Finite difference method*. Wikipedia. Accessed: November 24, 2019. URL: `https://en.wikipedia.org/wiki/Finite_difference_method`.

[6] *Gradient*. Wikipedia. Accessed: August 3, 2021. URL: `https://en.wikipedia.org/wiki/Gradient`.

[7] Daniel R. Herberg. `DTQP_hessian_complex_step` *from DT QP Project*. MATLAB Central File Exchange. Accessed: August 3, 2021. URL: `https://www.mathworks.com/matlabcentral/fileexchange/65434-dt-qp-project?s_tid=srchtitle`.

[8] *Hessian matrix*. Wikipedia. Accessed: August 3, 2021. URL: `https://en.wikipedia.org/wiki/Hessian_matrix`.

[9] *Is there an analytic expression for a number of elements inside a triangular matrix (with and without items on diagonal)*. Stack Exchange. Accessed: December 26, 2021. URL: `https://math.stackexchange.com/questions/2388887/is-there-an-analytic-expression-for-a-number-of-elements-inside-a-triangular-mat/2388889`.

[10] *Jacobian matrix and determinant*. Wikipedia. Accessed: August 3, 2021. URL: `https://en.wikipedia.org/wiki/Jacobian_matrix_and_determinant`.

[11] Joaquim R. R. A. Martins, Peter Sturdza, and Juan J. Alonso. "The Complex-Step Derivative Approximation". In: *ACM Transactions on Mathematical Software* 29.3 (Sept. 2003), pp. 245–262. DOI: `10.1145/838250.838251`.

[12] *Matlab Implementation*. MDO Lab. Accessed: August 3, 2021. URL: `https://mdolab.engin.umich.edu/misc/complex-step-guide-matlab`.

[13] Cleve Moler. *Complex Step Differentiation*. MathWorks. Accessed: August 3, 2021. URL: `https://blogs.mathworks.com/cleve/2013/10/14/complex-step-differentiation/`.

[14] *Newton's method*. Wikipedia. Accessed: June 10, 2020. URL: `https://en.wikipedia.org/wiki/Newton%27s_method`.

[15] *Secant method*. Wikipedia. Accessed: January 15, 2020. URL: `https://en.wikipedia.org/wiki/Secant_method`.

[16] William Squire and George Trapp. "Using Complex Variables to Estimate Derivatives of Real Functions". In: *SIAM Review* 40.1 (Mar. 1998), pp. 110–112. DOI: `10.1137/S003614459631241X`.

[17] Todd Young and Martin J. Mohlenkamp. *Lecture 27: Numerical Differentiation*. Introduction to Numerical Methods and Matlab Programming for Engineers. Accessed: August 3, 2021. URL: http://www.ohiouniversityfaculty.com/youngt/IntNumMeth/lecture27.pdf.