Adaptive Finite-Difference Interval Estimation for Noisy Derivative-Free Optimization

Hao-Jun Michael Shi* Yuchen Xie* Melody Qiming Xuan* Jorge Nocedal*

October 12, 2021

Abstract

A common approach for minimizing a smooth nonlinear function is to employ finite-difference approximations to the gradient. While this can be easily performed when no error is present within the function evaluations, when the function is noisy, the optimal choice requires information about the noise level and higher-order derivatives of the function, which is often unavailable. Given the noise level of the function, we propose a bisection search for finding a finite-difference interval for any finite-difference scheme that balances the *truncation error*, which arises from the error in the Taylor series approximation, and the *measurement error*, which results from noise in the function evaluation. Our procedure produces near-optimal estimates of the finite-difference interval at low cost without knowledge of the higher-order derivatives. We show its numerical reliability and accuracy on a set of test problems. When combined with L-BFGS, we obtain a robust method for minimizing noisy black-box functions, as illustrated on a subset of synthetically noisy unconstrained CUTEst problems.

Keywords: derivative-free optimization, noisy optimization, zeroth-order optimization, nonlinear optimization, finite differences

1 Introduction

A powerful approach for derivative-free optimization is to utilize finite differences. This is done by computing a finite-difference approximation to the gradient, and substituting the exact gradient with the approximation within a known nonlinear optimization method; see [22]. These methods operate by spending at least $n + 1$ function evaluations at each iteration to take a meaningful step, where n is the total number of variables. This lies in contrast to interpolation-based methods, which utilize prior function evaluations with only

*Department of Industrial Engineering and Management Sciences, Northwestern University. Shi and Xie were supported by the Office of Naval Research grant N00014-14-1-0313 P00003. Xuan was supported by the National Science Foundation grant DMS-1620022. Nocedal was supported by AFOSR grant FA95502110084, and by National Science Foundation grant DMS-1620022.
hjmshi@u.northwestern.edu, ycxie@u.northwestern.edu, qxuan@u.northwestern.edu, j-nocedal@northwestern.edu

one new evaluation at each iteration; see [6, 16]. Therefore, in order for the finite-difference approach to be effective, one must ensure that the quality of the gradient is satisfactory and significant progress is being made at each iteration of the algorithm (as opposed to n steps of the interpolation-based approach).

Often, black-box functions to be optimized are contaminated by stochastic or computational noise. This noise could arise naturally from modeling randomness within a simulation, or as a bi-product of an adaptive computation, for example through the early termination of an iterative solver. The presence of noise has largely prevented finite-difference methods from gaining more popularity within the derivative-free optimization community, as the precise choice of the finite-difference interval becomes increasingly critical as the noise level increases.

In particular, the finite-difference interval requires knowledge of both the noise level and higher-order derivative of the function. While the former may be known *a priori* or can be estimated by sampling or computing difference tables [18], the latter quantity is not normally available to the user. In order to make finite-difference methods a viable alternative in the presence of noise, a robust procedure is needed for estimating higher-order derivatives, either implicitly or explicitly.

To put this more precisely, let us consider the problem of estimating the d-th order derivative of a smooth univariate function $\phi : \mathbb{R} \to \mathbb{R}$. We will assume that we are only provided noisy function evaluations of the form

$$f(t) = \phi(t) + \epsilon(t) \tag{1.1}$$

where $\epsilon : \mathbb{R} \to \mathbb{R}$ models the error, and that the error is bounded, i.e., there exists $\epsilon_f \geq 0$ such that $|\epsilon(t)| \leq \epsilon_f$. We call ϵ_f the *noise level* of the function. We focus on the univariate case, although this can be easily extended to the multivariate setting for computing the gradient by applying the procedure to each component.

The simplest and cheapest finite-difference approximation to the first derivative is the forward-difference approximation. If $\phi^{(d)}$ denotes the d-th order derivative of ϕ, then the forward-difference approximation is computed by

$$\phi^{(1)}(t) \approx \frac{f(t+h) - f(t)}{h} \triangleq f^{(1)}(t; h) \tag{1.2}$$

where $h > 0$ is the finite-difference interval. Note the slight abuse of notation by denoting $f^{(d)}$ as the finite-difference approximation to the d-th order derivative. With no noise, excluding round-off error, one would ideally choose h as small as possible, the common practical choice being $h = \max\{1, |x|\}\sqrt{\epsilon_M}$, where ϵ_M is machine precision, to handle rounding errors. However, this choice of the finite-difference interval may be poor under the presence of large errors, as is well-known.

To see this, consider the following decomposition of the error in the forward-difference approximation:

$$\left| f^{(1)}(t; h) - \phi^{(1)}(t) \right| \leq \left| \frac{\phi(t+h) - \phi(t)}{h} - \phi^{(1)}(t) \right| + \left| \frac{\epsilon(t+h) - \epsilon(t)}{h} \right|. \tag{1.3}$$

We will call the error induced by the first term *truncation error* since it arises from truncation of the Taylor series, and the error induced by the second term *measurement error* due to error in the function evaluations.

Note that if h is small, then the truncation error is small but the measurement error may be large. On the other hand, if h is too large, the measurement error may be small but the truncation error may be too high. Therefore, the optimal h trades off these two terms by making the error from each of these two sources equal. In this paper, we propose an adaptive procedure for estimating the finite-difference interval in the presence of noise that properly balances these two different sources of error. The procedure must be: (1) reliable, that is, applicable to most, if not all, practical problems of interest; (2) accurate, producing near-optimal estimates of the finite-difference interval; and (3) efficient, employing the least number of function evaluations possible. We argue that our procedure achieves these goals in many practical situations.

This paper is organized into five sections. We present the notation and literature review in the rest of this section. In Section 2, we introduce our finite-difference interval estimation procedure for the forward-difference case. In Section 3, we present the generalized procedure for arbitrary finite-difference schemes and provide theoretical guarantees for the termination of our procedure. Extensive numerical results on synthetic problems with injected noise are provided in Section 4, and concluding remarks are made in Section 5.

1.1 Literature Review

The problem of estimating derivatives, particularly in the presence of rounding errors, is a fundamental question within numerical analysis and scientific computing. Fornberg proposed a stable algorithm for generating finite-difference formulas on arbitrarily spaced grids [8]. Lyness and Moler observed that the Cauchy integral theorem allows one to evaluate the d-th derivative of a complex function as a closed complex integral via numerical integration techniques [17]. This was simplified and extended by Squire and Trapp who observed that one could avoid cancellation error by using complex perturbations in the Taylor expansion, called complex step differentiation [23]. This has more recently led to extensions of the complex step to evaluating the Hessian by Hare and Srivastava [13]. Brekelmans, et al. compared design of experiments schemes against standard finite-difference schemes within the stochastic noise regime [4].

To handle rounding errors, Curtis and Reid describe a heuristic that estimates the truncation and rounding errors using central and forward-difference estimates. The ratio between the two estimates of these errors are used to determine the finite-difference interval [7]. Stepleman and Winarsky use a set of decreasing central-difference intervals. The optimal interval is obtained by the smallest interval that does not violate monotonic decrease in the absolute difference between consecutive central-difference estimates [24]. Gill, Murray, Saunders, and Wright introduced an adaptive procedure for computing forward-difference intervals by utilizing a ratio to determine the second derivative [9, 10]. Their procedure has some similarities with our approach, which we discuss in Section 2.1. Barton proposed an adaptive procedure for handling rounding or multiplicative errors by interpreting the function values as correct up to a fixed number of significant digits and ensuring that at

least a certain number of significant digits change from the resulting difference interval [1]. Most recently, Moré and Wild proposed a heuristic for estimating the second derivative for determining the forward-difference interval that checks two conditions: (1) if the noise dominates the second-order derivative; and (2) if the forward and backward difference is too large relative to the function values [19]. A comparison of the resulting errors between finite-difference and simplex gradients were analyzed in [3, 12].

Incorporating finite differences into optimization methods have also had a long history. Kiefer and Wolfowitz first applied finite differences to stochastic approximation [15]. Kelley developed an implicit filtering BFGS method that utilizes finite differences in the case where noise decays as the iterates converge to the solution [5, 14]. Berahas, et al. proposed a finite-difference L-BFGS method that incorporates `ECNoise` and a heuristic for estimating the second derivative by Moré and Wild into L-BFGS [2, 18, 19]. Most recently, Shi, et al. tested finite-difference methods within the unconstrained, least squares, and constrained settings assuming knowledge of the noise level [22].

1.2 Notation

In the following sections, we will use Bachmann-Landau notation liberally. Suppose $f, g : \mathbb{R} \to \mathbb{R}^{\geq 0}$. We will write $g(h) = \mathcal{O}(f(h))$ if there exists a $C \in \mathbb{R}$ such that $g(h) = Cf(h)$. If $g(h) = o(f(h))$, then for every $\epsilon > 0$ there exists a constant N such that $|g(h)| \leq \epsilon |f(h)|$ for all $h \leq N$. Similarly, if $g(h) = O(f(h))$, then there exists constants $\epsilon > 0$ and N such that $|g(h)| \leq \epsilon |f(h)|$ for all $h \leq N$.

We will use $\phi^{(d)} : \mathbb{R} \to \mathbb{R}$ to denote the d-th order derivative of ϕ. For a given vector $x \in \mathbb{R}^n$, $[x]_i$ denotes the i-th component of x. Similarly, for a given matrix $A \in \mathbb{R}^n$, $[A]_{ij}$ denotes the (i, j)-th entry of A. We will use $\| \cdot \|$ to denote the standard Euclidean norm unless otherwise specified.

2 An Adaptive Forward-Difference Interval Estimation Procedure

Suppose we are interested in determining the finite-difference interval for the forward-difference approximation of the first derivative of ϕ. Since the Taylor expansion of the function ϕ is given by

$$\phi(t + h) = \phi(t) + \phi^{(1)}(t)h + \frac{\phi^{(2)}(t)}{2}h^2 + o(h^2),$$

the total error can be bounded by

$$|\phi^{(1)}(t) - f^{(1)}(t; h)| \leq \underbrace{\frac{|\phi^{(2)}(t)|h}{2}}_{T_1} + \underbrace{\frac{2\epsilon_f}{h}}_{T_2} + o(h).$$

By ignoring the higher-order term, this yields an optimal interval (with respect to the upper bound) of

$$h^* \approx 2\sqrt{\frac{\epsilon_f}{|\phi^{(2)}(t)|}}. \tag{2.1}$$

This formula requires an estimate of the second derivative $|\phi^{(2)}(t)|$. We now propose a procedure that yields an interval $h = \mathcal{O}\left(\sqrt{\frac{\epsilon_f}{|\phi^{(2)}(t)|}}\right)$ without estimating $|\phi^{(2)}(t)|$ separately.

Our procedure balances the truncation T_1 and measurement error T_2. To do so, it estimates the ratio between these two errors directly and attempts to find an interval h for which this ratio is close to some constant value. We claim that the ratio T_1/T_2 of the truncation over measurement error can be approximated, for example, by the *testing ratio*

$$r(h; f, t, \epsilon_f) = \frac{|f(t+4h) - 4f(t+h) + 3f(t)|}{8\epsilon_f}. \tag{2.2}$$

Given $r_l > 1$ and $r_u > r_l + 2$, we perform a bisection search to find an interval $h > 0$ that satisfies

$$r(h; f, t, \epsilon_f) \in [r_l, r_u]. \tag{2.3}$$

In particular, if $r(h; f, t, \epsilon_f) < r_l$, then the numerator is dominated by noise, indicating that h is too small. On the other hand, if $r(h; f, t, \epsilon_f) > r_u$, then the numerator significantly dominates the noise, which implies that h is too large. Our procedure for forward differences is summarized in Algorithm 1.

Algorithm 1: Adaptive Forward-Difference Interval Estimation

Input: One-dimensional noisy function $f : \mathbb{R} \to \mathbb{R}$; noise level $\epsilon_f > 0$; lower- and upper-bound $(r_l, r_u) = (1.5, 6)$;

Output: finite-difference interval h.

$h \leftarrow \frac{2}{\sqrt{3}}\sqrt{\epsilon_f}$;
$l \leftarrow 0$, $u \leftarrow +\infty$;
while ***True*** **do**
  Evaluate $r(h; f, t, \epsilon_f) = \frac{|f(t+4h)-4f(t+h)+3f(t)|}{8\epsilon_f}$;
  if $r(h; f, t, \epsilon_f) < r_l$ **then**
    $l \leftarrow h$;
  else if $r(h; f, t, \epsilon_f) > r_u$ **then**
    $u \leftarrow h$;
  else
    break;
  if $u = +\infty$ **then**
    $h \leftarrow 4h$;
  else if $l = 0$ **then**
    $h \leftarrow h/4$;
  else
    $h \leftarrow (l + u)/2$;
return h

To see why this procedure works to give us a near-optimal h, note that

$$\phi(t + 4h) - 4\phi(t + h) + 3\phi(t) = 6\phi^{(2)}(t)h^2 + o(h^2). \tag{2.4}$$

Therefore, if we expand (2.2), we obtain:

$$r(h; f, t, \epsilon_f) = \left| \frac{3\phi^{(2)}(t)h^2}{4\epsilon_f} + \frac{\epsilon(t + 4h) - 4\epsilon(t + h) + 3\epsilon(t)}{8\epsilon_f} + o(h^2) \right|. \tag{2.5}$$

Since $\left|\frac{\epsilon(t+4h)-4\epsilon(t+h)+3\epsilon(t)|}{8\epsilon_f}\right| \leq 1$ by the fact that $|\epsilon(t)| \leq \epsilon_f$ for all $t \in \mathbb{R}$, by imposing (2.3) and ignoring the $o(h^2)$ term, we approximately have

$$\frac{3|\phi^{(2)}(t)|h^2}{4\epsilon_f} \in [r_l - 1, r_u + 1] \iff h \in \frac{2}{\sqrt{3}}\sqrt{\frac{\epsilon_f}{|\phi^{(2)}(t)|}} \cdot [\sqrt{r_l - 1}, \sqrt{r_u + 1}]. \tag{2.6}$$

Therefore, if r_l and r_u are chosen properly, such as $r_l = 1.5$ and $r_u = 6$, we obtain $h \in [\sqrt{0.5}, \sqrt{7}] \cdot \sqrt{\frac{\epsilon_f}{|\phi^{(2)}(t)|}}$, which is the same order as the optimal finite-difference interval (2.1), differing only by a small constant factor.

By scaling h by a factor of 4 in Algorithm 1 when $u = \infty$ or $l = 0$, the new trial h only requires a single new function evaluation to check the testing ratio when h is updated monotonically.

In addition, the testing ratio is affine-invariant with respect to the function of interest in the sense that $r(h; f, t, \epsilon_f)$ remains unchanged if applied to a modified function $\tilde{f}(t) = af(t) + b$ for $a \neq 0$ and $b \in \mathbb{R}$ with noise level $|a|\epsilon_f$, i.e. $r(h; \tilde{f}, t, |a|\epsilon_f) = r(h; f, t, \epsilon_f)$. Therefore, the finite-difference interval h will correctly remain unchanged under this transformation.

2.1 Comparison to Prior Methods

Although the definition of the testing ratio appears similar to the ratio in Gill, et al. [9], which is defined as the inverse ratio

$$\frac{4\epsilon_f}{|f(t+\tilde{h}) - 2f(t) + f(t-\tilde{h})|}, \tag{2.7}$$

our approach markedly differs from theirs in three aspects: (1) we utilize the h derived from Algorithm 1 directly as the chosen difference interval, whereas Gill, et al. use the difference interval $\tilde{h}$ to estimate the second derivative; (2) our approach utilizes a bisection search to find h rather than monotonically increasing or decreasing h; and (3) it relies on second-order forward-difference instead of central-difference estimates.

The first aspect follows from the observation made in (2.6) that $h = \mathcal{O}\left(\sqrt{\frac{\epsilon_f}{|\phi^{(2)}(t)|}}\right)$. Hence, the difference interval h obtained by our bisection procedure is near-optimal in that it is only a small constant factor away from the optimal interval (except for certain cases discussed below). Two observations can be made from this derivation. The first is that since $f(t+h)$ and $f(t)$ are used in the evaluation of the testing ratio, one can reuse prior function evaluations from the bisection procedure to estimate $f^{(1)}(t; h)$. The second observation is that the derivation motivates an initial choice of $h = \mathcal{O}(\sqrt{\epsilon_f})$ as opposed to $h = \mathcal{O}(\sqrt[4]{\epsilon_f})$, as used in Moré and Wild [19]. This is corroborated by our experiments in Section 4.

The second aspect follows from different tradeoffs between cost and accuracy in the finite-difference interval estimate. In particular, whereas Gill, et al.'s procedure is cheaper but yields a less accurate estimate, our procedure is able to guarantee a sufficiently accurate estimate at higher cost.

Lastly, the third aspect is designed to avoid cancellation due to symmetry in the numerator of the testing ratio. In particular, Gill, et al. [9] rely on a testing ratio using the central difference:

$$\frac{4\epsilon_f}{|f(t+\tilde{h}) - 2f(t) + f(t-\tilde{h})|} \in [0.001, 0.1]. \tag{2.8}$$

However, we can obtain poor estimates of the derivative due to cancellation under symmetry of the function, for example, on a quartic function $\phi(t) = t^4$ near (but not at) $t = 0$ with sufficiently large noise.

Our procedure also differs from Moré and Wild's procedure [19]. Their procedure estimates the second derivative by

$$\phi^{(2)}(t) \approx \frac{f(t+\tilde{h}) - 2f(t) + f(t-\tilde{h})}{\tilde{h}^2} = f^{(2)}(t; \tilde{h}), \tag{2.9}$$

with interval $\tilde{h} > 0$, then inserts this estimate into the optimal formula (2.6). The difference interval $\tilde{h}$ is required to satisfy

$$|f(t+\tilde{h}) - 2f(t) + f(t-\tilde{h})| \geq \tau_1 \epsilon_f \tag{2.10}$$

$$|f(t \pm \tilde{h}) - f(t)| \leq \tau_2 \max\{|f(t)|, |f(t \pm \tilde{h})|\} \tag{2.11}$$

with $\tau_1 \gg 1$ and $\tau_2 \in (0,1)$. Their method attempts to satisfy this within two trials as follows:

1. Set $\tilde{h}_1 = \sqrt[4]{\epsilon_f}$ and compute $\mu_1 = |f^{(2)}(t;\tilde{h}_1)|$. If conditions (2.10) and (2.11) are satisfied for $\tilde{h}_1$, return μ_1.
2. Set $\tilde{h}_2 = \sqrt[4]{\epsilon_f/\mu_1}$ and compute $\mu_1 = |f^{(2)}(t;\tilde{h}_2)|$. If conditions (2.10) and (2.11) are satisfied for $\tilde{h}_2$, return μ_2.
3. If $|\mu_1 - \mu_2| \leq \frac{1}{2}\mu_2$, return μ_2.

If the heuristic is unable to return an estimate μ after two trials, this is considered as a failure.

Similar to Gill's procedure, this method differs from ours in that it estimates the second derivative rather than utilizing the h derived from the testing ratio directly. As a result, it initializes $\tilde{h} = \mathcal{O}(\sqrt[4]{\epsilon_f})$ rather than $\mathcal{O}(\sqrt{\epsilon_f})$. We have found this to be a poor initial choice of $\tilde{h}$, particularly when ϵ_f is large, as we will see in Section 4.

While (2.10) appears similar to the testing ratio, it is better interpreted as ensuring that noise does not dominate the second-derivative estimation due to the large choice of $\tau_1 = 100$. The second condition (2.11) is not affine-invariant in the sense that adding a sufficiently large b can force the condition to be satisfied. This is undesirable as perturbations of the function should not change the overall behavior of the method.

3 Generalized Finite-Difference Interval Estimation

Typically, finite-difference interval estimation procedures for numerical optimization focus on forward differences [1, 9, 19]. However, in the noisy regime, higher-order finite-difference approximations, such as central differences, can yield more accurate approximations; see [22]. As a result, in order to attain the highest possible accuracy, one must design methods that efficiently find a near-optimal difference interval for more general finite-difference schemes. To handle this, we propose a generalization of the forward-difference case, Algorithm 1, for d-th order derivatives.

Consider a finite-difference approximation scheme $S = (w, s)$ defined over m points, where we approximate $\phi^{(d)}(t)$ using the equation

$$f_S^{(d)}(t;h) = \frac{\sum_{j=1}^m w_j \cdot f(t + h s_j)}{h^d} \approx \phi^{(d)}(t) \tag{3.1}$$

where $w \in \mathbb{R}^m$ and $s \in \mathbb{R}^m$ are the associated weights and shifts of the finite-difference scheme. As in the forward-difference setting, we will use a slight abuse of notation by

denoting the finite-difference approximation as $f_S^{(d)}(t;h)$. *Forward-difference* and *central-difference* schemes for approximating the first derivative (i.e., $d=1$) are obtained by defining s and w as $s=(0,1)^T$ and $w=(-1,1)^T$ and $s=(-1,1)^T$ and $w=(-\frac{1}{2},\frac{1}{2})^T$, respectively. The standard *second-order central-difference* scheme is defined as $s=(-1,0,1)^T$ and $w=(1,-2,1)^T$.

In order for the finite-difference scheme to be valid, the coefficients w and shifts s must be chosen such that the Taylor expansion of the finite-difference approximation over the function ϕ satisfies

$$\sum_{j=1}^{m} w_j \cdot \phi(t+hs_j) = \phi^{(d)}(t)h^d + c_q\phi^{(q)}(t)h^q + o\left(h^q\right), \tag{3.2}$$

where $q \geq d+1$ denotes the order of the remainder term[1]. This ensures that $f_S^{(d)}(t;h) \approx \phi^{(d)}(t)$. In order to guarantee this, the finite-difference scheme S must satisfy

$$\frac{1}{l!}\sum_{j=1}^{m} w_j s_j^l = \begin{cases} 1 & \text{if } l=d \\ 0 & \text{if } l<q,\ l\neq d \end{cases}$$

and as a result

$$c_q = \frac{1}{q!}\sum_{j=1}^{m} w_j s_j^q.$$

See Appendix A for more detail on how generic finite-difference schemes are derived.

Therefore, in the presence of noise, the worst-case error for the finite-difference scheme of interest can be bounded by

$$|f^{(d)}(t;h) - \phi^{(d)}(t)| \leq |c_q|\left|\phi^{(q)}(t)\right| h^{q-d} + \|w\|_1\epsilon_f h^{-d} + o\left(h^{q-d}\right).$$

One can define an approximately optimal choice of h:

$$h^* \approx \left|\frac{d}{q-d}\cdot\frac{\|w\|_1\epsilon_f}{c_q\phi^{(q)}(t)}\right|^{1/q}. \tag{3.3}$$

While ϵ_f is assumed to be known and d, q, w and c_q are available, the q-th order derivative $\phi^{(q)}(x)$ is unknown and often difficult to estimate. Following the idea from the forward-difference case, we propose a procedure for estimating (3.3) directly. We first construct a testing ratio r_S associated with scheme S:

$$r_S(h;f,t,\epsilon_f) = \frac{\left|\sum_{j=1}^{\tilde{m}} \tilde{w}_j \cdot f(t+h\tilde{s}_j)\right|}{\epsilon_f}$$

[1]Note that the order of accuracy can be higher than $d+1$ for certain schemes, such as central-difference approximations.

where $\tilde{w}, \tilde{s} \in \mathbb{R}^{\tilde{m}}$ where $\tilde{m} \geq q+1$, $\tilde{s}_j \neq \tilde{s}_k$ for all $j \neq k$, and $\tilde{w}$ and $\tilde{s}$ satisfies

$$\sum_{j=1}^{\tilde{m}} \tilde{w}_j \cdot \phi(t + h\tilde{s}_j) = c_r \phi^{(q)}(t) h^q + o\left(h^q\right), \quad c_r = \frac{1}{q!} \sum_{j=1}^{\tilde{m}} \tilde{w}_j \tilde{s}_j^q \neq 0. \tag{3.4}$$

Without loss of generality, we require $\tilde{w}$ to satisfy

$$\|\tilde{w}\|_1 = 1,$$

the reason for which will be evident below. We then perform a bisection search to find an interval h that satisfies

$$r_S(h; f, t, \epsilon_f) \in [r_l, r_u] \tag{3.5}$$

for some $r_l > 1$ and $r_u > r_l + 2$. The procedure is summarized in Algorithm 2.

Algorithm 2: Adaptive Finite-Difference Interval Estimation

Input: One-dimensional noisy function $f : \mathbb{R} \to \mathbb{R}$; noise level $\epsilon_f > 0$; testing ratio $r_S(h; f, t, \epsilon_f)$ for scheme S; lower- and upper-bound r_l and r_u satisfying $1 < r_l < r_u - 2$; initial interval h_0; scaling factor η

Output: finite-difference interval h.

```
h ← h_0;
l ← 0, u ← +∞;
while True do
    Evaluate r_S(h; f, t, ε_f);
    if r_S(h; f, t, ε_f) < r_l then
        l ← h;
    else if r_S(h; f, t, ε_f) > r_u then
        u ← h;
    else
        break;
    if u = +∞ then
        h ← ηh;
    else if l = 0 then
        h ← h/η;
    else
        h ← (l + u)/2;
return h
```

By (3.4), we can see that

$$r_S(h; f, t, \epsilon_f) = \left| \frac{c_r \phi^{(q)}(t) h^q}{\epsilon_f} + \frac{\sum_{j=1}^{\tilde{m}} \tilde{w}_j \cdot \epsilon(t + h\tilde{s}_j)}{\epsilon_f} + o\left(h^q\right) \right| \tag{3.6}$$

$$= \left| \frac{c_r \phi^{(q)}(t) h^q}{\epsilon_f} + \Delta + o\left(h^q\right) \right|, \quad \Delta \triangleq \frac{\sum_{j=1}^{\tilde{m}} \tilde{w}_j \cdot \epsilon(t + h\tilde{s}_j)}{\epsilon_f}. \tag{3.7}$$

Note that by definition of Δ, $|\Delta| \leq 1$. This is a consequence of the requirement that $\|\tilde{w}\|_1 = 1$ and that $|\epsilon(t)| \leq \epsilon_f$ for all $t \in \mathbb{R}$. Therefore, if we have $r_S(h; f, t, \epsilon_f) \in [r_l, r_u]$ and if we the ignore $o(h^q)$ term, then we (approximately) have

$$\left| \frac{c_r \phi^{(q)}(t) h^q}{\epsilon_f} \right| \in [r_l - 1, r_u + 1], \tag{3.8}$$

i.e.,

$$h \in \left[\left(\frac{r_l - 1}{|c_r|} \frac{\epsilon_f}{\left|\phi^{(q)}(t)\right|} \right)^{1/q}, \left(\frac{r_u + 1}{|c_r|} \frac{\epsilon_f}{\left|\phi^{(q)}(t)\right|} \right)^{1/q} \right]. \tag{3.9}$$

Note from (3.3) that h has the same dependence on ϵ_f and $\phi^{(q)}(t)$ as h^*. As in the forward-difference case, our algorithm is invariant to affine transformations with respect to the function.

Example 1 (First-Order Central Difference). Consider the first-order central-difference scheme for approximating the first derivative:

$$f_S^{(1)}(t; h) = \frac{f(t + h) - f(t - h)}{2h}, \tag{3.10}$$

where $s = (-1, 1)^T$ and $w = (-\frac{1}{2}, \frac{1}{2})^T$. The Taylor expansion of the numerator is given as:

$$\frac{\phi(t + h) - \phi(t - h)}{2h} = \phi^{(1)}(t) + \frac{\phi^{(3)}(t) h^2}{6} + o(h^2).$$

The full error of the derivative approximation and the approximate optimal choice of h are:

$$\left| f_S^{(1)}(t; h) - \phi^{(1)}(t) \right| \leq \frac{\left|\phi^{(3)}(t)\right| h^2}{6} + \frac{\epsilon_f}{h} + o(h^2), \qquad h^* \approx \sqrt[3]{\frac{3\epsilon_f}{\left|\phi^{(3)}(t)\right|}}.$$

One example of a valid testing ratio is:

$$r_S(h; f, t, \epsilon_f) = \frac{|f(t + 3h) - 3f(t + h) + 3f(t - h) - f(t - 3h)|}{8\epsilon_f}. \tag{3.11}$$

Example 2 (Second-Order Central Difference). Consider the second-order central-difference scheme for approximating the second derivative:

$$f_S^{(2)}(t; h) = \frac{f(t + h) - 2f(t) + f(t - h)}{h^2}, \tag{3.12}$$

where $s = (-1, 0, 1)^T$ and $w = (-\frac{1}{2}, \frac{1}{2})^T$. The Taylor expansion of the numerator is given as:

$$\frac{\phi(t + h) - 2\phi(t) + \phi(t - h)}{h^2} = \phi^{(2)}(t) + \frac{\phi^{(4)}(t) h^2}{24} + o(h^2).$$

The full error of the derivative approximation and the approximate optimal choice of h are:

$$\left| f_S^{(2)}(t;h) - \phi^{(2)}(t) \right| \leq \frac{\left|\phi^{(4)}(t)\right| h^2}{24} + \frac{4\epsilon_f}{h^2} + o(h^2), \qquad h^* \approx 2\sqrt[4]{\frac{6\epsilon_f}{\left|\phi^{(4)}(t)\right|}}.$$

One example of a valid testing ratio is:

$$r_S(h; f, t, \epsilon_f) = \frac{|f(t+2h) - 4f(t+h) + 6f(t) - 4f(t-h) + f(t-2h)|}{16\epsilon_f}. \tag{3.13}$$

3.1 Practical Considerations

In order to make the procedure both efficient and robust, we discuss a number of practical considerations below.

I. Generation of Testing Ratio. Although many choices of r_S are possible for any finite-difference scheme S, it would be useful to have a method for automatically generating valid testing ratios that efficiently utilize function values. A simple yet useful way to construct r_S is through the formula

$$r_S^{\alpha}(h; f_S, t, \epsilon_f) = \frac{\left| \left(f_S^{(d)}(t;h) - \alpha^{-d} f_S^{(d)}(t;\alpha h) \right) h^d \right|}{A\epsilon_f}, \tag{3.14}$$

where $\alpha \neq 1$ and A is computed by normalizing the coefficients such that $\|\tilde{w}\|_1 = 1$ is satisfied.

This approach is guaranteed to generate a valid testing ratio r_S for any $\alpha \neq 1$ since it cancels out the $\phi^{(d)}(t)$ term in the Taylor expansion, leaving only the relevant higher-order term of order q of interest. In particular, since

$$\begin{aligned}
\sum_{j=1}^{m} w_j \cdot \phi(t + hs_j) &= \phi^{(d)}(t)h^d + c_q\phi^{(q)}(t)h^q + o(h^q) \\
\sum_{j=1}^{m} w_j \cdot \phi(t + \alpha hs_j) &= \phi^{(d)}(t)(\alpha h)^d + c_q\phi^{(q)}(t)(\alpha h)^q + o(h^q),
\end{aligned}$$

we obtain that

$$\begin{aligned}
\left(f_S^{(d)}(t;h) - \alpha^{-d} f_S^{(d)}(t;\alpha h) \right) h^d &= \sum_{j=1}^{m} \left(w_j \cdot \phi(t + hs_j) - \alpha^{-d} w_j \cdot \phi(t + \alpha hs_j) \right) \\
&= c_q(1 - \alpha^{q-d})\phi^{(q)}(t)h^q + o(h^q),
\end{aligned}$$

which satisfies (3.4) with an effective $c_r = c_q(1 - \alpha^{q-d})/A$ as desired.

For finite-difference schemes with equidistant points, a small modification to the bisection search allows us to reuse $1 - 2$ function evaluations at each iteration, depending on the original scheme S. To do this, we can multiply or divide by the same factor $\eta = \alpha$ when

monotonically increasing or decreasing h within the bisection search, as done with $\eta = 4$ in the forward-difference algorithm (Algorithm 1).

In addition, with this choice of the testing ratio, the function evaluations needed within the finite-difference scheme is implicitly evaluated within the testing ratio. We can therefore obtain the finite-difference approximation using previously computed function values at no additional cost.

II. Choice of r_l and r_u. Ideally, one should choose r_l and r_u such that they are close to the optimal ratio

$$r^* = \frac{d}{q-d} \cdot \left|\frac{c_r}{c_q}\right| \cdot \|w\|_1$$

in order to yield an h that is close to h^* in (3.3). However, this is not directly possible in the presence of noise, which requires that $1 < r_l < r_u - 2$ in order to ensure finite-termination; see Section 3.2. We therefore select (r_l, r_u) sufficiently large such that $1 < r_l < r_u - 2$ and, if possible, such that r^* is logarithmically centered within the interval $[r_l, r_u]$:

$$r_l = \max\left\{1 + \eta, \frac{r^*}{\beta}\right\}, \qquad r_u = \max\left\{3(1+\eta), \beta r^*\right\} \tag{3.15}$$

for some $\eta > 0$ and $\beta > 1$. (In our experiments, we set $\eta = 0.1$ and $\beta = 2$.)

Note that when $r_l, r_u > r^*$, the algorithm may overestimate $|\phi^{(q)}(t)|$ and hence underestimate h. In order to avoid this in practice, we have found that it is preferable to choose a testing ratio such that the optimal ratio $r^* \geq \beta(1+\eta)$. This could be done by choosing a different testing ratio, such as by choosing a larger α in (3.14).

III. Initialization of h_0. Since the difference interval h that satisfies the procedure is approximately of the form (3.9), it is preferable to initialize $h_0 = \mathcal{O}(\epsilon_f^{1/q})$. Two possible choices are $h_0 = \epsilon_f^{1/q}$ or $\left(\frac{d}{q-d} \cdot \frac{\|w\|_1}{|c_q|} \cdot \epsilon_f\right)^{1/q}$. The latter is based on the assumption that $|\phi^{(q)}(t)| \approx 1$. If instead the finite-difference interval is re-estimated within an optimization algorithm, we can initialize h_0 as the difference interval h used at the prior outer iteration of the algorithm.

We observe that on rare occasions a poor initial choice of h_0 can result in large error in the derivative approximation. This occurs when the initial choice of h_0 is too large to capture the local behavior of the function. Reducing the initial interval h_0 resolves this issue.

IV. Handling of Special Cases. The Taylor expansion analysis elucidates two possible failure cases for our procedure. In particular, observe that

$$r_S(h; f, t, \epsilon_f) = \left|\frac{c_r \phi^{(q)}(t) h^q}{\epsilon_f} + \Delta + o(h^q)\right|.$$

If h is large (for example, when the noise level ϵ_f is high), the higher-order terms $o(h^q)$ can dominate the other terms in the Taylor series expansion. This can yield poor estimates of

h even if the condition $r_S(h; f, t, \epsilon_f) \in [r_l, r_u]$ is satisfied. In practice, we have not found this to be a common issue.

The more common case is when $\phi^{(q)}(t) \approx 0$. In this case, r_S will be dominated by Δ. In this case, $r_S(h; f, t, \epsilon_f) < r_l$ for all h and h will thus monotonically increase until the maximum number of iterations is reached (which we set `max_iter` to 20). This occurs, for example, with any $(q-1)$-th degree polynomial. In this case, the method provides a warning but does not flag this as a failure. Note that in this case, h is a good choice because sending $h^* \to \infty$ would allow for indefinite reduction in the noise.

V. Extension to Standard Deviation. In some settings, we only have access to the standard deviation of the noise, where $\epsilon(x)$ is modeled as a random variable. Assuming $\mathbb{E}[\epsilon(x)] = 0$, one can extend this procedure to the stochastic setting by replacing ϵ_f with $\sigma_f = \sqrt{\mathbb{E}[\epsilon(x)^2]}$. While finite termination (see next subsection) is not guaranteed, if the procedure succeeds, it will yield an h that has the same dependence on σ_f and $|\phi^{(2)}(t)|$ as the optimal finite-difference interval with respect to its mean-squared error.

VI. Error Bound Estimate on Gradient. Using the h we obtain from our procedure, we can approximate the error bound. Ignoring the $o(h^{q-d})$ term, the error is approximately given by:

$$\epsilon_g(t; h) \approx |c_q| \left|\phi^{(q)}(t)\right| h^{q-d} + \|w\|_1 \epsilon_f h^{-d}.$$

As we have shown in (3.8), we can bound $\left|\phi^{(q)}(t)\right|$ by

$$\left|\phi^{(q)}(t)\right| \leq \frac{\epsilon_f(r_u + 1)}{|c_r|\, h^q}.$$

Therefore, we can approximately bound the error by

$$\epsilon_g(t; h) \lesssim \left(\frac{|c_q|}{|c_r|}(r_u + 1) + \|w\|_1\right) \epsilon_f h^{-d}.$$

In practice, we have found that this error bound is able to obtain an order-of-magnitude of the actual error, but may underestimate the error due to the $o(h^{q-d})$ term.

3.2 Finite Termination

Next, we prove a finite termination theorem for Algorithm 2. We start by making the following assumptions:

Assumption 3.1. *The testing ratio r_S satisfies:*

$$|r_S(h; \phi, t, \epsilon_f) - r_S(h; f, t, \epsilon_f)| \leq 1, \quad \forall t \in \mathbb{R},\ h > 0$$

This assumption is satisfied by our requirement that $\|\tilde{w}\|_1 = 1$ and that $|\epsilon(t)| \leq \epsilon_f$. Note that this can be easily satisfied by simply rescaling the numerator to ensure that the total noise accumulated in the numerator is bounded by ϵ_f.

Assumption 3.2. *As a function of h, $r_S(h;\phi,t,\epsilon_f)$ is continuous with $r_S(0;\phi,t,\epsilon_f)=0$, and there exists an integer $K\in\mathbb{N}$ such that*

$$r_S(2^K h_0;\phi,t,\epsilon_f)\geq r_u-1$$

Assuming that $\epsilon_f>0$, the requirement that $r_S(0;\phi,t,\epsilon_f)=0$ is automatically satisfied by validity of the testing ratio (3.4). The second part of Assumption 3.2, while technical, is satisfied, for example, when $\left|\phi^{(q)}(\xi)\right|\geq\eta>0$ for all $\xi\in[\min_j\{t+h\tilde{s}_j\},\max_j\{t+h\tilde{s}_j\}]$. With these assumptions, we can now show finite termination.

Theorem 3.3. *Suppose Assumptions 3.1 and 3.2 are satisfied. In addition, suppose r_u and r_l are chosen such that $0<r_l<r_u-2$. Then, Algorithm 2 will terminate successfully in a finite number of iterations.*

Proof. We assume that $h\geq 0$. Assume by contradiction that Algorithm 2 does not terminate finitely. We denote the variables l,u,h used *at the beginning of* the k-th iteration of Algorithm 2 as l_k,u_k,h_k, respectively. Obviously, we have

$$0\leq l_k\leq h_k\leq u_k,\ \forall k\in\mathbb{N},$$

and

$$l_k\leq l_{k+1}<u_{k+1}\leq u_k,\ \forall k\in\mathbb{N}.$$

First, we show that $r_S(l_k;\phi,t,\epsilon_f)<r_l+1$ for all $k\in\mathbb{N}$, by induction on k. Clearly this is true for $k=0$ since $l_0=0$, and we have $r_S(0;\phi,t,\epsilon_f)=0$ by Assumption 3.2. Suppose the statement holds for $k\leq K$. We have two cases: (1) $r_S(h_K;f,t,\epsilon_f)<r_l$, which by Assumption 3.1 implies $r_S(h_K;\phi,t,\epsilon_f)\leq r_S(h_K;f,t,\epsilon_f)+1<r_l+1$. In this case $l_{K+1}=h_K$, so $r_S(l_{K+1};\phi,t,\epsilon_f)=r_S(h_K;\phi,t,\epsilon_f)<r_l+1$. (2) $r_S(h_K;f,t,\epsilon_f)>r_u$, in which case $l_{K+1}=l_K$ so by the induction hypothesis $r_S(l_{K+1};\phi,t,\epsilon_f)=r_S(l_K;\phi,t,\epsilon_f)<r_l+1$. Therefore the induction hypothesis holds for $(K+1)$-th iteration.

By a similar argument, we can show that either $u_k=+\infty$, or $u_k<+\infty$ and $r_S(u_k;\phi,t,\epsilon_f)>r_u-1$ for all $k\in\mathbb{N}$.

In summary, we can show that for all $k\in\mathbb{N}$, we have

$$\text{either}\quad r_S(l_k;\phi,t,\epsilon_f)<r_l+1<r_u-1<r_S(u_k;\phi,t,\epsilon_f), \tag{3.16}$$

$$\text{or}\quad r_S(l_k;\phi,t,\epsilon_f)<r_l+1\ \text{ and }\ u_k=+\infty. \tag{3.17}$$

Next, we claim that there exists $K_1\in\mathbb{N}$ such that $u_k<+\infty$ for $k\geq K_1$. Suppose this is not the case, then we have $r_S(h_k;f,t,\epsilon_f)<r_l,\ \forall k\in\mathbb{N}$. In this case, we have $h_{k+1}=2l_{k+1}=2h_k$, so $h_k=2^k h_0$ for all $k\in\mathbb{N}$. By Assumption 3.2, there exists $K\in\mathbb{N}$ such that $r_S(h_K;\phi,t,\epsilon_f)\geq r_u-1$, and since $r_S(h_K;f,t,\epsilon_f)\geq r_S(h_K;\phi,t,\epsilon_f)-1$, we have $r_S(h_K;f,t,\epsilon_f)\geq r_u-2>r_l$, contradicting the inequality $r_S(h_k;f,t,\epsilon_f)<r_l,\ \forall k\in\mathbb{N}$. This proves the existence of K_1.

We are now ready to present the contradiction. For $k\geq K_1$, since $u_k<\infty$, we have

$$u_{k+1}-l_{k+1}=\frac{1}{2}(u_k-l_k)$$

This implies that $u_k - l_k \to 0$. Since $r_S(h;\phi,t,\epsilon_f)$ (as a function of h) is continuous and $u_{K_1} < +\infty$, $[0, u_{K_1}]$ is compact so $r_S(h;\phi,t,\epsilon_f)$ (as a function of h) is *uniformly continuous* on $[0, u_{K_1}]$. Note that $l_k, u_k \in [0, u_{K_1}]$ for $k \geq K_1$, therefore we have

$$r_S(u_k;\phi,t,\epsilon_f) - r_S(l_k;\phi,t,\epsilon_f) \to 0$$

This contradicts the fact that

$$r_S(l_k;\phi,t,\epsilon_f) < r_l + 1 < r_u - 1 < r_S(u_k;\phi,t,\epsilon_f),\ \forall k \in \mathbb{N},\ k \geq K_1$$

Therefore, Algorithm 2 must terminate finitely. Clearly, whenever it terminates, the output h_R must satisfy

$$r_S(h_R; f, t, \epsilon_f) \in [r_l, r_u].$$

□

4 Numerical Experiments

In this section, we present numerical results demonstrating the reliability of our finite-difference interval estimation procedure. We first utilize the method for computing first and second derivatives of commonly tested functions, with added noise. We then insert our procedure into a standard L-BFGS implementation and demonstrate its usefulness on a subset of synthetic noisy `CUTEst` problems [11]. All methods were implemented in Python 3.

4.1 Finite-Difference Interval Estimation

We first test our proposed procedure on several univariate functions. We focus on the case where $d = 1$ and 2 as this is most relevant to optimization. We test Algorithm 2 using 6 different estimating schemes, shown in Table 4.1. The testing ratios are generated using formula (3.14) with different choices of α. The α for each scheme is chosen as the smallest integer such that $r^* > \beta = 2$.

label	d	s	w	q	α	r^*	Comment
`FD`	1	$(0,1)$	$(-1,1)$	2	4	3	forward difference
`CD`	1	$(-1,1)$	$(-1/2,1/2)$	3	3	3	central difference
`FD_3P`	1	$(0,1,2)$	$(-3/2,2,-1/2)$	3	3	3.69	forward difference w/ 3 points
`FD_4P`	1	$(0,1,2,3)$	$(-11/6,3,-3/2,1/1)$	4	3	8.25	forward difference w/ 4 points
`CD_4P`	1	$(-2,-1,1,2)$	$(1/12,-2/3,2/3,-1/12)$	5	2	2.5	central difference w/ 4 points
`L2_CD`	2	$(-1,0,1)$	$(1,-2,1)$	4	2	3	2nd-order central difference

Table 4.1: Schemes for approximating the d-th order derivative used in the experiments. The scheme is defined by $S = (w, s)$ as in (3.1); q is defined in (3.2).

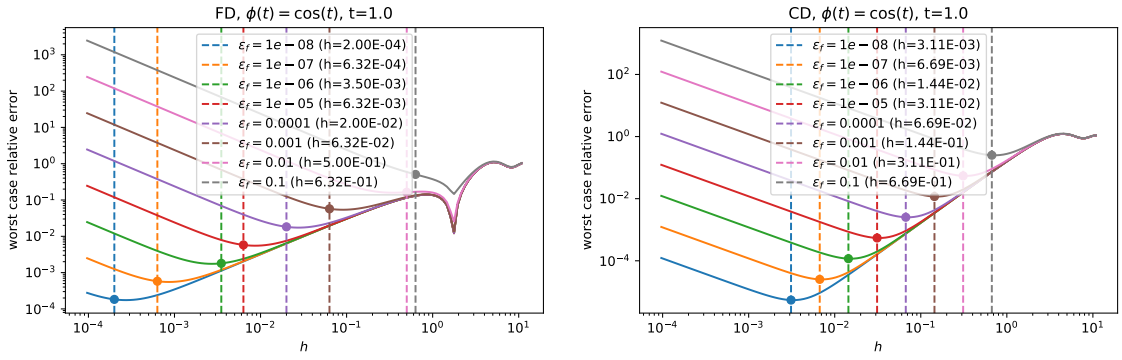

Figure 4.1: Worst case relative error $\delta_S(h; \phi, t, \epsilon_f)$ for forward and central differences against h on function $\phi(t) = \cos(t)$ with different noise levels; the vertical dashed line represents the $h_\dagger$ output by Algorithm 2.

For a specific testing function ϕ at point t with noise ϵ_f and scheme $S = (w, s)$, we plot the *worst case relative error*, as a function of differencing interval h, defined as:

$$\delta_S(h; \phi, t, \epsilon_f) = \frac{1}{\left|\phi^{(d)}(t)\right|} \left[\left| \frac{\sum_{j=1}^{p} w_j \phi(t + s_j h)}{h^d} - \phi^{(d)}(t) \right| + \|w\|_1 \frac{\epsilon_f}{h^d} \right].$$

This function captures the relative error of the estimation scheme S on the noisy function f at t, in the worst case. The differencing interval h that minimizes $\delta_S(h; \phi, t, \epsilon_f)$ is the optimal h. Notice that $\delta_S(h; \phi, t, \epsilon_f)$ is a deterministic function that does not rely on the realization of actual noise in $f(t)$.

We manually inject uniformly distributed, stochastic noise into ϕ,

$$f(t) = \phi(t) + \epsilon(t), \quad \epsilon(t) \sim \text{Uniform}(-\epsilon_f, \epsilon_f),$$

independent of all other quantities. We then apply Algorithm 2 to obtain $h_\dagger$. We plot $h_\dagger$ and observe how far it is from the minimizer of $\delta_S(h; \phi, t, \epsilon_f)$. In Appendix B, we report the minimizer of the function $\delta_S(h; \phi, t, \epsilon_f)$ obtained by `scipy.optimize.minimize_scalar`.

4.1.1 Robustness to Different Noise Levels

To demonstrate that our method is reliable across a range of noise levels, we test our adaptive procedure for both forward and central differences (`FD, CD`) on the simple function $\phi(t) = \cos(t)$ for different noise levels. The plot of the worst case relative error and obtained interval $h_\dagger$ are illustrated in Figure 4.1. Our results demonstrate that our method performs consistently well across a range of noise levels. For all figures on all finite-difference schemes listed in Table 4.1 and complete numerical results, see Appendix B.

4.1.2 Difficult and Special Examples

In this subsection, we consider examples of difficult functions given in [9] and [22]. These examples include:

1. $\phi(t) = \left(e^t - 1\right)^2$, at $t = -8$. This function has extremely small first and second-order derivative at $t = -8$, but quickly increases as t increases beyond $t = 0$; a naive choice of $h = \sqrt{\epsilon_f / |\phi^{(2)}(t)|}$ for forward differences can result in an extremely large h and lead to huge error.

2. $\phi(t) = e^{100t}$, at $t = 0.01$.

3. $\phi(t) = t^4 + 3t^2 - 10t$, at $t = 0.99999$. This function is considered difficult because $\phi'(1) = 0$, and represents a case where the estimated derivative is very close to 0. In addition, this function is a fourth-order polynomial, so the optimal h for `CD_4P` is $+\infty$.

4. $\phi(t) = 10000t^3 + 0.01t^2 + 5t$, at $t = 10^{-9}$. This example is difficult in that it has approximate central symmetry at $t = 0$, which can lead to issues for adaptive procedures such as those proposed in [9].

For each example, we again fix $\epsilon_f = 10^{-3}$, and perform our estimation procedure for different schemes, and plot the worst case relative error. The results can be found in Figure 4.2.

For the first two examples, we see that our procedure is able to estimate the derivative well even when the function increases rapidly. These are cases where using our adaptive procedure is significantly more effective than computing an interval based on higher-order derivative information at the point of interest, as observed in [22].

It is also interesting to observe the results for the two polynomials. For $\phi(t) = t^4 + 3t^2 - 10t$ and scheme `CD_4P`, our procedure generates a large $h_\dagger$; this is consistent with the fact that scheme `CD_4P` has $q = 5$, and $\phi^{(5)}(\xi) = 0$ for all ξ on this example, which implies that we should choose h to be as large as possible. This similarly holds true for the schemes `FD_4P` and `CD_4P` on the function $\phi(t) = 10000t^3 + 0.01t^2 + 5t$.

While theoretically speaking we should choose $h = \infty$ in such cases, we can observe in Figure 4.2 that this is not the case. When plotting the worst-case relative error, we see that there exists a large h such that $\delta_S(h; \phi, t, \epsilon_f)$ is minimized, beyond which the relative error begins to sharply increase. This phenomenon is due to round-off error. When h becomes too large, round-off error (which is multiplicative) will dominate ϵ_f; this approximately happens when $\max_j |\phi(t + s_j h)| \, \epsilon_M$ becomes comparable to ϵ_f.

4.2 Finite-Difference L-BFGS

In order to show the robustness of our procedure, we apply it within the L-BFGS method. We now let ϕ denote a smooth multivariate function, $\phi : \mathbb{R}^n \to \mathbb{R}$, and consider the problem

$$\min_{x \in \mathbb{R}^n} \phi(x), \tag{4.1}$$

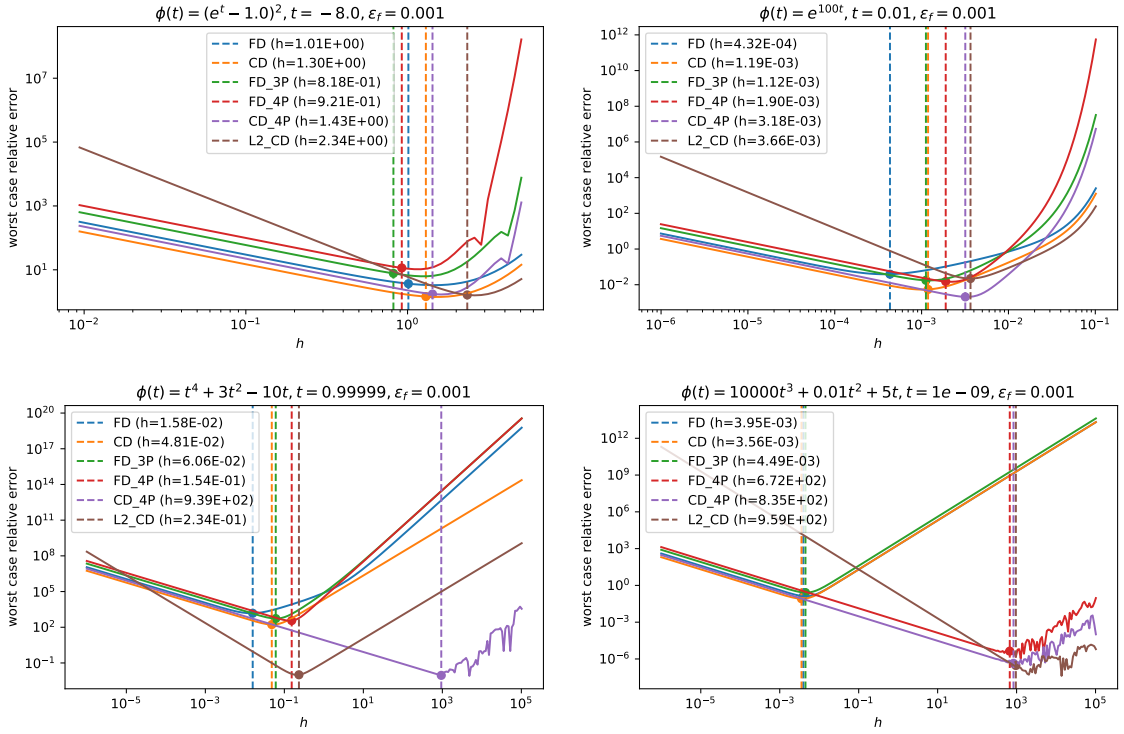

Figure 4.2: Worst case relative error $\delta_S(h; \phi, t, \epsilon_f)$ against h on several special cases; the vertical dashed line represents the $h_\dagger$ output by Algorithm 2.

while only provided noisy function evaluations of the form

$$f(x) = \phi(x) + \epsilon(x), \qquad \epsilon(x) \sim \text{Uniform}(-\epsilon_f, \epsilon_f), \tag{4.2}$$

where $\epsilon_f \in \{10^{-1}, 10^{-3}, 10^{-5}, 10^{-7}\}$. We perform our tests on a subset of synthetically generated noisy `CUTEst` problems [11] detailed in Table 4.2.

Problem	Dim (n)	Problem	Dim (n)	Problem	Dim (n)	Problem	Dim (n)
AIRCRFTB	5	CRAGGLVY	100	FREUROTH	100	PFIT4LS	3
ALLINITU	4	CUBE	2	GENROSE	100	QUARTC	100
ARWHEAD	100	DENSCHND	3	GULF	3	SINEVAL	2
BARD	3	DENSCHNE	3	HAIRY	2	SINQUAD	100
BDQRTIC	100	DIXMAANH	90	HELIX	3	SISSER	2
BIGGS3	3	DQRTIC	100	NCB20B	100	SPARSQUR	100
BIGGS5	5	EDENSCH	36	NONDIA	100	TOINTGSS	100
BIGGS6	6	EIGENALS	110	NONDQUAR	100	TQUARTIC	100
BOX2	2	EIGENBLS	110	OSBORNEA	5	TRIDIA	100
BOX3	3	EIGENCLS	30	OSBORNEB	11	WATSON	31
BRKMCC	2	ENGVAL1	100	PENALTY1	100	WOODS	100
BROWNAL	100	EXPFIT	2	PFIT1LS	3	ZANGWIL2	2
BROWNDEN	4	FLETCBV3	100	PFIT2LS	3		
CLIFF	2	FLETCHBV	100	PFIT3LS	3		

Table 4.2: Subset of unconstrained `CUTEst` problems and their problem dimensions [11].

The L-BFGS method has the form

$$x_{k+1} = x_k - \alpha_k H_k g(x_k), \tag{4.3}$$

where $g(x_k)$ is a finite-difference approximation to the gradient, H_k is the L-BFGS matrix with memory of 10 (see [20]), and α_k is a steplength selected by a relaxed Armijo-Wolfe line search designed to handle noise.

To describe the line search, let α_k^j denote the jth trial steplength at iteration k. Similar to Shi et al.[21], the Armijo condition is relaxed as follows:

$$f(x_k + \alpha_k^j p_k) \begin{cases} \le f(x_k) + c_1 \alpha_k^j g(x_k)^T p_k & \text{if } j = 0,\ g(x_k)^T p_k < -\epsilon_g(x_k)\|p_k\| \\ \le f(x_k) + c_1 \alpha_k^j g(x_k)^T p_k + 2\epsilon_f & \text{if } j \ge 1,\ g(x_k)^T p_k < -\epsilon_g(x_k)\|p_k\| \\ < f(x_k) & \text{if } g(x_k)^T p_k \ge -\epsilon_g(x_k)\|p_k\|, \end{cases} \tag{4.4}$$

where $c_1 = 10^{-4}$, $c_2 = 0.9$, and $\epsilon_g(x_k)$ is the estimated gradient error described below. Thus, we relax the line search only when the gradient is reliable; otherwise, we enforce simple decrease. We check the Wolfe condition by evaluating the directional derivative $\nabla\phi(x)^T p$ using finite differences along the direction p, as in [21].

4.2.1 Forward Differences

In the first set of experiments, the gradient approximation $g(x_k)$ is obtained by forward differences,

$$[g(x_k)]_i = \frac{f(x + h_i e_i) - f(x)}{h_i}, \quad i = 1, \dots n,$$

where the differencing interval h_i is determined by one of the following three strategies.

1. `Fixed`. The interval h is fixed across all components i and for the entire iteration. This strategy tries to emulate the common practice of hand-tuning h_i at the start using problem specific information. We simulate this using the formula

$$h = 2\sqrt{\frac{\epsilon_f}{L_2}}, \qquad \text{where } L_2 = \max\left\{10^{-1}, \sqrt{\frac{\sum_{i=1}^{n}[\nabla^2\phi(x_0)]_{ii}^2}{n}}\right\}, \tag{4.5}$$

which assumes that the diagonals of the Hessian are known. The gradient error is approximated assuming L_2 is correct, that is, $\epsilon_g(x) = 2\sqrt{nL_2\epsilon_f}$. We created this option for benchmarking purposes only.

2. `MW`. The Moré-Wild heuristic for estimating and interval h_i for every component i of the gradient. We set $L_2 = \max\{10^{-1}, \hat{L}_2\}$, where $\hat{L}_2$ is the estimate given by the Moré and Wild heuristic. If the heuristic fails, we set $L_2 = 10^{-1}$. The gradient error is estimated similar to `Fixed` but componentwise, i.e., $\epsilon_g(x) = 2\sqrt{\epsilon_f \sum_{i=1}^{n} L_{2,i}}$.

3. `Adaptive` Our adaptive procedure for estimating h_i along each component using Algorithm 1.

For the `MW` and `Adaptive` strategies, we re-estimate the second derivative or finite-difference interval whenever a partial derivative needs to be approximated. For example, when computing the full gradient, we estimate the finite-difference interval along each coordinate direction separately. We chose not to compare against Gill et al. [9] as we regard the Moré-Wild heuristic to be an improvement over their approach.

We present results for a few representative problems in Figure 4.3. The L-BFGS method described above is terminated if no further progress is made on the objective function over 5 consecutive iterations. Figure 4.3 plots the optimality gap $\phi(x_k) - \phi^*$ against the number of function evaluations. The optimal value ϕ^* is obtained by solving the original problem to completion without noise with L-BFGS. While we found Moré and Wild's heuristic to work well for $\epsilon_f < 10^{-1}$, their heuristic fails frequently for the case where $\epsilon_f = 10^{-1}$. (This can be seen in the complete results presented in Appendix B.) For this reason, we report results for $\epsilon_f = 10^{-1}$ and 10^{-5} to demonstrate the robustness of our algorithm compared to Moré and Wild for different noise levels. When Moré and Wild's heuristic succeeds, we observe that our algorithm (`Adaptive`) is able to more efficiently achieve comparable accuracy to the Moré and Wild heuristic, while attaining more accurate solutions than using a fixed interval for some problems. The lack of accuracy in the `Fixed` strategy can be explained by inability for a fixed interval to adapt to changes in the Hessian over the course of the iteration — an exception being the TRIDIA problem, which is very well scaled.

4.2.2 Central Differences

In the second set of experiments, we employ central differences,

$$[g(x;h)]_i = \frac{f(x + h_i e_i) - f(x - h_i e_i)}{2h_i}.$$

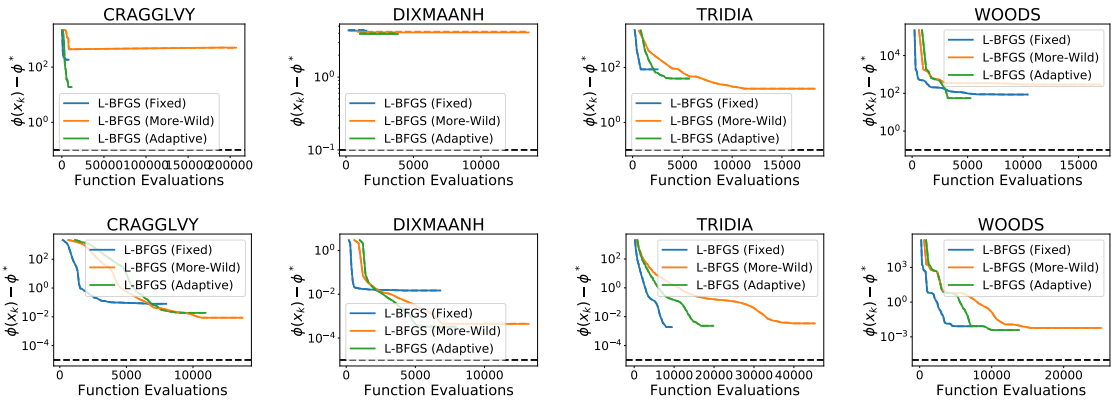

Figure 4.3: Comparison of forward-difference L-BFGS methods with difference intervals determined using a fixed interval, the Moré and Wild heuristic, and our adaptive algorithm. Comparisons are made on representative problems with noise level $\epsilon_f = 10^{-1}$ (top) and 10^{-5} (bottom). The solid line plots the observed function value and the dashed line plots the true function value. The dashed black line shows the noise level ϵ_f of the function.

The differencing interval is determined via a `Fixed` strategy or the `Adaptive` procedure described in Algorithm 2. (The Moré and Wild's heuristic does not apply to this case.) For the `Fixed` strategy, we choose

$$h = \sqrt[3]{\frac{3\epsilon_f}{L_3}}, \qquad \text{where } L_3 = \max\left\{10^{-1}, \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{[\nabla^2\phi(x_0+\tilde{h}e_i)]_{ii} - [\nabla^2\phi(x_0)]_{ii}}{\tilde{h}}\right)^2}\right\} \tag{4.6}$$

and $\tilde{h} = \max\{1, |[x_0]_i|\}\sqrt{\epsilon_M}$. Note that noiseless forward differences are applied to the true Hessian to estimate the third derivative along each coordinate direction at the initial point. This synthetic `Fixed` strategy is presented for benchmarking purposes; it is not generally viable in practice.

Representative results are shown in Figure 4.4. Similar to the forward-difference case, our algorithm is able to obtain higher accuracy in the solution compare to the `Fixed` strategy, but at higher cost as expected. Complete experimental results for all problems and noise levels are presented in Appendix B.

5 Final Remarks

We have developed a principled and robust procedure for determining the difference interval for estimating gradients in optimization methods, assuming that the noise level is known. Our procedure applies to any finite-difference scheme, including central- and higher-order difference schemes. It performs a bisection search on a ratio that balances the truncation and measurement errors such that one typically attains a near-optimal difference interval.

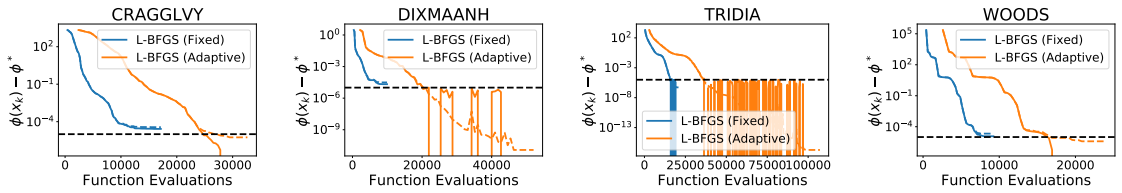

Figure 4.4: Comparison of central-difference L-BFGS methods with difference intervals determined using a fixed interval and our adaptive algorithm. Comparisons are made on representative problems with noise level $\epsilon_f = 10^{-5}$. The solid line plots the observed function value and the dashed line plots the true function value. The dashed black line shows the noise level ϵ_f of the function.

Whereas some methods for estimating the difference interval prioritize efficiency, such as Moré and Wild [19], and others compromise cost and accuracy, such as Gill, et al. [9], our approach is designed to be as robust as possible so that finite-difference gradient approximations can be reliably used in established nonlinear optimization techniques for solving noisy problems.

As demonstrated in our experiments, reusing previous difference intervals from prior iterations allows us to reduce the cost of the estimation procedure. Additional savings can achieved by re-estimating the difference interval periodically; for simple problems only a few times during the course of the optimization will suffice. The ability to exploit parallelism by distributing the computation of the gradient is an advantage that should not be underestimated when comparing the finite-difference approach with other techniques for derivative-free optimization.

Acknowledgement

We are grateful to Oliver Zhuoran Liu and Shigeng Sun for their feedback on this work.

References

[1] Russell R Barton. Computing forward difference derivatives in engineering optimization. *Engineering optimization*, 20(3):205–224, 1992.

[2] Albert S Berahas, Richard H Byrd, and Jorge Nocedal. Derivative-free optimization of noisy functions via quasi-newton methods. *SIAM Journal on Optimization*, 29(2):965–993, 2019.

[3] Albert S Berahas, Liyuan Cao, Krzysztof Choromanski, and Katya Scheinberg. A theoretical and empirical comparison of gradient approximations in derivative-free optimization. *arXiv preprint arXiv:1905.01332*, 2019.

[4] R. C. M. Brekelmans, L. T. Driessen, H. J. M. Hamers, and Dick Den Hertog. Gradient estimation schemes for noisy functions. *Journal of Optimization Theory and Applications*, 126(3):529–551, 2005.

[5] TD Choi and Carl T Kelley. Superlinear convergence and implicit filtering. *SIAM Journal on Optimization*, 10(4):1149–1162, 2000.

[6] Andrew R Conn, Katya Scheinberg, and Luis N Vicente. *Introduction to derivative-free optimization*, volume 8. SIAM, 2009.

[7] A. R. Curtis and J. K. Reid. The choice of step lengths when using differences to approximate jacobian matrices. *IMA Journal of Applied Mathematics*, 13(1):121–126, 1974.

[8] Bengt Fornberg. Generation of finite difference formulas on arbitrarily spaced grids. *Mathematics of computation*, 51(184):699–706, 1988.

[9] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. Computing forward-difference intervals for numerical optimization. *SIAM Journal on Scientific and Statistical Computing*, 4(2):310–321, 1983.

[10] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization*. Academic Press, London, 1981.

[11] Nicholas IM Gould, Dominique Orban, and Philippe L Toint. CUTEst: a constrained and unconstrained testing environment with safe threads for mathematical optimization. *Computational Optimization and Applications*, 60(3):545–557, 2015.

[12] Warren Hare, Gabriel Jarry-Bolduc, and Chayne Planiden. Error bounds for overdetermined and underdetermined generalized centred simplex gradients. *arXiv preprint arXiv:2006.00742*, 2020.

[13] Warren Hare and Kashvi Srivastava. Applying complex-step derivative approximations in model-based derivative-free optimization, 2020.

[14] Carl T. Kelley. *Implicit filtering*, volume 23. SIAM, 2011.

[15] Jack Kiefer, Jacob Wolfowitz, et al. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3):462–466, 1952.

[16] Jeffrey Larson, Matt Menickelly, and Stefan M Wild. Derivative-free optimization methods. *Acta Numerica*, 28:287–404, 2019.

[17] James N Lyness and Cleve B Moler. Numerical differentiation of analytic functions. *SIAM Journal on Numerical Analysis*, 4(2):202–210, 1967.

[18] Jorge J Moré and Stefan M Wild. Estimating computational noise. *SIAM Journal on Scientific Computing*, 33(3):1292–1314, 2011.

[19] Jorge J Moré and Stefan M Wild. Estimating derivatives of noisy simulations. *ACM Transactions on Mathematical Software (TOMS)*, 38(3):19, 2012.

[20] Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer New York, 2 edition, 1999.

[21] Hao-Jun Michael Shi, Yuchen Xie, Richard Byrd, and Jorge Nocedal. A noise-tolerant quasi-newton algorithm for unconstrained optimization. *arXiv preprint arXiv:2010.04352*, 2020.

[22] Hao-Jun Michael Shi, Melody Qiming Xuan, Figen Oztoprak, and Jorge Nocedal. On the numerical performance of derivative-free optimization methods based on finite-difference approximations. *arXiv preprint arXiv:2102.09762*, 2021.

[23] William Squire and George Trapp. Using complex variables to estimate derivatives of real functions. *SIAM review*, 40(1):110–112, 1998.

[24] R. S. Stepleman and N. D. Winarsky. Adaptive numerical differentiation. *Mathematics of Computation*, 33(148):1257–1264, 1979.

A Finite-Difference Formula Derivation and Tables

We summarize the different standard finite-difference schemes with equidistant points, their theoretical error, optimal steplength, and optimal error in terms of the noise level ϵ_f and local bound on the q-th derivative L_q for a smooth univariate function $\phi : \mathbb{R} \to \mathbb{R}$ in Tables A.1 and A.2. For completeness, we provide a complete derivation of the errors for a generic finite-difference approximation to the d-th order derivative below.

We will use $f : \mathbb{R} \to \mathbb{R}$ to denote the noisy function evaluations $f(t) = \phi(t) + \epsilon(t)$. We will consider two settings for $\epsilon(t)$: (1) we will assume that $\epsilon(t)$ is bounded, i.e., $|\epsilon(t)| \leq \epsilon_f$ for all t; (2) we will assume that $\epsilon(t)$ is a random variable with $\mathbb{E}[\epsilon(t)] = 0$ and $\mathbb{E}[\epsilon(t)^2] = \sigma_f^2$ for all t. The tables vary the number of evaluated points m and is dependent on the local Lipschitz constant $L_q \geq 0$ which bounds the q-th derivative

$$|\phi^{(q)}(t + h_0 s)| \leq L_q$$

for all $s \in [s_1, s_m]$, where q is the order of the remainder term in the Taylor expansion.

In the most general case, given distinct shifts $\{s_j\}_{j=1}^m$ and points $\{t_1, ..., t_m\} = \{t + hs_1, t + hs_2, ..., t + hs_m\}$, one can derive a generic finite-difference method to approximate the d-th derivative of the form:

$$\phi^{(d)}(t) \approx \frac{\sum_{j=1}^m w_j f(t + s_j h)}{h^d} = f^{(d)}(t; h).$$

We will assume without loss of generality that $s_1 < s_2 < ... < s_m$. First, note that $f^{(d)}$ can be decomposed into a noiseless finite-difference formula and its corresponding error:

$$f^{(d)}(t; h) = \frac{\sum_{j=1}^m w_j \phi(t + s_j h)}{h^d} + \frac{\sum_{j=1}^m w_j \epsilon(t + s_j h)}{h^d}.$$

Considering the noiseless finite-difference term, since the function is smooth, one can write the Lagrange remainder form of the Taylor series expansions for each function evaluation without noise as:

$$\phi(t + hs_j) = \sum_{l=0}^{q-1} \frac{1}{l!} \phi^{(l)}(t) s_j^l + \frac{1}{q!} \phi^{(q)}(\xi_j) s_j^q$$

for $\xi_j \in [t, t + hs_j]$ for $j = 1, ..., m$. Therefore, if the weights w satisfy

$$\frac{1}{d!} \sum_{j=1}^m w_j s_j^l = \begin{cases} 0 & \text{for } l \neq d, \ l = 0, 1, ..., q-1 \\ 1 & \text{for } l = d \end{cases}$$

then

$$\frac{\sum_{j=1}^m w_j \phi(t + s_j h)}{h^d} = \phi^{(d)}(t) + \frac{h^{q-d}}{q!} \sum_{j=1}^m w_j \phi^{(q)}(\xi_j) s_j^q.$$

This can be written compactly by the linear system of equations:

$$V(s)^T w = d! \cdot e_{p-d}$$

where $V(s) \in \mathbb{R}^{m \times q}$ is the Vandermonde matrix defined as

$$V(s) = \begin{bmatrix} s_1^{q-1} & s_1^{q-2} & \dots & s_1^0 \\ s_2^{q-1} & s_2^{q-2} & \dots & s_2^0 \\ \vdots & \vdots & \ddots & \vdots \\ s_m^{q-1} & s_m^{q-2} & \dots & s_m^0 \end{bmatrix}$$

and $e_{p-d} \in \mathbb{R}^p$ is the $(p-d)$-th coordinate vector.

To derive a reasonable bound on the total error, suppose we are given $h_0 > 0$ and a bound on $\phi^{(q)}$

$$|\phi^{(q)}(t + sh_0)| \leq L_q$$

for all $s \in [s_1, s_m]$. If we assume that the error is bounded, i.e., $|\epsilon(t)| \leq \epsilon_f$, then one can then bound the error in the approximation by:

$$|f^{(d)}(t;h) - \phi^{(d)}(t)| \leq \frac{L_q h^{q-d}}{q!} \sum_{j=1}^{m} |w_j s_j^q| + \frac{\|w\|_1 \epsilon_f}{h^d} = \epsilon_g(h)$$

for all $0 < h \leq h_0$. If we assume instead that $\mathrm{Var}(\epsilon(t)) = \sigma_f^2$, then we can similarly show

$$\mathbb{E}[(f^{(d)}(t;h) - \phi^{(d)}(t))^2] \leq \frac{L_q^2 h^{2(q-d)}}{(q!)^2} \sum_{j=1}^{m} w_j^2 s_j^{2q} + \frac{\|w\|_2^2 \sigma_f^2}{h^{2d}} = \sigma_g^2(h)$$

for all $0 < h \leq h_0$.

The above Taylor series analysis is pessimistic in that it requires multiple ξ_j points, and therefore yields a loose bound when applying the triangle inequality. Instead, one can consider the derivation of finite-difference schemes for approximating the *first* derivative at an interpolation point using Lagrange polynomials, which yields a tighter bound on the error.

As above, suppose we are given distinct points $\{t_1, ..., t_m\} = \{t + hs_1, ..., t + hs_m\}$ and we are interested in approximating $\phi^{(1)}(t)$. Recall that the Lagrange basis polynomials are defined as:

$$\psi_{p,j}(\tilde{t}) = \frac{\prod_{k \neq j}(\tilde{t} - t_k)}{\prod_{k \neq j}(t_j - t_k)} = \frac{\omega_m(\tilde{t})}{\omega_m^{(1)}(t_j)(\tilde{t} - t_j)}, \qquad \omega_m(\tilde{t}) = \prod_{j=1}^{m} (\tilde{t} - t_j).$$

Then the Lagrange interpolation is defined as:

$$\ell(\tilde{t}) = \sum_{j=1}^{m} \psi_{m,j}(\tilde{t}) \phi(t_j).$$

It is well-known that the remainder is

$$\phi(\tilde{t}) - \ell(\tilde{t}) = \frac{\omega_m(\tilde{t})}{m!} \phi^{(m)}(\xi)$$

for some $\xi \in [t_1, t_m]$. Note that the finite-difference formula can simply be obtained by differentiating the Lagrange polynomial

$$\ell^{(1)}(\tilde{t}) = \sum_{j=1}^{m} \psi_{m,j}^{(1)}(\tilde{t})\phi(t_j).$$

Therefore, the finite-difference coefficients are obtained by evaluating $\psi_{m,j}^{(1)}(\tilde{t})$. The error is also obtained by noting

$$\phi^{(1)}(\tilde{t}) = \ell^{(1)}(\tilde{t}) + \frac{\omega_m^{(1)}(\tilde{t})}{m!}\phi^{(m)}(\xi) + \frac{\omega_m(\tilde{t})}{m!}\phi^{(m)}(\xi)\frac{d\xi}{dx}.$$

Since

$$\omega_m^{(1)}(\tilde{t}) = \sum_{j=1}^{m}\prod_{k\neq j}(\tilde{t} - t_k),$$

plugging in $\tilde{t} = t_i$ for any $i = 1, ..., m$, we get the following equality

$$\phi^{(1)}(t_i) = \ell^{(1)}(t_i) + \frac{\omega_m^{(1)}(t_i)}{m!}\phi^{(m)}(\xi) = \ell^{(1)}(t_i) + \prod_{j\neq i}(t_i - t_j)\frac{\phi^{(m)}(\xi)}{m!}.$$

Given $h_0 > 0$ and a bound on $\phi^{(m)}$

$$|\phi^{(m)}(t + h_0 s)| \leq L_m$$

for all $s \in [s_1, s_m]$ and assuming $t = t_i$ is one of the interpolation points, we obtain the bound

$$|\phi^{(1)}(t) - \ell^{(1)}(t)| \leq \frac{L_m h^{m-1}}{m!}\left|\prod_{j\neq i} s_j\right|$$

and if we incorporate the error in the function evaluations, we obtain a error and variance bounds of

$$|f^{(1)}(t;h) - \phi^{(1)}(t)| \leq \frac{L_m h^{m-1}}{m!}\left|\prod_{j\neq i} s_j\right| + \frac{\|w\|_1 \epsilon_f}{h} = \epsilon_g(h)$$

$$\mathbb{E}[(f^{(1)}(t;h) - \phi^{(1)}(t))^2] \leq \frac{L_m^2 h^{2(m-1)}}{(m!)^2}\prod_{j\neq i} s_j^2 + \frac{\|w\|_2^2 \sigma_f^2}{h^2} = \sigma_g^2(h)$$

for all $0 < h \leq h_0$.

m	$f^{(1)}(t;h)$	$\epsilon_g(h)$	h^*	$\epsilon_g(h^*)$
2	$\frac{f(t+h)-f(t)}{h}$	$\frac{L_2 h}{2}+\frac{2\epsilon_f}{h}$	$2\sqrt{\frac{\epsilon_f}{L_2}}$	$2\sqrt{L_2\epsilon_f}$
3	$\frac{-3f(t)+4f(t+h)-f(t+2h)}{2h}$	$\frac{L_3 h^2}{3}+\frac{4\epsilon_f}{h}$	$\sqrt[3]{\frac{6\epsilon_f}{L_3}}$	$6^{2/3}L_3^{1/3}\epsilon_f^{2/3}$
4	$\frac{-11f(t)+18f(t+h)-9f(t+2h)+2f(t+3h)}{6h}$	$\frac{L_4 h^3}{4}+\frac{20\epsilon_f}{3h}$	$\sqrt[4]{\frac{80\epsilon_f}{9L_4}}$	$\frac{8\cdot 5^{3/4}}{3\sqrt{3}}L_4^{1/4}\epsilon_f^{3/4}$
5	$\frac{-25f(t)+48f(t+h)-36f(t+2h)+16f(t+3h)-3f(t+4h)}{12h}$	$\frac{L_5 h^4}{5}+\frac{32\epsilon_f}{3h}$	$\sqrt[5]{\frac{40\epsilon_f}{3L_5}}$	$4\left(\frac{5}{3}\right)^{4/5}2^{2/5}L_5^{1/5}\epsilon_f^{4/5}$

Table A.1: Table containing the finite-difference formula, deterministic error bound $|f^{(1)}(t;h)-\phi^{(1)}(t)|\leq\epsilon_g(h)$ for generic h, optimal h^*, and optimal error $\epsilon_g(h^*)$ for forward-difference schemes with number of points $m\in\{2,3,4,5\}$.

m	$f^{(1)}(t;h)$	$\epsilon_g(h)$	h^*	$\epsilon_g(h^*)$
2	$\frac{f(t+h)-f(t-h)}{2h}$	$\frac{L_3 h^2}{6}+\frac{\epsilon_f}{h}$	$\sqrt[3]{\frac{3\epsilon_f}{L_3}}$	$\frac{3^{2/3}}{2}L_3^{1/3}\epsilon_f^{2/3}$
4	$\frac{f(t-2h)-8f(t-h)+8f(t+h)-f(t+2h)}{12h}$	$\frac{L_5 h^4}{30}+\frac{3\epsilon_f}{2h}$	$\sqrt[5]{\frac{45\epsilon_f}{4L_5}}$	$\frac{1}{4}\left(\frac{3}{2}\right)^{4/5}5^{4/5}L_5^{1/5}\epsilon_f^{4/5}$
6	$\frac{-f(t-3h)+9f(t-2h)-45f(t-h)+45f(t+h)-9f(t+2h)+f(t+3h)}{60h}$	$\frac{L_7 h^6}{140}+\frac{11\epsilon_f}{6h}$	$\sqrt[7]{\frac{385\epsilon_f}{9L_7}}$	$\frac{77^{6/7}}{12\cdot 3^{5/7}\cdot\sqrt[7]{5}}L_7^{1/7}\epsilon_f^{6/7}$

Table A.2: Table containing the finite-difference formula, deterministic error bound $|f^{(1)}(t;h)-\phi^{(1)}(t)|\leq\epsilon_g(h)$ for generic h, optimal h^*, and optimal error $\epsilon_g(h^*)$ for central-difference schemes with number of points $m\in\{2,4,6\}$.

p	$f^{(1)}(t;h)$	$\sigma_g^2(h)$	h^*	$\sigma_g(h^*)$
1	$\frac{f(t+h)-f(t)}{h}$	$\frac{L_2^2h^2}{4}+\frac{2\epsilon_f^2}{h^2}$	$8^{1/4}\sqrt{\frac{\epsilon_f}{L_2}}$	$2^{1/4}\sqrt{L_2\epsilon_f}$
2	$\frac{-3f(t)+4f(t+h)-f(t+2h)}{2h}$	$\frac{L_3^2h^4}{9}+\frac{13\epsilon_f^2}{2h^2}$	$(\frac{3}{2})^{1/3}13^{1/6}\sqrt[3]{\frac{\epsilon_f}{L_3}}$	$\frac{\sqrt[6]{3}\sqrt[3]{13}}{2^{2/3}}L_3^{1/3}\epsilon_f^{2/3}$
3	$\frac{-11f(t)+18f(t+h)-9f(t+2h)+2f(t+3h)}{6h}$	$\frac{L_4^2h^6}{16}+\frac{265\epsilon_f^2}{18h^2}$	$(\frac{2}{3})^{3/8}265^{1/8}\sqrt[4]{\frac{\epsilon_f}{L_4}}$	$\frac{1}{3}\sqrt[8]{\frac{2}{3}}265^{3/8}L_4^{1/4}\epsilon_f^{3/4}$
4	$\frac{-25f(t)+48f(t+h)-36f(t+2h)+16f(t+3h)-3f(t+4h)}{12h}$	$\frac{L_5^2h^8}{25}+\frac{2245\epsilon_f^2}{72h^2}$	$\frac{5^{3/10}449^{1/10}}{\sqrt{2}\sqrt[5]{3}}\sqrt[5]{\frac{\epsilon_f}{L_5}}$	$\frac{5^{7/10}449^{2/5}}{4\cdot3^{4/5}}L_5^{1/5}\epsilon_f^{4/5}$

Table A.3: Table containing the finite-difference formula, MSE error bound $\mathbb{E}[(f^{(1)}(t;h)-\phi^{(1)}(t))^2]\leq\sigma_g^2(h)$ for generic h, optimal h^*, and optimal error $\sigma_g(h^*)$ for forward-difference schemes with number of points $m\in\{2,3,4,5\}$.

p	$f^{(1)}(t;h)$	$\sigma_g^2(h)$	h^*	$\sigma_g(h^*)$
2	$\frac{f(t+h)-f(t-h)}{2h}$	$\frac{L_3^2h^4}{36}+\frac{\epsilon_f^2}{2h^2}$	$\sqrt[3]{3}\sqrt[3]{\frac{\epsilon_f}{L_3}}$	$\frac{\sqrt[6]{3}}{2}L_3^{1/3}\epsilon_f^{2/3}$
4	$\frac{f(t-2h)-8f(t-h)+8f(t+h)-f(t+2h)}{12h}$	$\frac{L_5^2h^8}{900}+\frac{65\epsilon_f^2}{72h^2}$	$\left(\frac{5}{2}\right)^{3/10}13^{1/10}\sqrt[5]{\frac{\epsilon_f}{L_5}}$	$\frac{5^{7/10}\cdot13^{2/5}}{12\cdot\sqrt[5]{2}}L_5^{1/5}\epsilon_f^{4/5}$
6	$\frac{-f(t-3h)+9f(t-2h)-45f(t-h)+45f(t+h)-9f(t+2h)+f(t+3h)}{60h}$	$\frac{L_7^2h^{12}}{140^2}+\frac{2107\epsilon_f^2}{1800h^2}$	$\frac{7^{2/7}43^{1/14}}{3^{3/14}}\sqrt[7]{\frac{\epsilon_f}{L_7}}$	$\frac{7^{17/14}43^{3/7}}{60\cdot3^{2/7}}L_7^{1/7}\epsilon_f^{6/7}$

Table A.4: Table containing the finite-difference formula, MSE error bound $\mathbb{E}[(f^{(1)}(t;h)-\phi^{(1)}(t))^2]\leq\sigma_g^2(h)$ for generic h, optimal h^*, and optimal error $\sigma_g(h^*)$ for central-difference schemes with number of points $m\in\{2,4,6\}$.

B Complete Experimental Results

Here, we present the complete experimental results from Section 4.

B.1 Robustness to Different Noise Levels

We test our procedure on a simple function $\phi(t) = \cos(t)$ for different noise levels using different schemes listed in Table 4.1. These are shown in Figure B.1. Detailed numerical results, including the number of iterations and relative error, are listed in Table B.1.

Observe that our method is able to consistently achieve low relative error using a similar number of function evaluations across all tested noise levels. This is a desirable property, as it demonstrates that our initial choice of the interval h and our method is consistent over different noise levels.

B.2 Affine Invariance

One advantage of our proposed method is that the testing ratio remains unchanged under affine transformations of the function. It is particularly obvious that our procedure is invariant when adding a constant to the function. Hence, we focus on transformations of the form $\phi(t) \to a \cdot \phi(b \cdot t)$ for some $a, b \neq 0$.

To do this, we test Algorithm 2 on the function $\phi(t) = a \cdot \sin(b \cdot t)$ at $t = 0$ for various a and b. We fix the noise level to be $\epsilon_f = 10^{-3}$. The results are shown in Figure B.2. Detailed results can be found in Table B.2 and B.3. As seen in Figure B.2, our method is affine-invariant and can output consistently correct results for different a and b.

B.3 Difficult and Special Examples

Here, we present the full table of results for the examples listed in Section 4 in Table B.4 with $\epsilon_f = 10^{-3}$. For reference, the considered problems are:

1. $\phi(t) = \left(e^t - 1\right)^2$, at $t = -8$.
2. $\phi(t) = e^{100t}$, at $t = 0.01$.
3. $\phi(t) = t^4 + 3t^2 - 10t$, at $t = 0.99999$.
4. $\phi(t) = 10000t^3 + 0.01t^2 + 5t$, at $t = 10^{-9}$.

B.4 Comparison with Moré-Wild Heuristic

We compare our adaptive forward-difference procedure against the Moré-Wild heuristic [19], as described in Section 2.1.

First, observe that if function ϕ has (near) central symmetry at t, then Moré-Wild heuristic is very likely to fail. To demonstrate this, we test on $\phi(t) = \sin(t)$ with various value of t close to 0 and different noise levels ϵ_f. The results are summarized in Table B.5.

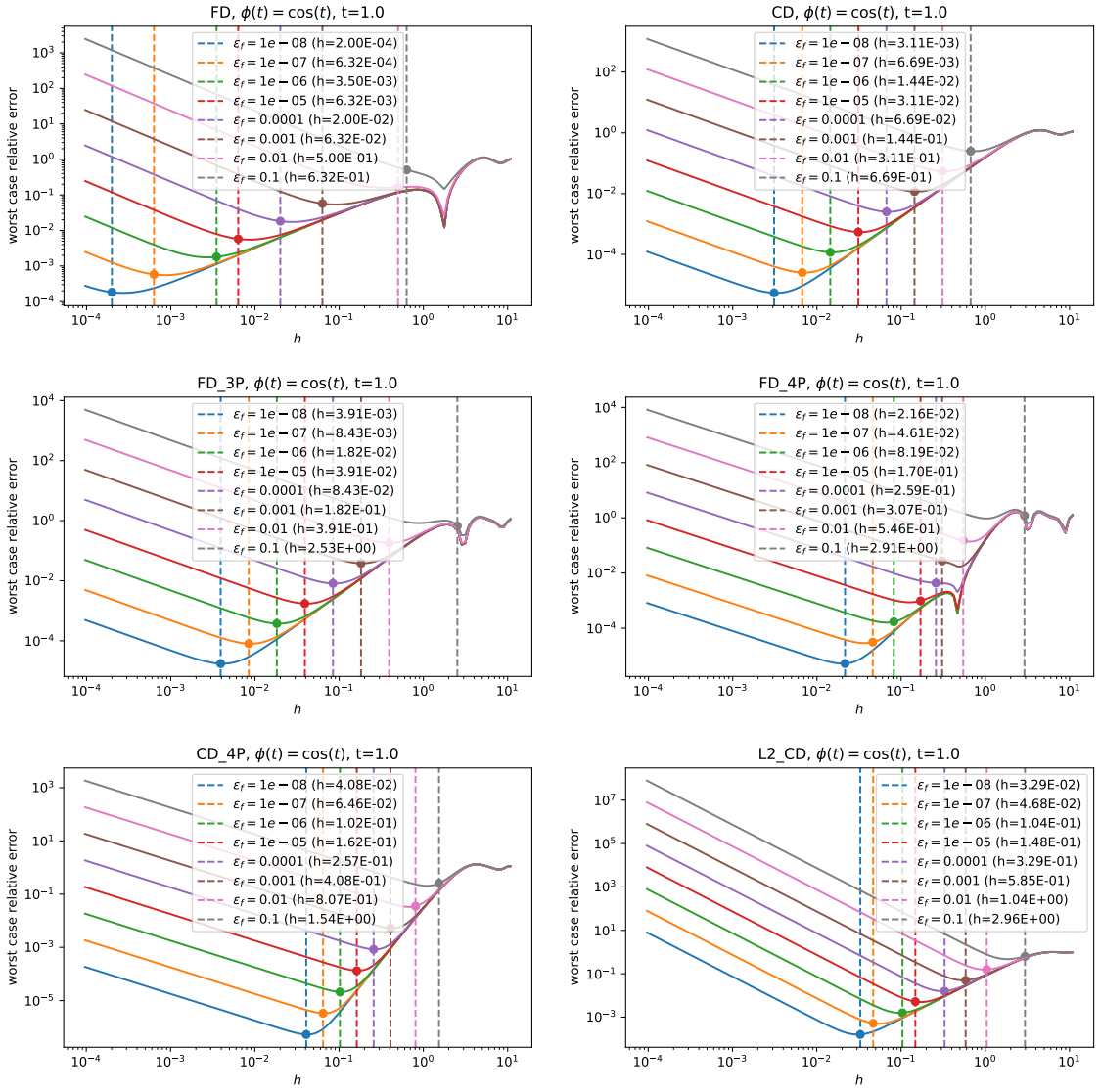

Figure B.1: Worst case relative error $\delta_S(h; \phi, t, \epsilon_f)$ against h on function $\phi(t) = \cos(t)$ with different noise levels; the vertical dashed line represents the $h_\dagger$ output by Algorithm 2.

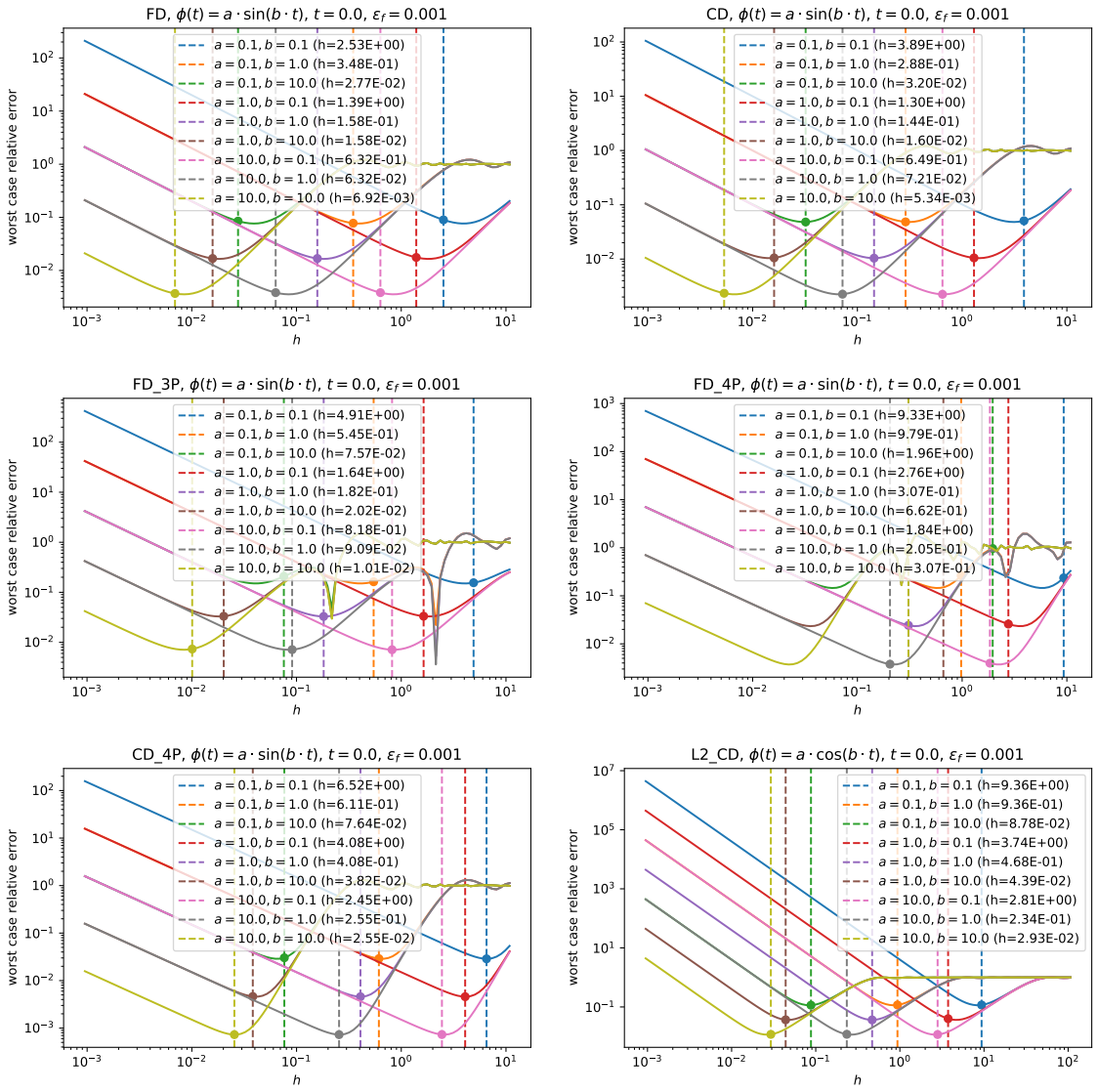

Figure B.2: Worst case relative error $\delta_S(h; \phi, t, \epsilon_f)$ against h on function $\phi(t) = a \cdot \sin(b \cdot t)$ for different a and b; the vertical dashed line represents the $h_\dagger$ output by Algorithm 2.

Next, we test our adaptive procedure and the Moré-Wild heuristic on $\phi(t) = a \cdot (\exp(b \cdot t) - 1)$ at $t = 0$, with a fixed noise level: $\epsilon_f =$ `1E-3`. We summarize our result in Table B.6. Notice that Moré-Wild heuristic may not be able to find a suitable estimation for h, in which case a failure is declared. In such cases, we will report the result as "$--$".

We can see that when the Moré-Wild heuristic does not declare a failure, it usually outputs an interval h that is quite close to our procedure and produces similar relative error as ours. However, there are many cases where Moré-Wild heuristic fails, while our procedure works very robustly in all cases.

B.5 Finite-Difference L-BFGS

We present the total number of function evaluations and final optimality gap $\phi(x) - \phi^*$ used by each method in Tables B.7–B.14.

In general, our adaptive procedure is more robust to different noise levels. Our method only fails when the initial choice of h is not sufficiently small to initially identify the local behavior of the function. This can be seen, for example, with the `BOX2` example. On the other hand, Moré and Wild's heuristic frequently fails when the noise level is large (for example, with $\epsilon_f = 10^{-1}$). This is due both to the case where $\phi(x) \approx 0$ and hence (2.11) fails, as well as the case where two iterations are insufficient to find an h that satisfies their conditions. In both cases, we denote a failure case with *. As expected, using a fixed interval is always efficient, but may perform poorly when the Hessian in the function changes, as described in Section 4.

scheme	$h_\dagger$	h^*	r	n_iter	num_eval	relative error	ϵ_f
FD	2.00e-04	2.72e-04	2.08	1	3	1.86e-05	1.00e-08
FD	6.32e-04	8.59e-04	2.00	1	3	4.21e-05	1.00e-07
FD	3.50e-03	2.73e-03	4.79	4	8	1.17e-03	1.00e-06
FD	6.32e-03	8.64e-03	1.77	1	3	1.72e-03	1.00e-05
FD	2.00e-02	2.76e-02	2.00	1	3	1.69e-03	1.00e-04
FD	6.32e-02	9.05e-02	1.73	1	3	5.07e-04	1.00e-03
FD	5.00e-01	1.73e+00	3.89	3	6	9.86e-02	1.00e-02
FD	6.32e-01	8.26e+00	1.52	1	3	2.97e-01	1.00e-01
CD	3.11e-03	3.29e-03	2.40	1	4	1.89e-06	1.00e-08
CD	6.69e-03	7.09e-03	2.66	1	4	5.39e-06	1.00e-07
CD	1.44e-02	1.53e-02	2.72	1	4	9.35e-06	1.00e-06
CD	3.11e-02	3.29e-02	2.05	1	4	3.34e-04	1.00e-05
CD	6.69e-02	7.09e-02	2.18	1	4	1.33e-03	1.00e-04
CD	1.44e-01	1.53e-01	2.55	1	4	3.32e-03	1.00e-03
CD	3.11e-01	3.30e-01	1.89	1	4	3.84e-02	1.00e-02
CD	6.69e-01	7.74e+00	2.01	1	4	5.71e-02	1.00e-01
FD_3P	3.91e-03	4.14e-03	2.88	1	5	1.01e-05	1.00e-08
FD_3P	8.43e-03	8.92e-03	3.24	1	5	2.01e-05	1.00e-07
FD_3P	1.82e-02	1.92e-02	2.76	1	5	2.05e-04	1.00e-06
FD_3P	3.91e-02	4.11e-02	3.28	1	5	5.82e-04	1.00e-05
FD_3P	8.43e-02	8.77e-02	2.86	1	5	5.65e-03	1.00e-04
FD_3P	1.82e-01	1.86e-01	3.10	1	5	2.29e-02	1.00e-03
FD_3P	3.91e-01	2.99e+00	2.88	1	5	9.96e-02	1.00e-02
FD_3P	2.53e+00	2.12e+01	5.75	2	7	4.17e-01	1.00e-01
FD_4P	2.16e-02	2.04e-02	9.36	5	18	2.14e-06	1.00e-08
FD_4P	4.61e-02	3.67e-02	16.51	4	13	1.14e-05	1.00e-07
FD_4P	8.19e-02	4.76e-01	10.80	4	13	9.43e-05	1.00e-06
FD_4P	1.70e-01	1.25e-01	4.40	5	18	6.15e-04	1.00e-05
FD_4P	2.59e-01	3.28e+00	14.62	4	13	7.80e-04	1.00e-04
FD_4P	3.07e-01	3.28e+00	4.23	1	6	5.46e-03	1.00e-03
FD_4P	5.46e-01	8.78e+00	6.44	1	6	3.19e-02	1.00e-02
FD_4P	2.91e+00	8.79e+00	4.28	2	8	9.55e-01	1.00e-01
CD_4P	4.08e-02	4.22e-02	2.52	1	6	1.16e-07	1.00e-08
CD_4P	6.46e-02	6.69e-02	2.04	1	6	8.34e-07	1.00e-07
CD_4P	1.02e-01	1.06e-01	1.95	1	6	8.81e-06	1.00e-06
CD_4P	1.62e-01	1.68e-01	1.79	1	6	4.29e-05	1.00e-05
CD_4P	2.57e-01	2.67e-01	1.71	1	6	4.00e-04	1.00e-04
CD_4P	4.08e-01	4.25e-01	1.89	1	6	8.32e-04	1.00e-03
CD_4P	8.07e-01	7.97e+00	4.83	4	20	5.87e-03	1.00e-02
CD_4P	1.54e+00	2.06e+01	4.25	3	14	2.34e-01	1.00e-01
L2_CD	3.29e-02	3.07e-02	3.78	4	15	1.00e-04	1.00e-08
L2_CD	4.68e-02	5.46e-02	1.89	1	5	8.55e-05	1.00e-07
L2_CD	1.04e-01	9.71e-02	4.22	4	15	7.28e-04	1.00e-06
L2_CD	1.48e-01	1.73e-01	1.90	1	5	1.06e-03	1.00e-05
L2_CD	3.29e-01	3.07e-01	4.03	4	15	8.28e-03	1.00e-04
L2_CD	5.85e-01	5.49e-01	3.81	4	15	2.84e-02	1.00e-03
L2_CD	1.04e+00	9.87e-01	3.45	4	15	7.95e-02	1.00e-02
L2_CD	2.96e+00	1.55e+01	5.45	2	7	5.40e-01	1.00e-01

Table B.1: Detailed results for $\phi(t) = \cos(t)$ with different noise levels; r represents the final testing ratio; h^* is the h that minimizes $\delta_S(h; \phi, t, \epsilon_f)$ reported by `minimize_scalar` function in `scipy.optimize` and could be unreliable.

a	b	scheme	$h_\dagger$	h^*	r	n_iter	num_eval	relative error
0.10	0.10	FD	2.53e+00	3.94e+00	1.89	5	8	1.75e-02
0.10	1.00	FD	3.48e-01	3.94e-01	4.34	6	11	5.29e-02
0.10	10.00	FD	2.77e-02	3.94e-02	2.68	4	8	2.01e-05
1.00	0.10	FD	1.39e+00	1.82e+00	2.61	7	12	1.30e-02
1.00	1.00	FD	1.58e-01	1.82e-01	4.91	3	6	3.01e-03
1.00	10.00	FD	1.58e-02	1.82e-02	4.58	2	4	6.97e-03
10.00	0.10	FD	6.32e-01	8.44e-01	3.23	4	7	4.76e-04
10.00	1.00	FD	6.32e-02	8.44e-02	3.38	1	3	2.83e-04
10.00	10.00	FD	6.92e-03	8.44e-03	3.34	5	9	2.92e-03
0.10	0.10	CD	3.89e+00	3.12e+00	5.47	4	10	2.51e-02
0.10	1.00	CD	2.88e-01	3.12e-01	2.30	3	10	1.38e-02
0.10	10.00	CD	3.20e-02	3.12e-02	3.13	4	14	1.70e-02
1.00	0.10	CD	1.30e+00	1.44e+00	2.17	3	8	2.81e-03
1.00	1.00	CD	1.44e-01	1.44e-01	2.97	1	4	3.46e-03
1.00	10.00	CD	1.60e-02	1.44e-02	4.06	3	10	4.27e-03
10.00	0.10	CD	6.49e-01	6.70e-01	2.73	5	16	7.02e-04
10.00	1.00	CD	7.21e-02	6.70e-02	3.74	4	14	8.66e-04
10.00	10.00	CD	5.34e-03	6.70e-03	1.52	4	12	4.75e-04
0.10	0.10	FD_3P	4.91e+00	8.14e+01	2.93	4	11	2.13e-02
0.10	1.00	FD_3P	5.45e-01	8.14e+00	2.45	2	7	6.48e-02
0.10	10.00	FD_3P	7.57e-02	8.14e-01	4.90	5	15	1.61e-01
1.00	0.10	FD_3P	1.64e+00	2.14e+01	2.25	3	9	8.79e-03
1.00	1.00	FD_3P	1.82e-01	2.14e+00	3.63	1	5	1.86e-04
1.00	10.00	FD_3P	2.02e-02	1.83e-02	4.43	3	9	1.15e-02
10.00	0.10	FD_3P	8.18e-01	8.45e-01	3.48	5	13	1.71e-03
10.00	1.00	FD_3P	9.09e-02	8.45e-02	4.58	4	11	1.77e-03
10.00	10.00	FD_3P	1.01e-02	8.45e-03	6.67	6	15	1.00e-03
0.10	0.10	FD_4P	9.33e+00	8.44e+01	4.95	9	31	1.63e-01
0.10	1.00	FD_4P	9.79e-01	8.44e+00	9.64	8	32	1.50e-01
0.10	10.00	FD_4P	1.96e+00	1.78e+01	7.71	7	29	9.60e-01
1.00	0.10	FD_4P	2.76e+00	3.59e+00	4.47	3	11	3.59e-03
1.00	1.00	FD_4P	3.07e-01	3.59e-01	6.55	1	6	8.16e-03
1.00	10.00	FD_4P	6.62e-01	5.88e+00	10.15	8	35	9.49e-01
10.00	0.10	FD_4P	1.84e+00	2.25e+00	6.57	4	14	3.68e-04
10.00	1.00	FD_4P	2.05e-01	2.71e+00	10.68	3	10	2.90e-04
10.00	10.00	FD_4P	3.07e-01	4.62e+00	16.47	1	6	8.38e-01
0.10	0.10	CD_4P	6.52e+00	7.97e+01	2.12	5	14	5.73e-03
0.10	1.00	CD_4P	6.11e-01	7.97e+00	1.57	3	14	4.45e-03
0.10	10.00	CD_4P	7.64e-02	7.97e-01	4.32	5	18	1.06e-02
1.00	0.10	CD_4P	4.08e+00	4.10e+00	2.30	7	26	9.02e-04
1.00	1.00	CD_4P	4.08e-01	4.10e-01	2.30	1	6	9.02e-04
1.00	10.00	CD_4P	3.82e-02	4.10e-02	1.68	6	20	6.98e-04
10.00	0.10	CD_4P	2.45e+00	2.58e+00	1.89	5	18	1.18e-04
10.00	1.00	CD_4P	2.55e-01	2.58e-01	2.31	4	20	1.39e-04
10.00	10.00	CD_4P	2.55e-02	2.58e-02	2.31	5	14	1.39e-04

Table B.2: Detailed results for $\phi(t) = a \cdot \sin(b \cdot t)$ with $\epsilon_f =$ `1E-3`; r represents the final testing ratio; h^* is the h that minimizes $\delta_S(h; \phi, t, \epsilon_f)$ reported by `minimize_scalar` function in `scipy.optimize` and could be unreliable.

a	b	scheme	$h_\dagger$	h^*	r	n_iter	num_eval	relative error
0.10	0.10	L2_CD	9.36e+00	8.42e+00	4.39	8	23	5.97e-02
0.10	1.00	L2_CD	9.36e-01	8.42e-01	4.79	2	7	4.28e-02
0.10	10.00	L2_CD	8.78e-02	8.99e-01	3.28	5	15	5.32e-02
1.00	0.10	L2_CD	3.74e+00	4.70e+00	1.99	4	11	9.49e-03
1.00	1.00	L2_CD	4.68e-01	4.70e-01	2.57	1	5	2.13e-02
1.00	10.00	L2_CD	4.39e-02	4.70e-02	2.15	6	17	1.70e-02
10.00	0.10	L2_CD	2.81e+00	2.64e+00	3.59	5	15	7.63e-03
10.00	1.00	L2_CD	2.34e-01	2.64e-01	2.62	2	7	6.94e-04
10.00	10.00	L2_CD	2.93e-02	2.64e-02	5.02	5	13	3.94e-03

Table B.3: Detailed results for $\phi(t) = a \cdot \sin(b \cdot t)$ with $\epsilon_f =$ 1E-3; r represents the final testing ratio; h^* is the h that minimizes $\delta_S(h; \phi, t, \epsilon_f)$ reported by minimize_scalar function in scipy.optimize and could be unreliable.

$\phi(t)$	scheme	$h_\dagger$	h^*	r	n_iter	num_eval	relative error
$\left(e^t - 1.0\right)^2$	FD	1.01e+00	1.46e+00	4.49	3	5	1.02e+00
$\left(e^t - 1.0\right)^2$	CD	1.30e+00	1.53e+00	3.38	3	8	5.73e-02
$\left(e^t - 1.0\right)^2$	FD_3P	8.18e-01	3.82e+02	2.28	5	13	6.63e-01
$\left(e^t - 1.0\right)^2$	FD_4P	9.21e-01	3.82e+02	4.14	2	8	4.15e+00
$\left(e^t - 1.0\right)^2$	CD_4P	1.43e+00	3.82e+02	1.84	5	22	1.11e+00
$\left(e^t - 1.0\right)^2$	L2_CD	2.34e+00	8.68e+00	3.03	6	19	3.90e-01
e^{100t}	FD	4.32e-04	3.79e-04	3.72	7	11	2.74e-02
e^{100t}	CD	1.19e-03	1.03e-03	4.29	7	20	3.09e-03
e^{100t}	FD_3P	1.12e-03	3.82e+02	3.08	8	19	6.81e-03
e^{100t}	FD_4P	1.90e-03	3.82e+02	6.54	8	23	4.97e-03
e^{100t}	CD_4P	3.18e-03	3.82e+02	2.15	8	20	4.60e-04
e^{100t}	L2_CD	3.66e-03	3.64e-03	3.01	8	19	1.18e-02
$t^4 + 3t^2 - 10t$	FD	1.58e-02	1.48e-02	3.55	2	4	7.97e+02
$t^4 + 3t^2 - 10t$	CD	4.81e-02	5.00e-02	2.94	2	6	2.15e+01
$t^4 + 3t^2 - 10t$	FD_3P	6.06e-02	6.16e-02	3.64	2	7	2.38e+02
$t^4 + 3t^2 - 10t$	FD_4P	1.54e-01	1.39e-01	11.90	4	13	1.93e+02
$t^4 + 3t^2 - 10t$	CD_4P	9.39e+02	4.87e+03	1.62	16	48	2.71e-03
$t^4 + 3t^2 - 10t$	L2_CD	2.34e-01	2.11e-01	4.53	2	7	5.38e-03
$10000t^3 + 0.01t^2 + 5t$	FD	3.95e-03	4.64e-03	4.36	3	5	5.46e-02
$10000t^3 + 0.01t^2 + 5t$	CD	3.56e-03	3.68e-03	2.63	6	18	4.02e-02
$10000t^3 + 0.01t^2 + 5t$	FD_3P	4.49e-03	4.64e-03	3.65	6	15	1.17e-02
$10000t^3 + 0.01t^2 + 5t$	FD_4P	6.72e+02	3.20e+03	11.72	8	22	2.37e-06
$10000t^3 + 0.01t^2 + 5t$	CD_4P	8.35e+02	1.03e+04	1.95	12	28	1.99e-07
$10000t^3 + 0.01t^2 + 5t$	L2_CD	9.59e+02	2.84e+03	1.95	12	27	7.49e-08

Table B.4: Detailed results for special examples, with $\epsilon_f =$ 1E-3; r represents the final testing ratio; h^* is the h that minimizes $\delta_S(h; \phi, t, \epsilon_f)$ reported by minimize_scalar function in scipy.optimize and could be unreliable.

ϵ_f	t	h_{MW}	δ_{MW}	$\overline{\delta}_{\mathrm{MW}}$	h_{ada}	δ_{ada}	$\overline{\delta}_{\mathrm{ada}}$
1.00e-08	1.00e-08	−−	−−	−−	3.20e-03	0.000	0.000
1.00e-08	1.00e-06	−−	−−	−−	3.20e-03	0.000	0.000
1.00e-08	1.00e-04	1.68e-02	0.000	0.000	3.20e-03	0.000	0.000
1.00e-08	1.00e-02	1.68e-03	0.000	0.000	2.00e-03	0.000	0.000
1.00e-08	0.00e+00	−−	−−	−−	3.20e-03	0.000	0.000
1.00e-06	1.00e-08	−−	−−	−−	1.40e-02	0.000	0.000
1.00e-06	1.00e-06	−−	−−	−−	1.40e-02	0.000	0.000
1.00e-06	1.00e-04	−−	−−	−−	1.40e-02	0.000	0.000
1.00e-06	1.00e-02	1.69e-02	0.000	0.000	1.40e-02	0.000	0.000
1.00e-06	0.00e+00	−−	−−	−−	1.40e-02	0.000	0.000
1.00e-04	1.00e-08	−−	−−	−−	5.00e-02	0.001	0.004
1.00e-04	1.00e-06	−−	−−	−−	6.50e-02	0.003	0.004
1.00e-04	1.00e-04	−−	−−	−−	5.00e-02	0.001	0.004
1.00e-04	1.00e-02	−−	−−	−−	5.00e-02	0.001	0.005
1.00e-04	0.00e+00	−−	−−	−−	6.50e-02	0.000	0.004
1.00e-02	1.00e-08	−−	−−	−−	2.00e-01	0.058	0.107
1.00e-02	1.00e-06	5.20e-01	0.067	0.083	3.50e-01	0.018	0.077
1.00e-02	1.00e-04	−−	−−	−−	3.50e-01	0.019	0.077
1.00e-02	1.00e-02	−−	−−	−−	3.50e-01	0.029	0.079
1.00e-02	0.00e+00	6.10e-01	0.085	0.094	3.50e-01	0.032	0.077

Table B.5: Comparison between the Moré-Wild heuristic against our adaptive procedure on function $\phi(t) = \sin(t)$ with various ϵ_f and t. We use "−−" to report the cases where Moré-Wild heuristic fails. Subscript "MW" labels the results corresponding to Moré-Wild heuristic, and subscript "ada" labels the results corresponding to our adaptive procedure; δ is the relative error, and $\overline{\delta}$ is the worst-case relative error.

a	b	h_{MW}	δ_{MW}	$\overline{\delta}_{\mathrm{MW}}$	h_{ada}	δ_{ada}	$\overline{\delta}_{\mathrm{ada}}$
0.01	0.01	−−	−−	−−	4.05e+01	0.135	0.727
0.01	0.10	−−	−−	−−	4.05e+00	0.145	0.727
0.01	1.00	−−	−−	−−	4.43e-01	0.275	0.710
0.01	10.00	4.68e-02	0.177	0.702	3.95e-02	0.084	0.732
0.01	100.00	−−	−−	−−	3.95e-03	0.123	0.732
0.10	0.01	−−	−−	−−	1.62e+01	0.032	0.209
0.10	0.10	−−	−−	−−	1.77e+00	0.074	0.207
0.10	1.00	1.64e-01	0.072	0.209	1.58e-01	0.095	0.210
0.10	10.00	1.62e-02	0.096	0.209	1.58e-02	0.055	0.210
0.10	100.00	−−	−−	−−	1.73e-03	0.078	0.207
1.00	0.01	−−	−−	−−	7.08e+00	0.041	0.065
1.00	0.10	−−	−−	−−	6.32e-01	0.034	0.064
1.00	1.00	5.26e-02	0.029	0.065	6.32e-02	0.036	0.064
1.00	10.00	5.28e-03	0.012	0.065	6.92e-03	0.014	0.064
1.00	100.00	−−	−−	−−	6.18e-04	0.009	0.064
10.00	0.01	−−	−−	−−	2.53e+00	0.012	0.021
10.00	0.10	1.76e-01	0.011	0.020	2.53e-01	0.009	0.021
10.00	1.00	1.68e-02	0.006	0.020	1.58e-02	0.005	0.021
10.00	10.00	1.68e-03	0.002	0.020	2.47e-03	0.014	0.021
10.00	100.00	−−	−−	−−	2.47e-04	0.010	0.021
100.00	0.01	−−	−−	−−	6.32e-01	0.003	0.006
100.00	0.10	5.39e-02	0.002	0.006	6.32e-02	0.004	0.006
100.00	1.00	5.31e-03	0.001	0.006	6.92e-03	0.001	0.006
100.00	10.00	5.31e-04	0.000	0.006	6.18e-04	0.001	0.006
100.00	100.00	−−	−−	−−	6.18e-05	0.001	0.006

Table B.6: Comparison between the Moré-Wild heuristic against our adaptive procedure on function $\phi(t) = a \cdot (\exp(b \cdot t) - 1)$ with $\epsilon_f =$ `1E-3` at $t = 0$. We use "−−" to report the cases where Moré-Wild heuristic fails. Subscript "MW" labels the results corresponding to Moré-Wild heuristic, and subscript "ada" labels the results corresponding to our adaptive procedure; δ is the relative error, and $\overline{\delta}$ is the worst-case relative error.

Problem	n	ϵ_f	Fixed Interval #Evals	Fixed Interval $\phi(x)-\phi^*$	Moré-Wild #Evals	Moré-Wild $\phi(x)-\phi^*$	Adaptive #Evals	Adaptive $\phi(x)-\phi^*$
AIRCRFTB	5	1e−1	1373	7.651	1457	4.777e−1	559	4.950e−1
ALLINITU	4	1e−1	656	2.767	972	3.708e−1	686	1.003e−1
ARWHEAD	100	1e−1	2513	2.027e−1	8784	3.040e−1	2811	2.458e−1
BARD	3	1e−1	650	8.311e−1	602	7.581e−2	670	3.942e−1
BDQRTIC	100	1e−1	2601	6.278	9213	5.902	5246	4.438
BIGGS3	3	1e−1	650	1.195	726	1.348	674	1.545
BIGGS5	5	1e−1	662	1.294	1457	1.319	696	1.359
BIGGS6	6	1e−1	668	4.026e−1	853	5.483e−1	693	6.766e−1
BOX2	2	1e−1	644	4.345e−2	709	4.095e−2	1975	3.406*
BOX3	3	1e−1	650	4.843e−2	1385	1.887e−1	676	7.527e−2
BRKMCC	2	1e−1	644	4.391e−2	709	5.850e−3	658	1.674e−1
BROWNAL	100	1e−1	2513	5.097e−2	10800	4.318e−2	3293	1.250e−2
BROWNDEN	4	1e−1	656	1.159e−1	1364	1.568e−1	1328	1.465e−1
CLIFF	2	1e−1	644	2.902e2	861	1.180e1	1593	3.385e−1
CRAGGLVY	100	1e−1	8011	1.850e2	207136	5.048e2*	11498	1.906e1
CUBE	2	1e−1	644	4.352e−2	6340	4.227*	659	8.670e−2
DENSCHND	3	1e−1	650	2.254e2	2062	2.627e−1	1018	9.226e−3
DENSCHNE	3	1e−1	650	1.151	2062	1.059	542	1.018
DIXMAANH	300	1e−1	3713	1.382e1	22922	1.392e1	7809	1.280e1
DQRTIC	100	1e−1	2513	1.743e2	26299	4.930	10912	4.944e−1
EDENSCH	36	1e−1	2129	1.674	4995	1.642	4624	3.352
EIGENALS	110	1e−1	2609	5.105e1	63118	1.496e1	11630	1.236e1
EIGENBLS	110	1e−1	5103	6.717	8306	1.292e1	5098	6.533
EIGENCLS	30	1e−1	1452	1.401	10899	1.218e1	4088	1.907
ENGVAL1	100	1e−1	2295	7.316e−1	610646	2.114e2*	4956	1.186
EXPFIT	2	1e−1	644	6.682e−1	31080	1.270e2*	659	6.881e−2
FLETCBV3	100	1e−1	2549	1.785e5	9488	1.785e5	21950	8.454e3
FLETCHBV	100	1e−1	40856	−1.175e9	135927	1.154e9	166981	5.696e9
FREUROTH	100	1e−1	2295	7.097e1	9658	5.756e1	4008	5.912e1
GENROSE	100	1e−1	4326	1.421e2	36745	1.321e2	5347	1.364e2
GULF	3	1e−1	650	6.713	1385	6.820	670	6.664
HAIRY	2	1e−1	644	9.325e1	1328	4.889e2*	1008	7.985e−2
HELIX	3	1e−1	650	7.601	745	7.591	671	7.656
NCB20B	100	1e−1	2295	4.793e−1	5531	2.000e−1	5393	1.280e−1
NONDIA	100	1e−1	2513	5.546e−1	10800	4.717e−1	5076	3.770e−1
NONDQUAR	100	1e−1	2613	8.483e−1	33658	7.457	5707	3.571e−1
OSBORNEA	5	1e−1	662	1.748e−1	52825	1.142e2*	2869	1.722e−1
OSBORNEB	11	1e−1	698	1.178	1014	3.021	769	1.851
PENALTY1	100	1e−1	8895	1.008e2	20136	1.131	12532	1.268
PFIT1LS	3	1e−1	17316	1.346e2	745	4.139	675	7.796
PFIT2LS	3	1e−1	20616	2.684e2	1385	2.350e1	1991	4.006
PFIT3LS	3	1e−1	20088	9.024e2	2126	2.933	1988	2.171
PFIT4LS	3	1e−1	21702	2.771e3	3839	2.704	1989	9.426
QUARTC	100	1e−1	2513	1.743e2	26299	4.930	10912	4.944e−1
SINEVAL	2	1e−1	644	5.576	566	4.712	664	5.554
SINQUAD	100	1e−1	4326	1.254e1	28209	3.331e1	5281	9.760
SISSER	2	1e−1	644	3.393e−2	2026	1.611e−1	664	7.452e−3
SPARSQUR	100	1e−1	4326	1.365	506948	6.928e1*	5592	1.209e−1
TOINTGSS	100	1e−1	2513	2.071e1	8784	1.467e1	4133	9.678
TQUARTIC	100	1e−1	2513	1.108	62958	3.128*	3391	7.331e−1
TRIDIA	100	1e−1	2513	8.478e1	18300	1.691e1	5677	3.935e1
WATSON	31	1e−1	1881	7.753e−1	4702	1.993	2174	2.873
WOODS	100	1e−1	10414	8.618e1	16987	2.907e2	5255	5.533e1
ZANGWIL2	2	1e−1	644	2.686e−2	1349	3.307e−2	657	3.852e−2

Table B.7: Total number of function evaluations used and final accuracy achieved by forward-difference L-BFGS method with different choices of the finite-difference interval.

Problem	n	ϵ_f	**Fixed Interval** #Evals	$\phi(x) - \phi^*$	**Moré-Wild** #Evals	$\phi(x) - \phi^*$	**Adaptive** #Evals	$\phi(x) - \phi^*$
AIRCRFTB	5	1e−3	1008	3.772e−1	817	3.313e−1	560	3.286e−1
ALLINITU	4	1e−3	656	2.354e−3	563	1.858e−3	686	1.172e−3
ARWHEAD	100	1e−3	2513	4.510e−2	7331	4.407e−2	4951	4.160e−2
BARD	3	1e−3	650	2.093e−3	602	3.467e−3	538	3.273e−3
BDQRTIC	100	1e−3	6017	9.588e−2	8733	1.151e−1	8551	9.804e−2
BIGGS3	3	1e−3	650	8.010e−4	712	1.375e−2	672	2.297e−2
BIGGS5	5	1e−3	662	1.813e−1	1457	1.261e−1	680	1.453e−1
BIGGS6	6	1e−3	668	2.904e−1	853	2.907e−1	563	2.909e−1
BOX2	2	1e−3	644	1.964e−4	709	2.624e−6	661	1.882e−5
BOX3	3	1e−3	650	9.556e−4	726	4.567e−4	672	3.568e−4
BRKMCC	2	1e−3	644	3.552e−3	566	3.872e−3	661	8.029e−4
BROWNAL	100	1e−3	2513	6.869e−4	10800	4.648e−4	3293	1.670e−4
BROWNDEN	4	1e−3	656	1.183e−3	706	1.912e−3	1327	8.187e−4
CLIFF	2	1e−3	644	2.902e2	28980	8.220e−1	1008	3.059e−4
CRAGGLVY	100	1e−3	7993	4.852	8619	1.046	7272	9.025e−1
CUBE	2	1e−3	644	4.201e−2	709	4.951e−2	659	4.381e−2
DENSCHND	3	1e−3	631	4.109	1385	1.170e−3	1018	9.852e−4
DENSCHNE	3	1e−3	650	9.994e−1	745	1.000	672	1.002
DIXMAANH	300	1e−3	7774	1.041	37580	7.456e−2	15794	9.375e−2
DQRTIC	100	1e−3	6609	4.832	21149	9.935e−3	11600	1.713e−2
EDENSCH	36	1e−3	1194	5.382e−2	6085	2.188e−2	3234	3.748e−2
EIGENALS	110	1e−3	4850	3.394e−1	22676	8.848e−2	8753	7.456e−2
EIGENBLS	110	1e−3	1932	2.084	21095	1.631	6825	1.740
EIGENCLS	30	1e−3	1875	7.554e−2	8449	7.032e−2	3187	8.686e−2
ENGVAL1	100	1e−3	2295	8.024e−2	9658	2.733e−2	4841	4.366e−2
EXPFIT	2	1e−3	644	6.733e−3	566	3.801e−4	663	3.592e−4
FLETCBV3	100	1e−3	2549	1.785e5	9488	1.785e5	29889	−1.181e2
FLETCHBV	100	1e−3	33825	−7.812e8	141125	6.878e7	62282	−6.875e8
FREUROTH	100	1e−3	4861	2.366e−1	9725	2.041e−2	8226	4.143e−2
GENROSE	100	1e−3	5779	1.109e2	17379	1.108e2	7561	1.110e2
GULF	3	1e−3	650	6.626	599	6.622	672	6.622
HAIRY	2	1e−3	644	7.979e−3	566	7.030e−4	853	3.304e−3
HELIX	3	1e−3	650	4.079e−3	1091	2.495e−4	671	3.746e−4
NCB20B	100	1e−3	2295	1.400e−2	8984	5.856e−3	6650	4.462e−3
NONDIA	100	1e−3	2513	4.911e−1	13295	4.862e−1	8550	4.612e−1
NONDQUAR	100	1e−3	5534	9.503e−2	29253	1.247e−2	8653	3.602e−2
OSBORNEA	5	1e−3	662	1.525e−1	54694	2.093	553	1.567e−1
OSBORNEB	11	1e−3	555	3.697e−1	2314	3.170e−1	766	3.115e−1
PENALTY1	100	1e−3	4326	2.200e1	14928	1.345e−3	12554	2.741e−4
PFIT1LS	3	1e−3	17244	7.787	2062	6.076e−2	1312	4.829e−2
PFIT2LS	3	1e−3	19169	1.147e2	2062	5.424e−2	1310	8.222e−2
PFIT3LS	3	1e−3	21645	6.633e2	2062	2.615e−2	1993	4.635e−2
PFIT4LS	3	1e−3	22928	2.305e3	2631	2.308e−1	1992	2.053e−1
QUARTC	100	1e−3	6609	4.832	21149	9.935e−3	11600	1.713e−2
SINEVAL	2	1e−3	644	1.557	2026	2.780e−1	1299	1.695e−1
SINQUAD	100	1e−3	4326	3.852e−2	19928	1.067e−1	5543	7.637e−2
SISSER	2	1e−3	644	2.069e−3	566	1.051e−3	664	2.094e−4
SPARSQUR	100	1e−3	6609	3.112e−2	14021	8.064e−2	6625	3.022e−3
TOINTGSS	100	1e−3	2513	2.198e−1	5071	1.026e−1	3730	6.133e−2
TQUARTIC	100	1e−3	2513	7.227e−1	7529	7.241e−1	4090	7.220e−1
TRIDIA	100	1e−3	7921	3.607e−1	28195	3.539e−1	9887	4.283e−1
WATSON	31	1e−3	1881	1.862e−1	10162	2.201e−2	2919	1.754e−1
WOODS	100	1e−3	11923	7.268e−1	10536	6.643	5493	6.472
ZANGWIL2	2	1e−3	644	8.620e−4	566	8.797e−4	657	6.881e−4

Table B.8: Total number of function evaluations used and final accuracy achieved by forward-difference L-BFGS method with different choices of the finite-difference interval.

			Fixed Interval		Moré-Wild		Adaptive	
Problem	n	ϵ_f	#Evals	$\phi(x) - \phi^*$	#Evals	$\phi(x) - \phi^*$	#Evals	$\phi(x) - \phi^*$
AIRCRFTB	5	1e−5	662	3.023e−1	1880	2.674e−4	2014	4.084e−4
ALLINITU	4	1e−5	656	1.835e−5	706	1.146e−5	682	6.468e−6
ARWHEAD	100	1e−5	3832	1.827e−4	11900	3.262e−4	7570	5.117e−4
BARD	3	1e−5	650	1.894e−3	602	1.878e−3	674	1.903e−3
BDQRTIC	100	1e−5	10225	3.227e−3	14968	2.305e−4	9860	5.248e−4
BIGGS3	3	1e−5	631	2.826e−4	712	2.404e−4	670	1.581e−4
BIGGS5	5	1e−5	1008	2.170e−2	3911	1.064e−4	2030	9.355e−3
BIGGS6	6	1e−5	1014	1.863e−3	2170	9.419e−5	1348	4.434e−4
BOX2	2	1e−5	644	3.493e−6	1349	8.092e−6	658	2.480e−6
BOX3	3	1e−5	650	7.493e−6	726	4.612e−6	672	1.114e−5
BRKMCC	2	1e−5	644	3.927e−7	566	1.244e−5	661	3.079e−5
BROWNAL	100	1e−5	2513	1.963e−5	9567	1.629e−5	4851	1.487e−5
BROWNDEN	4	1e−5	656	3.971e−6	1145	1.268e−5	1324	1.062e−5
CLIFF	2	1e−5	644	2.902e2	1349	2.275e−4	1008	2.238e−4
CRAGGLVY	100	1e−5	7993	8.012e−2	13711	8.641e−3	10935	1.884e−2
CUBE	2	1e−5	644	4.069e−2	709	1.148e−2	662	4.524e−3
DENSCHND	3	1e−5	996	1.720e−1	936	1.141e−5	1016	2.311e−5
DENSCHNE	3	1e−5	650	9.993e−1	563	9.993e−1	672	9.993e−1
DIXMAANH	300	1e−5	19344	1.035e−2	38056	5.615e−3	15708	8.080e−3
DQRTIC	100	1e−5	3486	2.687e−1	16434	1.667e−4	14109	1.864e−4
EDENSCH	36	1e−5	1194	4.513e−4	4840	5.362e−4	4624	7.831e−4
EIGENALS	110	1e−5	4921	1.929e−2	18980	1.337e−2	10030	1.353e−2
EIGENBLS	110	1e−5	3178	1.556	11368	1.553	46254	1.018e−2
EIGENCLS	30	1e−5	4370	5.560e−4	9380	8.302e−4	6373	5.415e−4
ENGVAL1	100	1e−5	2601	2.285e−4	8619	2.302e−4	7134	2.310e−4
EXPFIT	2	1e−5	644	4.595e−5	563	6.982e−6	666	6.814e−6
FLETCBV3	100	1e−5	4326	1.785e5	44623	1.785e5	100756	−1.304e2
FLETCHBV	100	1e−5	113423	3.562e9	315886	5.660e7	213807	1.765e9
FREUROTH	100	1e−5	4326	2.559e−3	10097	5.248e−4	7166	6.820e−4
GENROSE	100	1e−5	27714	8.446e−3	86415	5.674e−3	56271	1.067e−2
GULF	3	1e−5	650	6.596e−3	1091	2.275e−3	675	3.771e−3
HAIRY	2	1e−5	644	1.129e−4	1328	4.688e−6	1008	8.137e−6
HELIX	3	1e−5	650	7.422e−4	1385	8.376e−5	650	4.881e−5
NCB20B	100	1e−5	4790	4.231e−4	13270	3.565e−4	9691	3.061e−4
NONDIA	100	1e−5	4326	2.174e−4	21377	1.572e−2	8423	1.396e−2
NONDQUAR	100	1e−5	11923	9.225e−3	33549	1.510e−3	14104	1.836e−3
OSBORNEA	5	1e−5	853	7.139e−4	62478	5.603e−1	1334	1.215e−3
OSBORNEB	11	1e−5	1137	1.051e−1	4506	2.381e−3	2936	3.325e−3
PENALTY1	100	1e−5	4326	1.113e−1	20475	1.893e−4	12804	1.877e−4
PFIT1LS	3	1e−5	17338	1.609e−2	2062	2.921e−6	1312	2.320e−5
PFIT2LS	3	1e−5	18947	3.652e−2	2062	3.200e−3	1310	2.463e−3
PFIT3LS	3	1e−5	21038	2.034e−1	2062	2.364e−2	1999	2.822e−2
PFIT4LS	3	1e−5	22064	2.451e−1	2631	1.004e−1	1990	1.177e−1
QUARTC	100	1e−5	3486	2.687e−1	16434	1.667e−4	14109	1.864e−4
SINEVAL	2	1e−5	625	4.640e−3	2026	3.676e−3	1977	1.667e−4
SINQUAD	100	1e−5	2613	1.417e−3	11616	1.581e−4	5545	2.677e−4
SISSER	2	1e−5	644	7.894e−6	709	2.995e−7	658	3.218e−6
SPARSQUR	100	1e−5	8365	9.873e−4	14478	7.713e−5	8320	3.739e−5
TOINTGSS	100	1e−5	2513	2.193e−3	10274	6.054e−4	3729	1.167e−3
TQUARTIC	100	1e−5	4571	2.622e−1	9184	1.534e−1	10072	1.199e−1
TRIDIA	100	1e−5	9428	1.900e−3	45233	3.442e−3	19815	2.348e−3
WATSON	31	1e−5	3072	3.088e−3	14287	8.549e−3	6393	6.386e−3
WOODS	100	1e−5	7964	8.185e−3	25179	5.822e−3	13779	3.840e−3
ZANGWIL2	2	1e−5	644	8.737e−6	670	1.685e−5	657	7.597e−6

Table B.9: Total number of function evaluations used and final accuracy achieved by forward-difference L-BFGS method with different choices of the finite-difference interval.

Problem	n	ϵ_f	Fixed Interval #Evals	Fixed Interval $\phi(x)-\phi^*$	Moré-Wild #Evals	Moré-Wild $\phi(x)-\phi^*$	Adaptive #Evals	Adaptive $\phi(x)-\phi^*$
AIRCRFTB	5	1e−7	662	9.768e−5	2985	1.199e−5	1334	1.001e−5
ALLINITU	4	1e−7	656	2.172e−7	706	1.061e−7	543	7.839e−8
ARWHEAD	100	1e−7	4326	1.643e−6	11900	5.117e−7	5842	1.049e−6
BARD	3	1e−7	650	2.376e−5	706	7.175e−7	660	1.273e−7
BDQRTIC	100	1e−7	8011	1.802e−5	14968	5.277e−6	12535	4.346e−6
BIGGS3	3	1e−7	650	2.348e−7	726	3.990e−7	1018	1.386e−6
BIGGS5	5	1e−7	1008	6.378e−6	1614	1.637e−6	2869	3.891e−5
BIGGS6	6	1e−7	1014	−5.616e−3	2724	−5.595e−3	2023	−5.647e−3
BOX2	2	1e−7	644	1.726e−6	690	1.547e−7	658	1.865e−6
BOX3	3	1e−7	650	7.911e−7	602	6.639e−7	862	5.213e−7
BRKMCC	2	1e−7	644	2.465e−7	670	4.240e−7	661	9.632e−7
BROWNAL	100	1e−7	4326	2.571e−7	9567	1.334e−5	7259	5.981e−7
BROWNDEN	4	1e−7	656	3.507e−7	1145	1.214e−7	1324	1.468e−7
CLIFF	2	1e−7	644	2.902e2	861	2.198e−4	1300	2.192e−4
CRAGGLVY	100	1e−7	4678	1.570e−2	18164	1.153e−4	14124	7.253e−5
CUBE	2	1e−7	990	1.798e−4	1349	1.188e−4	662	1.404e−4
DENSCHND	3	1e−7	650	3.918e−2	1385	3.805e−8	1311	1.209e−7
DENSCHNE	3	1e−7	650	9.993e−1	1091	1.427e−7	1015	8.898e−8
DIXMAANH	300	1e−7	62344	1.773e−4	63812	2.194e−5	37104	2.753e−5
DQRTIC	100	1e−7	4571	5.119e−3	20136	3.509e−7	17121	4.377e−6
EDENSCH	36	1e−7	2165	2.262e−6	3168	3.776e−6	4624	3.756e−6
EIGENALS	110	1e−7	23459	1.291e−3	132863	2.204e−4	43487	2.893e−4
EIGENBLS	110	1e−7	38955	1.169e−3	182840	9.604e−4	92870	9.672e−4
EIGENCLS	30	1e−7	3906	1.173e−5	12408	9.139e−6	6345	8.639e−6
ENGVAL1	100	1e−7	4326	1.734e−6	8997	3.737e−6	6527	2.550e−6
EXPFIT	2	1e−7	644	3.707e−7	709	5.737e−8	666	4.705e−8
FLETCBV3	100	1e−7	39622	4.451e2	281142	1.666e2	208557	−5.161e1
FLETCHBV	100	1e−7	116172	−1.393e9	311603	2.365e9	212997	5.073e9
FREUROTH	100	1e−7	4678	2.155e−5	10097	8.099e−6	7165	6.707e−6
GENROSE	100	1e−7	27731	1.016e−4	83521	6.277e−5	56975	8.924e−5
GULF	3	1e−7	650	4.478e−3	1163	3.909e−3	1992	1.893e−5
HAIRY	2	1e−7	644	1.478e−6	670	1.107e−7	1008	2.080e−7
HELIX	3	1e−7	650	7.990e−6	1091	1.355e−5	674	1.967e−6
NCB20B	100	1e−7	12717	2.174e−5	34691	2.432e−5	21290	2.303e−5
NONDIA	100	1e−7	5779	2.912e−6	24808	1.134e−5	5826	8.625e−4
NONDQUAR	100	1e−7	8438	1.256e−3	102555	1.348e−4	44583	1.199e−4
OSBORNEA	5	1e−7	662	8.747e−5	47678	1.721e−1	2014	2.290e−5
OSBORNEB	11	1e−7	2015	1.342e−5	4408	1.260e−5	3626	2.063e−5
PENALTY1	100	1e−7	4326	6.722e−4	15829	1.869e−4	13984	1.870e−4
PFIT1LS	3	1e−7	15490	5.585e−4	2062	1.259e−6	1021	7.981e−6
PFIT2LS	3	1e−7	18475	3.194e−2	2126	1.834e−3	1314	1.441e−3
PFIT3LS	3	1e−7	19911	4.062e−2	2062	2.782e−2	1989	3.161e−2
PFIT4LS	3	1e−7	22339	2.817e−1	2631	1.171e−1	2842	1.668e−1
QUARTC	100	1e−7	4571	5.119e−3	20136	3.509e−7	17121	4.377e−6
SINEVAL	2	1e−7	625	7.770e−4	2026	9.623e−7	1300	5.182e−4
SINQUAD	100	1e−7	3400	1.629e−5	11616	1.584e−6	5543	6.333e−6
SISSER	2	1e−7	644	2.687e−7	709	1.365e−9	661	6.235e−8
SPARSQUR	100	1e−7	4861	2.668e−5	26138	1.350e−7	12562	2.022e−7
TOINTGSS	100	1e−7	2513	2.192e−5	10651	6.014e−6	3729	1.168e−5
TQUARTIC	100	1e−7	8895	1.645e−2	14021	8.050e−4	10062	1.413e−3
TRIDIA	100	1e−7	10619	2.499e−5	47586	2.839e−5	21947	3.486e−5
WATSON	31	1e−7	4264	1.161e−3	13658	1.172e−3	4585	1.854e−3
WOODS	100	1e−7	6609	5.611e−5	25179	4.354e−5	10111	9.748e−5
ZANGWIL2	2	1e−7	644	8.738e−8	670	1.683e−7	657	7.832e−8

Table B.10: Total number of function evaluations used and final accuracy achieved by forward-difference L-BFGS method with different choices of the finite-difference interval.

Problem	n	ϵ_f	**Fixed Interval** #Evals	$\phi(x) - \phi^*$	**Adaptive** #Evals	$\phi(x) - \phi^*$
AIRCRFTB	5	1e−1	602	4.455e−1	3074	4.451e−1
ALLINITU	4	1e−1	848	7.698e−1	1371	3.378e−3
ARWHEAD	100	1e−1	4001	4.137e−2	5021	5.157e−2
BARD	3	1e−1	669	2.616e−3	947	1.535e−2
BDQRTIC	100	1e−1	8594	2.047e−1	11555	1.918e−1
BIGGS3	3	1e−1	669	2.607e−1	1150	1.154e−2
BIGGS5	5	1e−1	693	8.829e−2	1125	3.740e−2
BIGGS6	6	1e−1	705	2.966e−1	1419	2.907e−1
BOX2	2	1e−1	657	1.013e−2	560	6.435e−3
BOX3	3	1e−1	669	4.700e−2	594	1.444e−3
BRKMCC	2	1e−1	657	4.767e−4	2418	1.983e−4
BROWNAL	100	1e−1	7175	1.321e−5	12140	1.331e−7
BROWNDEN	4	1e−1	681	4.778e−4	2247	1.814e−3
CLIFF	2	1e−1	657	2.902e2	4819	2.413e−2
CRAGGLVY	100	1e−1	11144	2.765	23987	6.089e−1
CUBE	2	1e−1	657	4.421e−2	3101	4.462e−2
DENSCHND	3	1e−1	669	5.418e−3	2310	8.228e−3
DENSCHNE	3	1e−1	669	1.015	576	1.025
DIXMAANH	300	1e−1	32019	7.301e−1	41575	4.225e−2
DQRTIC	100	1e−1	10891	4.910e−1	28135	7.032e−4
EDENSCH	36	1e−1	4217	5.638e−2	6835	7.687e−2
EIGENALS	110	1e−1	7295	2.049	20145	1.099e−1
EIGENBLS	110	1e−1	12096	2.326	13672	1.801
EIGENCLS	30	1e−1	3161	6.304e−1	8124	3.930e−1
ENGVAL1	100	1e−1	6704	1.473e−1	11258	2.414e−1
EXPFIT	2	1e−1	7418	2.456e1	5915	2.093e−2
FLETCBV3	100	1e−1	2546	1.785e5	136921	−1.561e2
FLETCHBV	100	1e−1	138282	6.330e9	380580	−1.079e9
FREUROTH	100	1e−1	6216	2.650e−1	14495	4.930e−2
GENROSE	100	1e−1	7084	1.116e2	28786	1.117e2
GULF	3	1e−1	669	6.621	2124	6.728
HAIRY	2	1e−1	657	1.261e−4	1483	6.896e−4
HELIX	3	1e−1	505	8.544e2	4900	2.476e1
NCB20B	100	1e−1	3028	2.230	12309	4.906e−2
NONDIA	100	1e−1	4782	4.941e−1	9226	4.929e−1
NONDQUAR	100	1e−1	7210	1.974e−1	14490	7.337e−2
OSBORNEA	5	1e−1	693	1.530e−1	2345	8.790e−1
OSBORNEB	11	1e−1	746	6.496e−1	3119	1.366
PENALTY1	100	1e−1	7210	3.404e−2	21785	1.847e−4
PFIT1LS	3	1e−1	17843	1.276e1	3706	8.023e−1
PFIT2LS	3	1e−1	3466	1.453e2	2827	5.360e−1
PFIT3LS	3	1e−1	26780	8.283e2	3807	1.767e−1
PFIT4LS	3	1e−1	27602	2.416e3	3727	3.979e−1
QUARTC	100	1e−1	10891	4.910e−1	28135	7.032e−4
SINEVAL	2	1e−1	501	5.547	3266	7.918
SINQUAD	100	1e−1	5279	2.818e−1	11289	3.892
SISSER	2	1e−1	657	9.267e−3	491	2.900e−5
SPARSQUR	100	1e−1	5345	6.181e−2	18495	1.016e−2
TOINTGSS	100	1e−1	5391	1.021e−2	8324	4.000e−9
TQUARTIC	100	1e−1	2605	8.336e−1	12297	7.326e−1
TRIDIA	100	1e−1	10830	1.695e−1	88893	5.839e−13
WATSON	31	1e−1	2880	2.177e−1	5954	1.693e−1
WOODS	100	1e−1	5279	6.439	21136	5.545
ZANGWIL2	2	1e−1	657	4.902e−4	1351	−9.999e−11

Table B.11: Total number of function evaluations used and final accuracy achieved by central-difference L-BFGS method with different choices of the finite-difference interval.

Problem	n	ϵ_f	**Fixed Interval** #Evals	$\phi(x) - \phi^*$	**Adaptive** #Evals	$\phi(x) - \phi^*$
AIRCRFTB	5	1e−3	594	4.451e−1	2306	4.451e−1
ALLINITU	4	1e−3	574	3.325e−3	1377	1.847e−5
ARWHEAD	100	1e−3	7175	5.590e−4	8379	6.125e−4
BARD	3	1e−3	669	1.905e−3	713	1.883e−3
BDQRTIC	100	1e−3	11220	3.667e−3	31078	4.314e−4
BIGGS3	3	1e−3	669	1.009e−3	1363	1.314e−4
BIGGS5	5	1e−3	1333	1.649e−2	1921	1.418e−2
BIGGS6	6	1e−3	705	5.181e−2	1844	−3.749e−3
BOX2	2	1e−3	657	1.287e−5	560	1.740e−4
BOX3	3	1e−3	669	3.555e−5	1351	5.017e−5
BRKMCC	2	1e−3	657	2.500e−6	556	3.658e−7
BROWNAL	100	1e−3	4782	8.022e−6	12030	2.446e−8
BROWNDEN	4	1e−3	681	2.721e−6	1525	1.803e−7
CLIFF	2	1e−3	657	2.902e2	3590	2.497e−4
CRAGGLVY	100	1e−3	11976	1.745e−2	30115	2.086e−3
CUBE	2	1e−3	657	4.239e−2	2418	4.580e−2
DENSCHND	3	1e−3	669	4.714e−3	1969	6.472e−4
DENSCHNE	3	1e−3	669	9.993e−1	1002	9.994e−1
DIXMAANH	300	1e−3	28496	1.428e−2	48610	3.693e−3
DQRTIC	100	1e−3	11015	1.313e−2	25201	2.425e−5
EDENSCH	36	1e−3	2128	2.700e−4	4599	2.101e−4
EIGENALS	110	1e−3	8516	4.539e−2	27468	1.303e−2
EIGENBLS	110	1e−3	47792	3.108e−2	25441	1.551
EIGENCLS	30	1e−3	4087	2.953e−3	10419	1.545e−3
ENGVAL1	100	1e−3	7210	1.658e−4	14954	4.131e−4
EXPFIT	2	1e−3	9839	3.021	1013	2.602e−4
FLETCBV3	100	1e−3	2546	1.785e5	198233	−1.561e2
FLETCHBV	100	1e−3	107444	−1.473e9	239215	3.847e9
FREUROTH	100	1e−3	7175	4.683e−4	14972	1.047e−4
GENROSE	100	1e−3	54819	2.202e−3	124290	9.891e−4
GULF	3	1e−3	1309	2.995e−3	1782	4.249e−3
HAIRY	2	1e−3	418	5.279e−6	4113	1.608e−7
HELIX	3	1e−3	432	8.522e2	2867	2.475e1
NCB20B	100	1e−3	5918	1.353e−1	14781	1.608e−3
NONDIA	100	1e−3	6475	1.027e−2	10213	1.263e−2
NONDQUAR	100	1e−3	7084	1.973e−2	28713	3.712e−3
OSBORNEA	5	1e−3	1333	2.881e−3	2237	8.790e−1
OSBORNEB	11	1e−3	2082	2.869e−1	3565	7.271e−2
PENALTY1	100	1e−3	7210	1.928e−4	20312	1.917e−4
PFIT1LS	3	1e−3	24764	1.500e1	2027	1.183e−3
PFIT2LS	3	1e−3	28415	1.973e2	2030	4.006e−3
PFIT3LS	3	1e−3	31502	1.018e3	3796	3.066e−2
PFIT4LS	3	1e−3	31607	3.352e3	3891	9.033e−2
QUARTC	100	1e−3	11015	1.313e−2	25201	2.425e−5
SINEVAL	2	1e−3	588	5.327	3436	4.873e−5
SINQUAD	100	1e−3	5391	6.603e−4	14970	1.277e−4
SISSER	2	1e−3	657	9.026e−6	684	5.497e−5
SPARSQUR	100	1e−3	5542	2.341e−3	19979	2.489e−5
TOINTGSS	100	1e−3	2896	1.267e−5	9505	4.000e−9
TQUARTIC	100	1e−3	19220	2.492e−1	8491	7.221e−1
TRIDIA	100	1e−3	17514	3.205e−5	92231	4.283e−15
WATSON	31	1e−3	3173	2.462e−2	9834	2.186e−3
WOODS	100	1e−3	11220	3.320e−3	23074	6.287e−4
ZANGWIL2	2	1e−3	657	9.871e−7	598	−9.999e−11

Table B.12: Total number of function evaluations used and final accuracy achieved by central-difference L-BFGS method with different choices of the finite-difference interval.

Problem	n	ϵ_f	**Fixed Interval** #Evals	$\phi(x) - \phi^*$	**Adaptive** #Evals	$\phi(x) - \phi^*$
AIRCRFTB	5	1e−5	612	4.451e−1	2558	3.293e−1
ALLINITU	4	1e−5	681	7.277e−6	2035	7.767e−9
ARWHEAD	100	1e−5	7487	4.138e−6	7477	1.108e−6
BARD	3	1e−5	669	1.867e−3	1500	1.678e−7
BDQRTIC	100	1e−5	11220	1.539e−6	19106	4.834e−6
BIGGS3	3	1e−5	669	4.019e−6	1088	3.251e−7
BIGGS5	5	1e−5	1756	4.920e−5	3617	1.465e−5
BIGGS6	6	1e−5	2358	−5.228e−3	4142	−5.625e−3
BOX2	2	1e−5	657	2.137e−6	1327	1.773e−6
BOX3	3	1e−5	669	6.746e−7	1349	7.643e−7
BRKMCC	2	1e−5	437	5.850e−10	558	1.300e−9
BROWNAL	100	1e−5	3508	1.384e−7	19759	1.759e−16
BROWNDEN	4	1e−5	681	7.291e−9	2060	−1.455e−11
CLIFF	2	1e−5	657	2.902e2	4442	2.272e−4
CRAGGLVY	100	1e−5	17145	3.635e−5	32645	5.568e−6
CUBE	2	1e−5	657	2.696e−7	2510	3.013e−6
DENSCHND	3	1e−5	1015	3.165e−4	1878	2.382e−6
DENSCHNE	3	1e−5	669	9.993e−1	1073	7.559e−8
DIXMAANH	300	1e−5	45747	8.540e−5	235332	1.565e−10
DQRTIC	100	1e−5	13738	1.501e−4	28793	1.977e−6
EDENSCH	36	1e−5	2382	3.038e−7	6843	3.233e−7
EIGENALS	110	1e−5	27694	1.556e−3	93131	2.888e−4
EIGENBLS	110	1e−5	86034	9.991e−4	167257	1.114e−3
EIGENCLS	30	1e−5	5297	3.243e−5	15418	5.660e−6
ENGVAL1	100	1e−5	5244	1.653e−6	12432	7.819e−7
EXPFIT	2	1e−5	2953	1.015e−2	1097	3.382e−7
FLETCBV3	100	1e−5	61879	−7.404e1	405665	−1.561e2
FLETCHBV	100	1e−5	69727	1.786e9	168055	−3.029e8
FREUROTH	100	1e−5	6618	1.671e−6	17360	1.873e−7
GENROSE	100	1e−5	53552	5.526e−6	119843	3.104e−6
GULF	3	1e−5	669	4.155e−3	2094	1.869e−6
HAIRY	2	1e−5	657	3.882e−9	2015	2.389e−11
HELIX	3	1e−5	445	8.521e2	5026	2.475e1
NCB20B	100	1e−5	14030	1.578e−3	37332	7.086e−5
NONDIA	100	1e−5	7210	1.054e−5	13740	1.839e−5
NONDQUAR	100	1e−5	18545	9.490e−4	55137	2.686e−4
OSBORNEA	5	1e−5	693	9.012e−4	2295	8.790e−1
OSBORNEB	11	1e−5	3861	4.164e−4	5596	2.698e−5
PENALTY1	100	1e−5	9496	1.868e−4	22443	1.875e−4
PFIT1LS	3	1e−5	26200	5.375	2883	8.388e−6
PFIT2LS	3	1e−5	29030	1.010e2	2008	2.499e−3
PFIT3LS	3	1e−5	30940	6.034e2	2339	4.427e−2
PFIT4LS	3	1e−5	31645	2.140e3	6937	7.936e−2
QUARTC	100	1e−5	13738	1.501e−4	28793	1.977e−6
SINEVAL	2	1e−5	18967	3.956	3089	2.075e−7
SINQUAD	100	1e−5	5127	1.373e−6	16276	4.398e−7
SISSER	2	1e−5	657	3.088e−7	1337	5.212e−11
SPARSQUR	100	1e−5	7464	2.220e−5	19032	5.543e−6
TOINTGSS	100	1e−5	8221	2.253e−8	8385	4.000e−9
TQUARTIC	100	1e−5	12669	8.689e−4	15299	3.231e−4
TRIDIA	100	1e−5	20802	5.305e−7	107193	1.824e−17
WATSON	31	1e−5	7718	1.087e−3	16253	1.209e−3
WOODS	100	1e−5	9234	2.150e−5	24007	3.869e−6
ZANGWIL2	2	1e−5	657	2.019e−9	1347	−1.000e−10

Table B.13: Total number of function evaluations used and final accuracy achieved by central-difference L-BFGS method with different choices of the finite-difference interval.

			Fixed Interval		Adaptive	
Problem	n	ϵ_f	#Evals	$\phi(x) - \phi^*$	#Evals	$\phi(x) - \phi^*$
AIRCRFTB	5	1e−7	1585	6.912e−12	2929	2.059e−22
ALLINITU	4	1e−7	463	1.644e−8	1493	3.211e−10
ARWHEAD	100	1e−7	5391	1.192e−8	8298	2.636e−9
BARD	3	1e−7	669	2.756e−9	1357	4.989e−9
BDQRTIC	100	1e−7	11220	1.885e−8	27324	6.486e−9
BIGGS3	3	1e−7	669	1.115e−8	1357	5.031e−10
BIGGS5	5	1e−7	1530	2.281e−8	3911	4.135e−9
BIGGS6	6	1e−7	2022	−5.642e−3	5582	−5.649e−3
BOX2	2	1e−7	657	3.579e−9	678	1.419e−9
BOX3	3	1e−7	669	6.337e−7	1092	6.848e−10
BRKMCC	2	1e−7	657	2.043e−10	680	2.005e−10
BROWNAL	100	1e−7	3799	2.783e−8	24944	6.904e−19
BROWNDEN	4	1e−7	681	−3.638e−10	1695	−3.929e−10
CLIFF	2	1e−7	657	5.621e−1	10094	2.902e2
CRAGGLVY	100	1e−7	18747	5.581e−7	39527	3.028e−8
CUBE	2	1e−7	657	1.452e−7	1905	7.199e−9
DENSCHND	3	1e−7	1309	1.440e−5	2057	8.977e−8
DENSCHNE	3	1e−7	669	3.754e−11	2034	2.051e−10
DIXMAANH	300	1e−7	52791	4.060e−7	394134	0.000
DQRTIC	100	1e−7	14367	2.836e−6	29677	2.399e−9
EDENSCH	36	1e−7	2434	1.106e−9	6835	8.345e−10
EIGENALS	110	1e−7	86646	7.971e−6	194737	2.282e−6
EIGENBLS	110	1e−7	194875	1.129e−6	257660	9.239e−4
EIGENCLS	30	1e−7	7754	1.060e−7	17966	7.350e−8
ENGVAL1	100	1e−7	7210	5.117e−9	13746	7.956e−9
EXPFIT	2	1e−7	3032	6.553e−5	1341	8.708e−10
FLETCBV3	100	1e−7	101241	−2.737e1	693140	−8.437e1
FLETCHBV	100	1e−7	106670	1.635e9	410157	−1.297e9
FREUROTH	100	1e−7	6382	−3.050e−9	14775	−5.215e−9
GENROSE	100	1e−7	53767	2.215e−8	118006	2.538e−8
GULF	3	1e−7	1504	3.522e−7	2098	5.298e−8
HAIRY	2	1e−7	657	1.723e−12	1511	0.000
HELIX	3	1e−7	413	8.521e2	2825	5.755e−13
NCB20B	100	1e−7	31569	2.837e−5	73674	1.549e−5
NONDIA	100	1e−7	6618	6.417e−8	14890	6.216e−8
NONDQUAR	100	1e−7	77500	5.554e−5	239505	1.867e−5
OSBORNEA	5	1e−7	1333	2.245e−5	2162	1.562e−1
OSBORNEB	11	1e−7	2933	1.612e−8	6229	6.560e−9
PENALTY1	100	1e−7	8565	1.868e−4	76226	2.507e−6
PFIT1LS	3	1e−7	26224	4.862e−4	4269	2.294e−5
PFIT2LS	3	1e−7	28717	1.295e−2	14332	2.756e−5
PFIT3LS	3	1e−7	28777	5.848e−2	22899	1.676e−5
PFIT4LS	3	1e−7	29320	2.991e−1	25011	1.775e−5
QUARTC	100	1e−7	14367	2.836e−6	29677	2.399e−9
SINEVAL	2	1e−7	1722	1.177e−11	1879	1.601e−10
SINQUAD	100	1e−7	5544	5.534e−9	16166	3.215e−9
SISSER	2	1e−7	657	2.663e−8	703	6.479e−9
SPARSQUR	100	1e−7	10891	3.396e−7	24709	2.323e−9
TOINTGSS	100	1e−7	4087	4.042e−9	6671	4.000e−9
TQUARTIC	100	1e−7	9496	3.400e−7	19028	3.207e−7
TRIDIA	100	1e−7	30055	2.966e−10	114846	1.242e−19
WATSON	31	1e−7	8296	1.016e−4	21230	1.002e−4
WOODS	100	1e−7	8594	5.693e−7	20454	7.795e−8
ZANGWIL2	2	1e−7	657	−9.543e−11	1345	−1.000e−10

Table B.14: Total number of function evaluations used and final accuracy achieved by central-difference L-BFGS method with different choices of the finite-difference interval.