

A Randomized Bracketing Method for Derivative-Free Root Finding with Uniform Spacing Contraction

Dinesh Kumar

University of Delhi

Sudesh K. Srivastav

ssrivast@tulane.edu

Tulane University

Research Article

Keywords: randomized bracketing, interval contraction, derivative-free root finding, uniform spacings, order statistics, parallel computation

Posted Date: July 1st, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10070193/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

A Randomized Bracketing Method for Derivative-Free Root Finding with Uniform Spacing Contraction

Dinesh Kumar · Sudesh K. Srivastav

Abstract We introduce a randomized bracket-preserving interval-contraction framework for the derivative-free solution of nonlinear equations $f(x) = 0$ on a bounded interval. At each iteration, m interior points are sampled uniformly from the current bracket, and the next interval is chosen as the shortest admissible subinterval according to endpoint sign information, producing a nested sequence of random root-containing intervals and a midpoint estimator.

For the basic two-sample method and its general m -sample extension, we prove root preservation, derive explicit spacing-based upper bounds on the expected contraction factor using the geometry of uniform order statistics, establish convergence in probability of the midpoint estimator, and show almost sure interval contraction under a mild logarithmic moment condition. The contraction mechanism is governed by the random spacings induced by the ordered sample, linking the framework to classical results on maximal spacings and order statistics.

The tunable parameter m provides a natural contraction–cost tradeoff: larger m improves contraction but increases the number of function evaluations under serial execution, while it reduces the number of update stages under an idealized batched model. Numerical experiments confirm geometric interval decay and indicate that the proposed framework offers a robust bracket-preserving option for black-box settings in which function evaluations are expensive, mildly noisy, or naturally parallelizable.

Keywords: randomized bracketing, interval contraction, derivative-free root finding, uniform spacings, order statistics, parallel computation

2020 Mathematics Subject Classification: 65H05, 65C05, 60F05, 62G30

Dinesh Kumar
Department of Mathematics, Dyal Singh College, University of Delhi, India
E-mail: dkvatsayana@dsc.du.ac.in

Sudesh K. Srivastav (corresponding author)
Department of Biostatistics and Data Science, Tulane University, New Orleans, USA
E-mail: ssrivas@tulane.edu

1 Introduction

The numerical solution of nonlinear equations

$$f(x) = 0$$

is a fundamental problem in scientific computing and numerical analysis. Classical methods such as bisection, secant iteration, Newton’s method, and Brent-type hybrids offer different tradeoffs between robustness, efficiency, and regularity requirements [2, 14, 10, 3]. Bracketing methods are typically reliable but conservative, whereas open methods can converge rapidly when well initialized but may be sensitive to local geometry, poor starting values, or unavailable derivative information. These tradeoffs become especially important in modern black-box settings, where one may seek reliable bracketing, robustness to evaluation irregularity, and the ability to exploit multiple concurrent function evaluations.

In this paper we introduce a randomized bracket-preserving framework for derivative-free root finding on a bounded interval. Starting from an initial sign-changing bracket, the algorithm samples a finite number of interior points uniformly at each iteration and updates the interval to the shortest admissible subinterval determined by endpoint sign information. The resulting procedures combine guaranteed bracket preservation with a spacing-driven randomized contraction mechanism that admits explicit probabilistic analysis.

The framework is not intended to replace mature deterministic solvers in smooth serial settings. Instead, it targets derivative-free black-box regimes in which guaranteed bracketing, robustness, and batched evaluation are more important than aggressive extrapolative acceleration. Under exact function evaluations the method preserves a valid bracket at every iteration, while its randomized sampling admits a direct analysis in terms of uniform spacings.

Conceptually, the method is best viewed not as a randomized variant of deterministic bisection, but as a bracketing framework in which contraction is governed by the geometry of random spacings. The ordered interior sample partitions the current interval into random subintervals, and the update is controlled by the local spacing configuration containing the root. This links the contraction mechanism directly to classical results on order statistics and maximal spacings [11, 4, 5, 6, 13].

Conceptual and methodological perspective The main conceptual contribution is a bracket-preserving interval-contraction framework whose per-step behavior is governed by the geometry of random spacings. In contrast to deterministic interval subdivision, contraction arises from random spacings generated by uniform interior sampling. At the same time, unlike stochastic approximation or probabilistic bisection, the procedure retains an explicit admissible bracket at every iteration under exact evaluations. This combination enables direct probabilistic analysis through order-statistic spacing theory while preserving the structural robustness of classical bracketing methods.

Methodologically, the framework is characterized by four key ingredients: (i) deterministic bracket preservation under exact evaluations; (ii) randomized contraction driven by uniform interior sampling; (iii) explicit spacing-based analysis linking contraction to order statistics and maximal-spacing theory; and (iv) a tunable parameter m that

governs the tradeoff between contraction strength and per-iteration cost (serial versus batched).

Positioning and scope The intended niche is derivative-free black-box root finding where explicit bracket preservation matters and multiple interior evaluations can be performed concurrently. This leads to two natural cost models: serial (total function evaluations) and idealized parallel/batched (number of update stages). The resulting contraction–cost tradeoff is a central theme of the paper.

Each RIC- m update requires m newly sampled interior evaluations together with sorting and selection operations whose dominant arithmetic cost is $O(m \log m)$. The number of update stages required for a prescribed accuracy is governed by the spacing-based contraction bounds developed later. The algorithms are simple to implement, require only function evaluations and sign comparisons, and integrate naturally into existing bracketing-based software.

Summary of contributions The main contributions are as follows:

1. A randomized bracket-preserving interval-contraction framework yielding a family of derivative-free algorithms.
2. An explicit connection between randomized contraction, order statistics, and maximal-spacing theory.
3. For the two-sample method, we prove root preservation, expected interval contraction, and convergence in probability of the midpoint estimator, and we establish almost sure interval contraction in the spacing model considered.
4. Numerical experiments clarifying the contraction–cost tradeoff under serial and batched models, with comparisons to standard deterministic solvers.

To the best of our knowledge, this is the first bracket-preserving root-finding framework whose contraction analysis is formulated explicitly through maximal-spacing theory for uniform order statistics, rather than through deterministic interval subdivision alone. The resulting perspective links interval-based numerical root finding with classical spacing theory and yields explicit probabilistic contraction bounds for a family of randomized bracketing algorithms.

The theoretical analysis uses the transparent setting of a continuous, strictly monotone function on a bounded interval with a valid initial bracket. The numerical study primarily evaluates the method within this regime while also exploring non-monotone and noisy cases.

Figure 1 summarizes the workflow and the spacing-based contraction mechanism. Throughout, the emphasis is on the randomized bracketing framework and its probabilistic analysis.

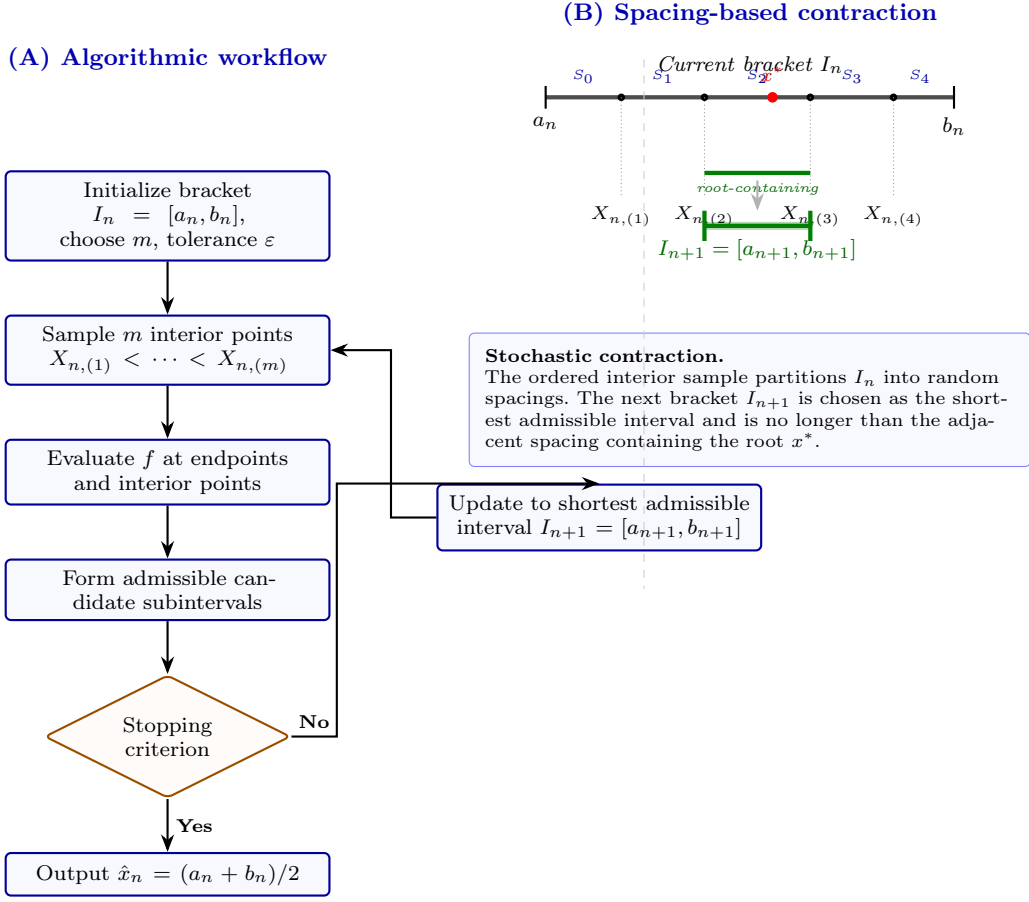


Fig. 1: Schematic of the randomized interval-contraction (RIC) method. Panel (A) shows one update cycle of the algorithm. Panel (B) illustrates the spacing-based contraction mechanism with $m = 4$: the ordered interior sample partitions the current bracket into random spacings, and the next bracket is selected as a shortest admissible interval controlled by the local spacing containing the root. The panel also highlights the distinction between the root-containing spacing, which governs the actual update, and the maximal spacing, which is used in the theoretical contraction bound.

Relation to deterministic bisection

It is important to distinguish the proposed framework from classical deterministic bisection. Both methods maintain a valid bracketing interval under exact function evaluations, but their update mechanisms and analyses differ. In bisection, the interval is halved deterministically at each step, leading to a fixed contraction factor of $1/2$ and a purely deterministic convergence theory. In the randomized interval-contraction (RIC) framework, by contrast, interval refinement is driven by random spacings generated by uniformly sampled interior points, and contraction is analyzed probabilistically through

order-statistic and maximal-spacing theory rather than through a fixed subdivision rule.

For ease of reference, Table 1 summarizes the main structural differences.

Table 1: Conceptual comparison between deterministic bisection and the randomized interval-contraction (RIC) framework.

Property	Bisection	RIC framework
Update rule	Deterministic midpoint split	Randomized, spacing-based
Contraction	Fixed factor 1/2	Distribution-dependent B_m
Analysis	Deterministic subdivision	Order statistics and spacings
Parallelization	One new evaluation per step	m concurrent evaluations
Convergence	Deterministic	Probabilistic (in prob., a.s.)

The purpose of this comparison is not to suggest that the RIC framework dominates deterministic bisection in classical serial environments, but to highlight that the two methods are based on different contraction mechanisms and therefore admit different analytical descriptions. The contribution of the present work lies in showing that interval contraction can be studied through spacing geometry, leading to explicit probabilistic contraction bounds and convergence results.

The remainder of the paper is organized as follows. Section 2 introduces the problem setup and algorithm. Section 3 presents the main structural and convergence results for the two-sample method. Section 4 develops the multi-sample extension and its spacing-based analysis. Section 5 reports numerical experiments and discusses performance under different computational cost models.

2 Problem Setup and Algorithm

We consider the problem of solving

$$f(x) = 0$$

for a continuous real-valued function on a bounded interval. The framework developed here is a randomized bracketing method: given a current root-containing interval, the algorithm samples interior points uniformly, forms a canonical family of candidate subintervals, and updates to the shortest admissible interval determined by endpoint sign information.

2.1 Standing assumptions

The main theoretical results are established under the following assumption.

Assumption 1 (Strict monotonicity and initial bracket) *The function $f : [a_0, b_0] \rightarrow \mathbb{R}$ is continuous and strictly monotone on $[a_0, b_0]$, and*

$$f(a_0)f(b_0) < 0.$$

Under Assumption 1, the equation $f(x) = 0$ has a unique root

$$x^* \in (a_0, b_0).$$

Moreover, if $[c, d] \subseteq [a_0, b_0]$ satisfies

$$c \leq x^* \leq d,$$

then continuity and monotonicity imply

$$f(c)f(d) \leq 0.$$

This sign property underlies the admissibility criterion used throughout the paper: an interval $[u, v] \subseteq [a_0, b_0]$ is called *admissible* whenever

$$f(u)f(v) \leq 0.$$

For a generic bracket $I_n = [a_n, b_n]$, we write

$$L_n = b_n - a_n, \quad \hat{x}_n = \frac{a_n + b_n}{2}$$

for its length and midpoint estimator, respectively.

Remark 1 (On the monotonicity assumption) The strict monotonicity assumption is imposed primarily to ensure uniqueness of the root and to make the endpoint-sign admissibility criterion especially transparent. It provides a clean setting in which the randomized interval process can be analyzed without additional complications arising from multiple roots or non-monotone behavior.

The underlying sampling-and-selection mechanism can, in principle, be adapted to broader settings by tracking sign changes across sampled subintervals or by incorporating additional screening rules. Such extensions require a different analytical treatment and are beyond the scope of the present paper. In settings with multiple roots or nonmonotone behavior, the same sampling-and-selection mechanism could in principle be combined with local sign-change detection to maintain multiple admissible brackets or to track a selected root within a specified subinterval. Such extensions would require additional bookkeeping and selection rules, but the basic spacing-driven update mechanism would remain applicable at the algorithmic level.

3 Two-sample randomized interval contraction

We first define the basic two-sample update.

Definition 1 (Two-sample randomized interval-contraction algorithm) Let $I_n = [a_n, b_n]$ be the current bracket. Draw

$$X_{n,1}, X_{n,2} \stackrel{\text{i.i.d.}}{\sim} \text{Unif}(a_n, b_n),$$

and let

$$X_{n,(1)} \leq X_{n,(2)}$$

denote the associated order statistics. Define the canonical candidate family

$$\mathcal{C}_n = \{[a_n, X_{n,(1)}], [a_n, X_{n,(2)}], [X_{n,(1)}, X_{n,(2)}], [X_{n,(1)}, b_n], [X_{n,(2)}, b_n]\}.$$

Among all admissible intervals in $\mathcal{C}_n$, select one having minimal length and denote it by

$$I_{n+1} = [a_{n+1}, b_{n+1}].$$

If multiple admissible intervals attain the minimal length, any one of them may be selected. This tie-breaking convention does not affect the root-preservation or contraction results below.

Although $\mathcal{C}_n$ includes both adjacent and non-adjacent intervals determined by the sampled points and endpoints, the basic contraction mechanism is governed by the three adjacent spacing intervals induced by the ordered sample. In particular, the subinterval containing the root is one of these adjacent spacings.

Lemma 1 (Existence of an admissible candidate) *Under Assumption 1, at each iteration of Definition 1, the candidate family $\mathcal{C}_n$ contains at least one admissible interval.*

Proof Since $x^* \in [a_n, b_n]$, the sampled points $X_{n,(1)} \leq X_{n,(2)}$ partition the interval into the three subintervals

$$[a_n, X_{n,(1)}], \quad [X_{n,(1)}, X_{n,(2)}], \quad [X_{n,(2)}, b_n].$$

Exactly one of these subintervals contains x^* . By the sign property following Assumption 1, the endpoints of that subinterval satisfy $f(u)f(v) \leq 0$, and hence the subinterval is admissible. Therefore $\mathcal{C}_n$ contains at least one admissible interval.

Consequently, the update is well defined at every iteration under Assumption 1. Since the selected interval is admissible, it contains the root x^* , and the resulting interval sequence is nested:

$$I_{n+1} \subseteq I_n \quad \text{for all } n \geq 0.$$

Remark 2 (Markovian viewpoint) The interval process $\{I_n\}_{n \geq 0}$ may be viewed as a time-homogeneous Markov chain on the class of admissible root-containing brackets. The update rule therefore defines a stochastic interval-contraction mechanism whose one-step behavior is governed by the random spacing configuration induced by the sampled interior points. This perspective helps organize the convergence arguments developed in the next sections.

3.1 Algorithmic summary

For convenience, we summarize the two-sample randomized interval-contraction method in procedural form.

Algorithm 1 Two-sample randomized interval contraction

```

1: Input: continuous strictly monotone  $f$ , initial bracket  $[a_0, b_0]$  with  $f(a_0)f(b_0) < 0$ , tolerance
    $\varepsilon > 0$ , maximum iteration count  $N_{\max}$ 
2: Initialize:  $n \leftarrow 0$ ,  $I_n = [a_n, b_n] \leftarrow [a_0, b_0]$ 
3: while  $b_n - a_n > \varepsilon$  and  $n < N_{\max}$  do
4:   Draw  $X_{n,1}, X_{n,2} \stackrel{\text{i.i.d.}}{\sim} \text{Unif}(a_n, b_n)$ 
5:   Sort to obtain  $X_{n,(1)} \leq X_{n,(2)}$ 
6:   Form the candidate family  $\mathcal{C}_n$  as in Definition 1
7:   Select a shortest admissible interval  $[u, v] \in \mathcal{C}_n$ 
8:   Set  $I_{n+1} = [a_{n+1}, b_{n+1}] \leftarrow [u, v]$ 
9:    $n \leftarrow n + 1$ 
10: end while
11: Output: final bracket  $I_n = [a_n, b_n]$  and midpoint estimate  $\hat{x}_n = (a_n + b_n)/2$ 

```

3.2 Implementation remarks

In practical implementations, function values at bracket endpoints can be cached and reused across iterations, so that each update requires only the newly sampled interior evaluations. For the two-sample method, this amounts to two new function evaluations per iteration. More generally, for the m -sample extension introduced later, the per-iteration evaluation cost is m new interior evaluations, together with sorting and interval-selection overhead.

Although the admissible candidate family is defined abstractly for analytical clarity, a practical implementation need not inspect all candidate intervals naively. Once the sampled points are ordered, sign information can be used to localize the root and restrict attention to a much smaller set of nearby admissible candidates.

Remark 3 (Practical localization) Let

$$Y_{n,0} = a_n, \quad Y_{n,1} = X_{n,(1)}, \quad Y_{n,2} = X_{n,(2)}, \quad Y_{n,3} = b_n.$$

Under Assumption 1, there is a unique index $j \in \{0, 1, 2\}$ such that

$$f(Y_{n,j})f(Y_{n,j+1}) \leq 0.$$

Hence a practical implementation may first identify this local sign-change location and then restrict interval selection to admissible intervals near that location, rather than scanning the full candidate family $\mathcal{C}_n$.

In black-box settings where function evaluation dominates arithmetic overhead, this localization makes the method computationally more attractive than the full candidate-family definition may initially suggest. Thus, each iteration consists of sampling, localizing the root through sign information, and shrinking the bracket according to the induced spacing structure. This viewpoint forms the basis for the contraction analysis developed in the next section.

Before turning to the general m -sample extension, it is useful to record the explicit spacing benchmark for the basic two-sample case.

Corollary 1 (Two-sample contraction benchmark) *For the basic two-sample procedure ($m = 2$), the spacing-based contraction benchmark can be written explicitly as*

$$B_2 = \mathbb{E}[\max\{S_0, S_1, S_2\}] = \frac{11}{18} \approx 0.6111,$$

where (S_0, S_1, S_2) denotes the normalized spacing vector induced by two independent uniform draws on $[0, 1]$. In particular, the expected interval contraction for the two-sample method is controlled by a concrete constant strictly below one.

4 Multi-sample extension

The multi-sample extension makes the spacing-based contraction mechanism particularly transparent. Instead of sampling two interior points at each iteration, we allow $m \geq 2$ sampled points, which partition the current bracket into $m + 1$ random spacings. The update selects the shortest admissible subinterval, yielding a family of randomized bracketing schemes in which larger m can improve contraction at the cost of increased per-iteration sampling effort.

Throughout this section, Assumption 1 remains in force. In particular, the equation $f(x) = 0$ has a unique root $x^* \in (a_0, b_0)$. For each $n \geq 0$, let

$$I_n^{(m)} = [a_n^{(m)}, b_n^{(m)}], \quad L_n^{(m)} = b_n^{(m)} - a_n^{(m)}, \quad \hat{x}_n^{(m)} = \frac{a_n^{(m)} + b_n^{(m)}}{2}$$

denote the current bracket, its length, and its midpoint estimator for the m -sample method.

4.1 Definition of the m -sample algorithm

Definition 2 (m -sample randomized interval-contraction algorithm) Fix an integer $m \geq 2$. Given the current bracket

$$I_n^{(m)} = [a_n^{(m)}, b_n^{(m)}],$$

draw

$$X_{n,1}, \dots, X_{n,m} \stackrel{\text{i.i.d.}}{\sim} \text{Unif}(a_n^{(m)}, b_n^{(m)}),$$

and let

$$X_{n,(1)} \leq \dots \leq X_{n,(m)}$$

denote the corresponding order statistics. Define the augmented ordered point set

$$\mathcal{P}_n^{(m)} = \{a_n^{(m)}, X_{n,(1)}, \dots, X_{n,(m)}, b_n^{(m)}\}.$$

The candidate family $\mathcal{C}_n^{(m)}$ consists of all intervals formed by ordered pairs of distinct points from $\mathcal{P}_n^{(m)}$, excluding the full current interval $I_n^{(m)}$. Among all admissible intervals in $\mathcal{C}_n^{(m)}$, select one having minimal length and denote it by

$$I_{n+1}^{(m)} = [a_{n+1}^{(m)}, b_{n+1}^{(m)}].$$

If multiple admissible intervals attain the minimal length, any one of them may be selected.

Implementation note. The full candidate family $\mathcal{C}_n^{(m)}$ is introduced for conceptual clarity and to state the admissibility principle independently of implementation. In practice, however, interval selection may be restricted to a much smaller collection of locally admissible candidates identified through sign localization, thereby avoiding a naive examination of all $O(m^2)$ intervals.

Remark 4 (Local implementation under monotonicity) Under Assumption 1, the ordered sampled points and the two endpoints partition the current bracket into $m + 1$ adjacent spacing intervals, and the unique root x^* lies in exactly one of these intervals. Hence the shortest admissible interval selected from $\mathcal{C}_n^{(m)}$ is controlled by the local spacing containing the sign change. Therefore, in the strictly monotone setting, a practical implementation may first locate the adjacent sign-changing spacing and then restrict interval selection to local admissible candidates near that spacing, rather than scanning all $O(m^2)$ intervals. The global candidate-family definition is retained for conceptual clarity and for possible extensions beyond the strictly monotone case.

Remark 5 (Implementation and structural interpretation) The candidate family $\mathcal{C}_n^{(m)}$ is defined using all ordered pairs of sampled points and endpoints, but the adjacent spacing structure induced by the ordered sample plays the central role in the contraction analysis. The root belongs to exactly one of the $m + 1$ adjacent spacing intervals, and the selected admissible interval is necessarily no longer than that root-containing spacing. The fuller candidate-family definition is retained primarily for conceptual clarity.

Algorithm 2 m -sample randomized interval-contraction (RIC- m)

- 1: **Input:** continuous function f , initial bracket $I_0^{(m)} = [a_0^{(m)}, b_0^{(m)}]$ with $f(a_0^{(m)})f(b_0^{(m)}) < 0$, sample size $m \geq 2$, tolerance $\varepsilon > 0$, maximum iteration count $N_{\max}$
 - 2: **Initialize:** $n \leftarrow 0$
 - 3: **while** $b_n^{(m)} - a_n^{(m)} > \varepsilon$ **and** $n < N_{\max}$ **do**
 - 4: Draw $X_{n,1}, \dots, X_{n,m} \stackrel{\text{i.i.d.}}{\sim} \text{Unif}(a_n^{(m)}, b_n^{(m)})$
 - 5: Sort to obtain $X_{n,(1)} \leq \dots \leq X_{n,(m)}$
 - 6: Form the candidate family $\mathcal{C}_n^{(m)}$ as in Definition 2, excluding the full interval $I_n^{(m)}$
 - 7: Select a shortest admissible interval from $\mathcal{C}_n^{(m)}$
 - 8: Set $I_{n+1}^{(m)} = [a_{n+1}^{(m)}, b_{n+1}^{(m)}]$
 - 9: $n \leftarrow n + 1$
 - 10: **end while**
 - 11: **Output:** final bracket $I_n^{(m)}$ and midpoint estimator $\hat{x}_n^{(m)} = (a_n^{(m)} + b_n^{(m)})/2$
-

Proposition 1 (Well-posedness and root preservation) *Under Assumption 1, the m -sample algorithm is well defined. Moreover,*

$$x^* \in I_n^{(m)} \quad \text{for all } n \geq 0.$$

4.2 Spacing-based contraction analysis

Definition 3 (Normalized spacings) Let $U_1, \dots, U_m \stackrel{\text{i.i.d.}}{\sim} \text{Unif}(0, 1)$, and write $U_{(1)} \leq \dots \leq U_{(m)}$ for the corresponding order statistics. Set $U_{(0)} := 0$, $U_{(m+1)} := 1$,

and define the normalized spacings

$$S_j := U_{(j+1)} - U_{(j)}, \quad j = 0, 1, \dots, m.$$

Thus $(S_0, \dots, S_m)$ is the spacing vector induced by the ordered sample on $[0, 1]$, with $S_j \geq 0$ and $\sum_{j=0}^m S_j = 1$.

The parameter m allows trading increased per-iteration sampling effort for a smaller spacing-based contraction benchmark

$$B_m := \mathbb{E} \left[\max_{0 \leq j \leq m} S_j \right].$$

Proposition 2 (Spacing-based upper bound) *For every integer $m \geq 2$,*

$$\mathbb{E} \left[L_{n+1}^{(m)} \mid I_n^{(m)} \right] \leq \mathbb{E} \left[\max_{0 \leq j \leq m} S_j \right] L_n^{(m)} \quad \text{almost surely.}$$

Consequently, $B_m < 1$ serves as a universal upper bound on the expected one-step contraction factor.

Remark 6 (Localization of the contraction mechanism) The bound above is governed by the maximal spacing and is therefore conservative. Pathwise, the selected interval is controlled by the spacing containing the root, which is typically smaller than the maximal spacing. This explains the use of the maximal spacing in the theory and why empirically observed contraction factors are often below the benchmark B_m .

Thus the maximal-spacing benchmark provides a universal, root-location-free upper bound on the contraction mechanism, while the actual update is typically governed by a smaller local spacing containing the root. A direct analysis of this root-containing spacing would require characterizing its distribution relative to the random ordered sample, which appears substantially more delicate than the maximal-spacing analysis used here. Establishing such a characterization could yield sharper contraction constants and a more refined description of the stochastic contraction mechanism, and therefore remains an interesting open problem.

Theorem 2 (Expected contraction for the m -sample method) *For each integer $m \geq 2$,*

$$\mathbb{E} \left[L_{n+1}^{(m)} \mid I_n^{(m)} \right] \leq B_m L_n^{(m)} \quad \text{almost surely,}$$

where

$$B_m := \mathbb{E} \left[\max_{0 \leq j \leq m} S_j \right] < 1$$

serves as a universal upper bound on the expected one-step contraction factor. Consequently,

$$\mathbb{E}[L_n^{(m)}] \leq B_m^n L_0^{(m)} \quad \text{for all } n \geq 0.$$

Proof The one-step inequality follows directly from Proposition 2. Taking expectations and iterating the bound yields

$$\mathbb{E}[L_n^{(m)}] \leq B_m^n L_0^{(m)}.$$

Corollary 2 (Interval shrinkage in probability) *Under the assumptions of Theorem 2,*

$$L_n^{(m)} \rightarrow 0 \quad \text{in probability as } n \rightarrow \infty.$$

Proof For every $\varepsilon > 0$, Markov's inequality and Theorem 2 imply

$$\mathbb{P}(L_n^{(m)} > \varepsilon) \leq \frac{\mathbb{E}[L_n^{(m)}]}{\varepsilon} \leq \frac{L_0^{(m)}}{\varepsilon} B_m^n \rightarrow 0.$$

Corollary 3 (Consistency of the midpoint estimator) *Under the assumptions of Theorem 2,*

$$\hat{x}_n^{(m)} \rightarrow x^* \quad \text{in probability as } n \rightarrow \infty.$$

Proof By Proposition 1,

$$x^* \in I_n^{(m)} \quad \text{for all } n \geq 0.$$

Hence

$$|\hat{x}_n^{(m)} - x^*| \leq \frac{1}{2} L_n^{(m)},$$

and the result follows from Corollary 2.

Corollary 4 (High-probability iteration bound for the m -sample method) *Let $\varepsilon > 0$ and $0 < \delta < 1$. Under the assumptions of Theorem 2, if*

$$n \geq \frac{\log(L_0^{(m)}/\varepsilon) + \log(1/\delta)}{-\log B_m},$$

then

$$\mathbb{P}(L_n^{(m)} \leq \varepsilon) \geq 1 - \delta.$$

Equivalently,

$$\mathbb{P}(L_n^{(m)} > \varepsilon) \leq \frac{L_0^{(m)}}{\varepsilon} B_m^n.$$

In particular, the resulting iteration bound depends logarithmically on both the accuracy level $1/\varepsilon$ and the confidence parameter $1/\delta$.

Proof By Markov's inequality and Theorem 2,

$$\mathbb{P}(L_n^{(m)} > \varepsilon) \leq \frac{\mathbb{E}[L_n^{(m)}]}{\varepsilon} \leq \frac{L_0^{(m)}}{\varepsilon} B_m^n.$$

Hence it is sufficient to choose n such that

$$\frac{L_0^{(m)}}{\varepsilon} B_m^n \leq \delta.$$

Taking logarithms gives

$$n \geq \frac{\log(L_0^{(m)}/\varepsilon) + \log(1/\delta)}{-\log B_m},$$

which proves the claim. This argument does not require independence across iterations, since it relies only on the expectation bound from Theorem 2.

Corollary 5 (Serial and batched complexity) *Let*

$$n_{\text{batch}}(\varepsilon, \delta, m)$$

denote the number of update stages required to ensure

$$\mathbb{P}(L_n^{(m)} \leq \varepsilon) \geq 1 - \delta.$$

Then

$$n_{\text{batch}}(\varepsilon, \delta, m) \leq \frac{\log(L_0^{(m)}/\varepsilon) + \log(1/\delta)}{-\log B_m},$$

and hence

$$n_{\text{batch}}(\varepsilon, \delta, m) = O\left(\frac{\log(L_0^{(m)}/\varepsilon) + \log(1/\delta)}{-\log B_m}\right).$$

Under a serial cost model, each update requires m new interior evaluations, so the corresponding serial evaluation complexity satisfies

$$N_{\text{serial}}(\varepsilon, \delta, m) \leq m \frac{\log(L_0^{(m)}/\varepsilon) + \log(1/\delta)}{-\log B_m},$$

and therefore

$$N_{\text{serial}}(\varepsilon, \delta, m) = O\left(m \frac{\log(L_0^{(m)}/\varepsilon) + \log(1/\delta)}{-\log B_m}\right).$$

Proof The stated bound on $n_{\text{batch}}(\varepsilon, \delta, m)$ follows directly from Corollary 4 by solving the inequality

$$\mathbb{P}(L_n^{(m)} > \varepsilon) \leq \frac{L_0^{(m)}}{\varepsilon} B_m^n \leq \delta$$

for n . The serial complexity is obtained by multiplying the number of update stages by m , since each iteration requires m newly sampled interior function evaluations.

Remark 7 (Expected midpoint error bound) Although Corollary 3 establishes consistency in probability, the midpoint error also inherits the geometric decay of the interval length in expectation. Since

$$|\hat{x}_n^{(m)} - x^*| \leq \frac{1}{2} L_n^{(m)}$$

and Theorem 2 yields

$$\mathbb{E}[L_n^{(m)}] \leq B_m^n L_0^{(m)},$$

it follows that

$$\mathbb{E}[|\hat{x}_n^{(m)} - x^*|] \leq \frac{1}{2} B_m^n L_0^{(m)}.$$

Thus the midpoint error admits a geometric upper bound in expectation governed by the same spacing-based contraction factor.

Theorem 3 (Almost sure interval contraction) *Let*

$$Z := \max_{0 \leq j \leq m} S_j,$$

where $(S_0, \dots, S_m)$ is the normalized spacing vector from Definition 3. Suppose that

$$\mathbb{E}[|\log Z|] < \infty \quad \text{and} \quad \mathbb{E}[\log Z] < 0.$$

Then

$$L_n^{(m)} \rightarrow 0 \quad \text{almost surely as } n \rightarrow \infty.$$

Proof By Proposition 2, one has the pathwise inequality

$$L_{n+1}^{(m)} \leq L_n^{(m)} Z_{n+1},$$

where

$$Z_{n+1} := \max_{0 \leq j \leq m} S_{n+1,j},$$

and $(S_{n+1,0}, \dots, S_{n+1,m})$ denotes the normalized spacing vector generated at iteration $n+1$.

Conditional on $I_n^{(m)} = [a_n^{(m)}, b_n^{(m)}]$, the rescaled sample

$$\left(\frac{X_{n,1} - a_n^{(m)}}{L_n^{(m)}}, \dots, \frac{X_{n,m} - a_n^{(m)}}{L_n^{(m)}} \right)$$

consists of i.i.d. $\text{Unif}(0, 1)$ random variables. Since fresh independent samples are drawn at each iteration, the normalized spacing vectors are i.i.d. across iterations. Consequently, the variables $\{Z_n\}_{n \geq 1}$ are i.i.d. copies of

$$Z = \max_{0 \leq j \leq m} S_j.$$

Iterating the one-step bound yields

$$L_n^{(m)} \leq L_0^{(m)} \prod_{k=1}^n Z_k.$$

Taking logarithms, we obtain

$$\log L_n^{(m)} \leq \log L_0^{(m)} + \sum_{k=1}^n \log Z_k.$$

By the strong law of large numbers,

$$\frac{1}{n} \sum_{k=1}^n \log Z_k \rightarrow \mathbb{E}[\log Z] \quad \text{almost surely.}$$

If $\mathbb{E}[\log Z] < 0$, then

$$\sum_{k=1}^n \log Z_k \rightarrow -\infty \quad \text{almost surely,}$$

which implies

$$\prod_{k=1}^n Z_k \rightarrow 0 \quad \text{almost surely.}$$

Therefore,

$$L_n^{(m)} \rightarrow 0 \quad \text{almost surely.}$$

Remark 8 (Verification of the logarithmic condition) In the present uniform-spacing setting, the maximal spacing variable

$$Z = \max_{0 \leq j \leq m} S_j$$

satisfies $0 < Z < 1$ almost surely, and therefore

$$\log Z < 0 \quad \text{almost surely.}$$

Moreover, since the $m + 1$ spacings are nonnegative and sum to one, their maximum must satisfy

$$Z \geq \frac{1}{m+1} \quad \text{almost surely.}$$

Hence

$$|\log Z| \leq \log(m+1),$$

so in particular $\mathbb{E}[|\log Z|] < \infty$. It follows that $\mathbb{E}[\log Z] < 0$, and therefore the hypothesis of Theorem 3 is automatically satisfied for every $m \geq 2$.

Remark 9 (Formula for the maximal-spacing benchmark) For m independent uniform points, the $m + 1$ normalized spacings have the Dirichlet(1, ..., 1) distribution. A standard inclusion–exclusion calculation gives

$$B_m = \mathbb{E} \left[\max_{0 \leq j \leq m} S_j \right] = \frac{1}{m+1} \sum_{k=1}^{m+1} \frac{1}{k}.$$

This yields, for example,

$$B_2 = \frac{11}{18}, \quad B_3 = \frac{25}{48}, \quad B_5 = \frac{49}{120}, \quad B_{10} \approx 0.2745.$$

Proposition 3 (Monotonicity of the spacing benchmark) *For*

$$B_m = \frac{1}{m+1} \sum_{k=1}^{m+1} \frac{1}{k},$$

one has

$$B_{m+1} < B_m \quad \text{for all } m \geq 1.$$

Proof Let

$$H_r := \sum_{k=1}^r \frac{1}{k}.$$

Then

$$B_m = \frac{H_{m+1}}{m+1}.$$

Thus $B_{m+1} < B_m$ is equivalent to

$$\frac{H_{m+2}}{m+2} < \frac{H_{m+1}}{m+1}.$$

Using

$$H_{m+2} = H_{m+1} + \frac{1}{m+2},$$

this becomes

$$(m+1) \left(H_{m+1} + \frac{1}{m+2} \right) < (m+2) H_{m+1},$$

which is equivalent to

$$\frac{m+1}{m+2} < H_{m+1}.$$

Since $H_{m+1} \geq 1$ for all $m \geq 1$, while $\frac{m+1}{m+2} < 1$, the inequality holds. Therefore $B_{m+1} < B_m$.

Corollary 6 (Exact threshold relative to bisection) *Let*

$$B_m = \frac{H_{m+1}}{m+1}, \quad H_r = \sum_{k=1}^r \frac{1}{k},$$

denote the maximal-spacing contraction benchmark and the r -th harmonic number, respectively. Then

$$B_m < \frac{1}{2} \iff m \geq 4.$$

Proof From the harmonic-number representation we obtain

$$B_3 = \frac{H_4}{4} = \frac{1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4}}{4} = \frac{25}{48} > \frac{1}{2},$$

whereas

$$B_4 = \frac{H_5}{5} = \frac{1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5}}{5} = \frac{137}{300} < \frac{1}{2}.$$

Proposition 3 shows that the sequence $\{B_m\}_{m \geq 1}$ is strictly decreasing in m . It follows that $B_m < 1/2$ for all $m \geq 4$, and that no smaller value of m can satisfy this inequality. This proves the claimed equivalence.

Remark 10 (Interpretation relative to bisection) Corollary 6 shows that the maximal-spacing benchmark B_m falls below the deterministic bisection factor $1/2$ as soon as $m \geq 4$. This does not imply superiority in serial cost, since B_m is only an upper bound on expected one-step contraction and larger m also increases function-evaluation requirements. The potential advantage of $B_m < 1/2$ is most relevant in batched or parallel settings, where iteration count is the dominant cost measure.

Table 2: Representative values of the maximal-spacing benchmark $B_m = \mathbb{E}[M_m]$, where M_m denotes the maximal spacing generated by m uniformly sampled interior points. The deterministic bisection contraction factor is $1/2$; the benchmark satisfies $B_m < 1/2$ for the first time at $m = 4$.

m	B_m (exact)	Approximation	Comparison with $1/2$
2	11/18	0.611	$> 1/2$
3	25/48	0.521	$> 1/2$
4	137/300	0.457	$< 1/2$
5	49/120	0.408	$< 1/2$
10	–	0.275	$< 1/2$

Corollary 7 (Asymptotic spacing bound for B_m) *As $m \rightarrow \infty$,*

$$B_m = O\left(\frac{\log m}{m}\right).$$

More precisely, there exist constants $C_{\text{sp}} > 0$ and $m_0 \geq 2$ such that

$$B_m \leq C_{\text{sp}} \frac{\log m}{m} \quad \text{for all } m \geq m_0.$$

Thus increasing m yields diminishing returns at the level of this benchmark: the contraction bound improves at a logarithmic-over-linear rate, while per-iteration sampling cost grows linearly in m .

Proof By definition,

$$B_m = \mathbb{E}\left[\max_{0 \leq j \leq m} S_j\right].$$

Classical results on maximal uniform spacings; see, for example, Deheuvels [4, 5], Devroye [6], and the survey discussion in Pyke [11], imply that there exists a constant $C_{\text{sp}} > 0$ such that

$$\mathbb{E}\left[\max_{0 \leq j \leq m} S_j\right] \leq C_{\text{sp}} \frac{\log m}{m}$$

for all sufficiently large m . This proves the stated bound.

Corollary 8 (Asymptotic batched complexity) *As $m \rightarrow \infty$,*

$$n_{\text{batch}}(\varepsilon, \delta, m) = O\left(\frac{\log(L_0^{(m)}/\varepsilon) + \log(1/\delta)}{\log(m/\log m)}\right).$$

Proof From the harmonic-number representation we have

$$B_m = \frac{H_{m+1}}{m+1}, \quad H_{m+1} = \log m + \gamma + o(1),$$

so

$$B_m = \frac{\log m + \gamma + o(1)}{m} = \frac{\log m}{m} \left(1 + \frac{\gamma + o(1)}{\log m}\right).$$

Thus

$$-\log B_m = -\log\left(\frac{\log m}{m}\right) - \log\left(1 + \frac{\gamma + o(1)}{\log m}\right) = \log\left(\frac{m}{\log m}\right) + O\left(\frac{1}{\log m}\right),$$

and hence

$$-\log B_m \asymp \log\left(\frac{m}{\log m}\right) \quad (m \rightarrow \infty).$$

The stated bound now follows from Corollary 5 by substituting this asymptotic expression for $-\log B_m$.

Remark 11 (Interpretation) The improvement obtained by increasing m is governed by the logarithmic factor $\log(m/\log m)$. Consequently, larger sample sizes can substantially reduce the required number of update stages, but the gain eventually exhibits diminishing returns as m grows. This behavior is consistent with the empirical scaling observed in Section 5.

4.3 A heuristic cost-contraction perspective

The parameter m governs a basic tradeoff between per-iteration cost and interval reduction: larger m improves the spacing-based upper bound through a smaller benchmark B_m , but each update requires m newly sampled interior function evaluations. Under a serial cost model, a natural efficiency index is

$$C_m^{\text{serial}} := m B_m,$$

which heuristically measures expected contraction per unit serial evaluation effort. Under an idealized parallel model in which the m interior evaluations are carried out simultaneously, the relevant benchmark is

$$C_m^{\text{parallel}} := B_m.$$

Remark 12 (Heuristic choice of the sample size m) The asymptotic bound $B_m = O((\log m)/m)$ implies that $C_m^{\text{serial}} = mB_m$ behaves like a constant multiple of $\log m$ for large m , so increasing m indefinitely does not yield unbounded serial gains. In practice there is a “sweet spot” at moderate values of m , while larger m can be advantageous when interior evaluations are performed concurrently. The Monte Carlo results in Section 5 illustrate this behavior in a finite-sample setting.

Proposition 4 (Per-iteration computational cost) *For fixed sample size m , one iteration of the RIC- m algorithm requires m newly sampled interior function evaluations. The arithmetic cost includes sorting the sampled points, which requires $O(m \log m)$ operations, together with interval-selection overhead. A naive scan of the full candidate family costs $O(m^2)$, but this can be reduced substantially by exploiting the ordered-spacing structure and sign localization. For fixed m , the per-iteration arithmetic cost is independent of the target accuracy level.*

5 Numerical experiments

This section examines the finite-sample behavior of the randomized interval-contraction (RIC) procedures in settings aligned with the theory and in an exploratory non-monotone

and noisy regime. The main goals are: (i) to assess whether the mean interval length exhibits approximately geometric decay; (ii) to examine how the sample-size parameter m affects empirical contraction, iteration count, and evaluation cost; and (iii) to compare the proposed method with standard deterministic root-finding benchmarks under method-appropriate stopping rules.

Because the method is motivated partly by batched evaluation settings, we distinguish from the outset between two cost interpretations. Under a serial cost model, each scalar function evaluation is charged sequentially. Under an idealized parallel cost model, the m interior evaluations at a randomized update are viewed as concurrent, so iteration count is a proxy for synchronization cost or wall-clock depth.

For the m -sample method, let

$$I_n^{(m)} = [a_n^{(m)}, b_n^{(m)}], \quad L_n^{(m)} = b_n^{(m)} - a_n^{(m)}, \quad \hat{x}_n^{(m)} = \frac{a_n^{(m)} + b_n^{(m)}}{2}.$$

For the two-sample case $m = 2$, we write

$$I_n = [a_n, b_n], \quad L_n = b_n - a_n, \quad \hat{x}_n = \frac{a_n + b_n}{2}.$$

5.1 Performance metrics and experimental design

For the randomized procedures, we record the midpoint root error

$$E_n^{(m)} = |\hat{x}_n^{(m)} - x^*|,$$

the residual magnitude

$$R_n^{(m)} = |f(\hat{x}_n^{(m)})|,$$

and the realized one-step contraction ratio

$$Q_n^{(m)} = \frac{L_{n+1}^{(m)}}{L_n^{(m)}}.$$

We also report the number of iterations, the number of function evaluations, the empirical success probability, and the median final root error.

For each iteration n , we compute the Monte Carlo means

$$\bar{L}_n^{(m)} = \frac{1}{R} \sum_{r=1}^R L_{n,r}^{(m)}, \quad \bar{E}_n^{(m)} = \frac{1}{R} \sum_{r=1}^R E_{n,r}^{(m)}, \quad \bar{R}_n^{(m)} = \frac{1}{R} \sum_{r=1}^R R_{n,r}^{(m)}.$$

The deterministic solvers are included as reference baselines rather than as targets to be uniformly outperformed. In smooth serial settings, Brent-type methods remain preferable. The proposed method is intended for settings where bracket preservation, randomized exploration, and batched evaluation are central design requirements. We compare the randomized interval-contraction method for

$$m \in \{2, 3, 5, 10\}$$

with three standard derivative-free benchmarks: bisection, secant, and Brent's method; see, for example, [2, 7, 10]. These benchmarks are included as reference methods in

favorable deterministic settings, and the comparison is intended to provide context for the randomized procedures rather than to claim dominance over mature deterministic solvers in smooth serial problems.

For each monotone test function in Table 3 and each value of m , we generate

$$R = 1000$$

independent runs of the randomized algorithm from the stated initial bracket. For the randomized methods and for bisection, a run is declared successful when the interval-width stopping rule

$$L_n^{(m)} \leq 10^{-8}$$

is first satisfied, or unsuccessful if the iteration cap

$$N_{\max} = 100$$

is reached first. For secant and Brent's method, which do not maintain a bracketing interval, success is assessed by the accuracy of the computed root estimate under the same iteration budget. Thus the stopping rules are method-aware: interval width for bracket-preserving methods, and estimate-based accuracy for open or hybrid methods.

Function-evaluation counts are recorded at the implementation level. For the randomized methods, this includes the initial endpoint evaluations, all newly sampled interior evaluations, and any additional endpoint evaluations required if previously computed values are not cached. In the reported implementation, cached endpoint values are reused across iterations whenever possible, so evaluation counts reflect actual oracle calls.

For bisection, evaluation counts include the initial endpoint evaluations and one new midpoint evaluation per iteration. For secant and Brent's method, the counts are the total scalar function calls made until termination or failure. Accordingly, evaluation totals are directly comparable as serial oracle counts, while iteration totals are more naturally interpreted under the idealized parallel viewpoint.

To summarize average interval contraction across replications, we use the empirical quantity

$$\hat{\rho}_m = \frac{1}{N^*} \sum_{n=0}^{N^*-1} \frac{\bar{L}_{n+1}^{(m)}}{\bar{L}_n^{(m)}},$$

where N^* is chosen so that the denominator remains numerically stable. This is a contraction summary for the average trajectory, not the average of pathwise ratios.

It is also informative to compare these empirical contraction summaries with the spacing-based benchmark

$$B_m = \mathbb{E} \left[\max_{0 \leq j \leq m} S_j \right].$$

Here $\hat{\rho}_m$ is an empirical summary, whereas B_m is the theoretical spacing-based upper bound from Section 4. For the values considered here,

$$B_2 \approx 0.6111, \quad B_3 \approx 0.5208, \quad B_5 \approx 0.4083, \quad B_{10} \approx 0.2745.$$

As expected from the conservative spacing analysis, the empirical contraction summaries are smaller than these benchmarks while displaying the same overall decreasing trend with m .

All randomized experiments were carried out with fixed pseudo-random seeds to ensure reproducibility. For the bracket-preserving methods, failure can occur only if the iteration cap is reached before the interval-width tolerance is attained. For secant and Brent’s method, failure means that the solver does not attain the required accuracy within the allowed budget. Implementation code and scripts used to generate the numerical results are provided in the supplementary material and will be archived in a public repository upon publication.

5.2 Test problems

We consider five monotone test functions satisfying Assumption 1. This choice aligns the main test set with the assumptions used in the theoretical development. The functions represent a range of local geometries, including nearly linear, moderately nonlinear, and relatively flat regimes near the root.

In addition, we include one exploratory non-monotone example to assess robustness beyond Assumption 1. This example is not included in the main monotone-function tables (Tables 6 and 7) and is discussed separately as an empirical stress test outside the formal theory.

Table 3: Test functions used in the numerical study. The first five functions satisfy Assumption 1 and constitute the main deterministic test set used in the reported summary tables and figures. The final example is a separate exploratory non-monotone robustness check and is not included in the main monotone pooled summaries.

Function	Expression	Root x^*	Initial interval $[a_0, b_0]$
f_1	$x - e^{-x}$	0.567143...	$[0, 1]$
f_2	$x^3 + x - 1$	0.682328...	$[0, 1]$
f_3	$\arctan(x) - 0.3$	$\tan(0.3)$	$[0, 1]$
f_4	$(x - x^*)^3 + 0.05(x - x^*)$	0.6	$[0, 1]$
f_5	$\log(1 + x) - 0.4$	$e^{0.4} - 1$	$[0, 1]$
f_6	$\sin(x) - 0.3$	0.304693...	$[0, 2]$

The additional test function f_6 is included only as an exploratory robustness check outside the monotone-theory regime; the function is not monotone on the initial interval, even though the chosen bracket contains a unique root. On this example, the method is implemented using the same shortest-admissible-interval rule based on endpoint sign information. No theoretical guarantees are claimed for this non-monotone example.

5.3 Scaling with sample size m

To investigate how contraction quality and computational cost vary with the sample-size parameter, we examine the randomized interval-contraction method over the range

$$m \in \{2, 3, 5, 10, 20, 50\}.$$

For each value of m , we report:

- the empirical contraction summary $\hat{\rho}_m$;
- the mean number of iterations to reach the stopping tolerance;
- the mean total number of scalar function evaluations to reach the stopping tolerance;
- the normalized serial cost

$$\frac{\text{mean total function evaluations}}{\log(L_0/\varepsilon)}.$$

Unless otherwise stated, the values reported in Table 4 are pooled averages over the five monotone test functions $f_1, \dots, f_5$, with $R = 1000$ replications per function and per value of m .

Table 4: Scaling behavior of the randomized interval-contraction method pooled over the five monotone test functions $f_1, \dots, f_5$. The empirical contraction summary $\hat{\rho}_m$ decreases substantially as m increases, while the normalized serial cost $\bar{N}_{\text{feval}}/\log(L_0/\varepsilon)$ increases, illustrating the contraction–cost tradeoff inherent in the method.

m	Mean Iter.	Mean Evals	$\mathbb{P}(\text{Succ.})$	Mean final length	$\hat{\rho}_m$	Norm. serial cost
2	23.117	48.234	1.000	5.95e-09	0.50419	2.618
3	17.838	55.513	1.000	5.59e-09	0.40742	3.014
5	13.466	69.328	1.000	4.88e-09	0.30681	3.764
10	9.752	99.518	1.000	4.09e-09	0.27175	5.403
20	7.510	152.192	1.000	3.46e-09	0.16923	8.262
50	5.773	290.650	1.000	2.64e-09	0.06594	15.778

Table 5: Serial and batched performance of the randomized interval-contraction method pooled over the five monotone test functions. Under the idealized batched model, the mean iteration count provides an empirical estimate of the batched complexity n_{batch} , whereas the mean number of function evaluations provides an empirical estimate of the serial cost N_{serial} .

m	Mean Iter.	Mean Evals	$\mathbb{P}(\text{Succ.})$	Empirical contraction $\hat{\rho}_m$	Benchmark B_m
2	23.1	48.2	1.000	0.504	0.6111
3	17.8	55.5	1.000	0.407	0.5208
5	13.5	69.3	1.000	0.307	0.4083
10	9.75	99.5	1.000	0.272	0.2745

Serial versus batched interpretation. Table 5 highlights the central tradeoff of the RIC- m framework. Increasing m substantially reduces the number of update stages (parallel depth) and improves the empirical contraction factor, but increases the total number of scalar function evaluations. The empirical contraction factors remain below the theoretical spacing benchmarks B_m , consistent with Proposition 2 and Remark 6, which show that the maximal-spacing analysis provides a conservative upper bound on the actual root-containing contraction mechanism. Larger values of m are therefore most attractive when interior evaluations can be performed concurrently, whereas more moderate values of m may be preferable under a purely serial cost model.

Because all runs are terminated using the same interval-width tolerance, the final median errors are nearly identical across values of m . The primary effect of increasing m is therefore seen in contraction, iteration count, and serial evaluation cost rather than in terminal accuracy under the common stopping rule.

The empirical contraction summary $\hat{\rho}_m$ decreases monotonically over the values considered, from approximately 0.50 at $m = 2$ to about 0.07 at $m = 50$, while the pooled mean number of iterations decreases from about 23 to about 6. This improvement is accompanied by a substantial increase in serial function-evaluation cost, which rises from roughly 48 evaluations to about 291 evaluations across the same range of m .

Under an idealized batched-evaluation model, the mean iteration count can be viewed as a surrogate for update-stage or synchronization depth. From this perspective, the decrease in pooled mean iterations from about 23.1 for $m = 2$ to about 5.8 for $m = 50$ indicates that larger sample sizes can substantially reduce stage depth even though they increase total serial function-evaluation cost. This parallel-depth viewpoint is taken up again in Section 5.6.

The corresponding visual summary is shown in Figure 2.

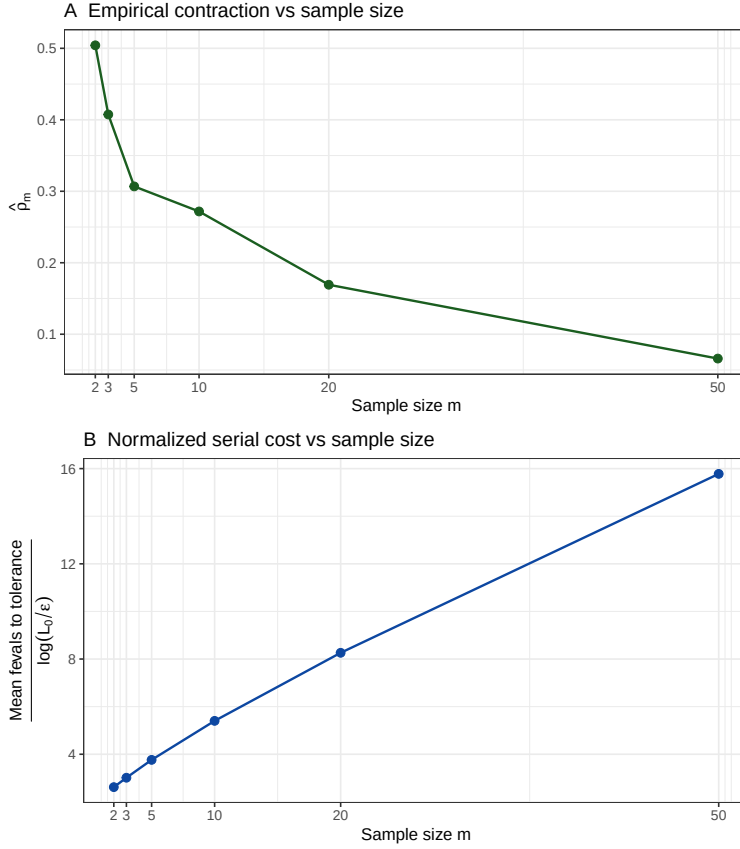


Fig. 2: Scaling behavior of the randomized interval-contraction method as the sample size m varies. Panel (A) shows the pooled empirical contraction summary $\hat{\rho}_m$ over the five monotone test functions $f_1, \dots, f_5$; smaller values correspond to stronger mean interval contraction. Panel (B) shows the pooled normalized serial cost, defined as the mean number of scalar function evaluations to tolerance divided by $\log(L_0/\epsilon)$. Larger m improves contraction but increases serial evaluation cost.

5.4 Overall performance

Table 6 reports method-by-method results for each of the five monotone test functions separately. In contrast, Table 7, deferred to Appendix A, reports empirical contraction summaries for the randomized methods by test function and sample size. The pooled scaling results in Table 4 aggregate across the five monotone test functions rather than reporting per-function behavior.

Table 6: Overall numerical performance on the five monotone test functions $f_1, \dots, f_5$. For each function and method, the table reports the mean number of iterations to tolerance, the mean total number of scalar function evaluations to tolerance, the empirical success probability, and the median final root error. Randomized methods and bisection are stopped by the interval-width criterion, whereas secant and Brent’s method are assessed by the accuracy of the final root estimate under the same iteration budget.

Func.	Method	Mean Iter.	Mean f -evals/run	Succ.	Median Error
f_1	RIC-2	23.149	72.447	1.000	1.27×10^{-9}
	RIC-3	17.777	74.108	1.000	1.15×10^{-9}
	RIC-5	13.453	83.718	1.000	9.77×10^{-10}
	RIC-10	9.754	110.294	1.000	7.26×10^{-10}
	Bisection	27.000	29.000	1.000	2.90×10^{-9}
	Secant	6.000	7.000	1.000	7.93×10^{-14}
	Brent	5.000	7.000	1.000	8.03×10^{-12}

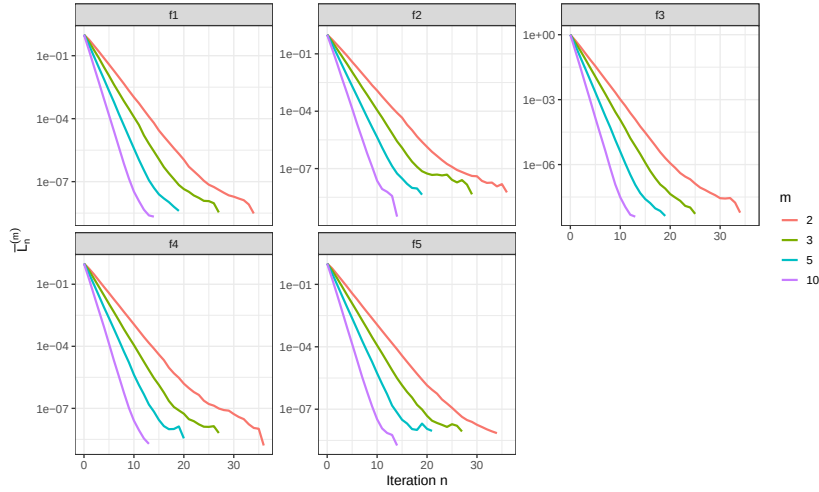


Fig. 3: Average interval length $\bar{L}_n^{(m)}$ versus iteration number n for the randomized interval-contraction algorithm with $m \in \{2, 3, 5, 10\}$ across the five monotone test functions. The approximately linear decay on the log scale is consistent with geometric interval contraction, and the slope becomes steeper as m increases.

Over the five monotone test problems, the randomized interval-contraction methods exhibit stable geometric decay of the mean interval length and an overall improvement trend in contraction as m increases.

Figure 3 shows that the average interval length is approximately linear on the log scale after a short transient, consistent with geometric decay. Additional per-function contraction summaries are reported in Table 7. For example, $\hat{\rho}_m$ decreases

from approximately 0.50 for $m = 2$ to roughly 0.24–0.29 for $m = 10$, illustrating the expected improvement in contraction with larger sampled batches.

Remark 13 (Function-insensitivity) The empirical contraction factors are quite stable across the five test functions. This reflects the fact that interval contraction is driven primarily by the spacing geometry induced by the sampled points rather than by fine details of the function beyond the monotonicity and bracketing structure assumed in the analysis.

Midpoint-error trajectories, reported in Figure 4 in Appendix A, closely track the interval-length curves, in agreement with the deterministic bound

$$|\hat{x}_n^{(m)} - x^*| \leq \frac{1}{2} L_n^{(m)}.$$

Table 6 also shows that the randomized methods achieved success probability one for every function and every value of m . The mean iteration count decreases markedly with m : across the five test problems, it is about 23.1 for $m = 2$, about 17.8 for $m = 3$, about 13.4 for $m = 5$, and about 9.75 for $m = 10$, so passing from $m = 2$ to $m = 10$ reduces iteration count by more than half, at the cost of additional function evaluations.

5.5 Deterministic benchmark comparison

On the smooth monotone test problems, the deterministic benchmarks behave as expected: bisection is uniformly reliable but conservative, while secant and Brent’s method attain the target accuracy with substantially fewer iterations and scalar function evaluations. The proposed method is therefore best viewed as a bracket-preserving alternative rather than as a serial replacement for classical solvers in low-noise settings.

Compared with bisection, the randomized method uses fewer iterations once $m \geq 3$. For example, with $m = 3$ the mean iteration count is about 17.8, and with $m = 10$ it is about 9.75, compared with 27 for bisection. However, the function-evaluation cost is considerably larger: approximately 72–74 evaluations for $m = 2, 3$, about 83 for $m = 5$, and about 110 for $m = 10$, versus 29 for bisection.

The contrast with secant and Brent’s method is stronger. On f_1 , for instance, Brent requires 5 iterations and 7 function evaluations with median final root error about 8.03×10^{-12} , whereas secant requires 6 iterations and 7 evaluations with error about 7.93×10^{-14} . By comparison, the randomized methods terminate at the prescribed bracket tolerance and deliver final errors at the 10^{-9} level while using substantially more evaluations. A graphical efficiency comparison is included in Figure 5.

Caveat for noisy evaluations. The theoretical root-preservation and contraction results in Sections 3 and 4 assume exact function evaluations. Under additive noise, signs may be misclassified, so deterministic bracket preservation is no longer guaranteed and the theory does not apply directly. The noisy examples in Appendix A.2 are therefore purely exploratory; a rigorous treatment of noisy sign information would require a separate analysis and is left for future work.

5.6 Parallel and cost interpretations

We finally reinterpret the numerical results under the idealized parallel evaluation viewpoint. If the m interior function evaluations at each randomized update are carried out simultaneously, then iteration count rather than total scalar function evaluations becomes the more relevant cost measure. Under this interpretation, the decrease in mean iteration count from about 23.1 for $m = 2$ to about 9.75 for $m = 10$ becomes operationally significant and the method compares more favorably with deterministic bracketing schemes such as bisection.

A simple summary of the idealized parallel benefit is the iteration ratio relative to the baseline $m = 2$ method. Using the mean iteration counts in Table 6, the ratio

$$\frac{\text{mean iterations for RIC-}m}{\text{mean iterations for RIC-2}}$$

is approximately 0.77 for $m = 3$, 0.58 for $m = 5$, and 0.42 for $m = 10$ across the monotone test set. If one charges one unit per update stage and treats the m interior evaluations within a stage as concurrent, then the mean stage count is exactly the mean iteration count in Table 6. Thus, although the randomized procedures are not competitive with secant or Brent's method in raw serial evaluation count, they exhibit a substantial reduction in synchronization depth as m increases.

Among the values considered here, $m = 3$ appears to offer a balanced compromise: it reduces the empirical contraction factor from roughly 0.505 to roughly 0.406, lowers the mean iteration count from about 23.1 to about 17.8, and increases the mean number of function evaluations only modestly relative to $m = 2$. In the present test set, $m = 3$ therefore occupies an intermediate regime in which contraction and stage count improve noticeably without incurring the higher serial cost associated with $m = 5$ or $m = 10$.

Each iteration requires m newly sampled interior evaluations, together with any additional endpoint or candidate evaluations needed to assess admissibility. Sorting the sampled points requires $O(m \log m)$ arithmetic operations. A naive scan of the full candidate family may require $O(m^2)$ interval checks, but the ordered-spacing structure and sign information can be exploited to reduce the practical interval-selection overhead substantially. In many black-box applications, the dominant cost is therefore the function evaluation stage rather than the associated arithmetic.

Remark 14 (Cost interpretation) Under a serial implementation, one update of the m -sample method requires m newly sampled interior evaluations, so the total evaluation cost after n iterations scales as $O(mn)$. Under an idealized parallel model in which these evaluations are carried out simultaneously, iteration (stage) count is a more relevant proxy for wall-clock cost than the raw number of function evaluations. The practical value of increasing m therefore depends on the computational environment and the target accuracy.

5.7 Discussion

The numerical experiments indicate that the proposed method should be interpreted as a robust randomized bracketing procedure rather than as a serial replacement for

Brent-type solvers. The computational niche of the framework is in settings where explicit bracket preservation is important and the dominant cost is function evaluation rather than arithmetic overhead. When evaluations are expensive or naturally batchable, reducing the number of update stages can be more relevant than minimizing the total number of serial evaluations. In this regime, the parameter m serves as a tuning parameter: smaller values favor serial evaluation economy, whereas larger values may be advantageous when multiple interior evaluations can be carried out concurrently.

Within the surrogate cost models studied here, the numerical evidence is consistent with this interpretation. Under a serial accounting model, the randomized procedures are more expensive than the deterministic benchmarks. Under the idealized batched viewpoint, however, increasing m reduces the number of update stages required to reach the target tolerance.

The experiments also illustrate the current limits of the theory. The main analysis covers continuous strictly monotone functions with a valid initial bracket and exact sign information. The exploratory non-monotone and noisy experiments suggest that the sampling-and-selection mechanism can remain useful beyond that regime, but they do not provide formal guarantees. Extending the theory to non-monotone brackets, multiple roots, or noisy-oracle settings remains an important direction for future work.

6 Concluding remarks

This paper introduced a spacing-based randomized interval-contraction framework for derivative-free root finding on bounded intervals. The central idea is to drive contraction by the geometry of random spacings induced by uniform interior sampling. The main conceptual contribution is to connect classical bracket-preserving root finding with order statistics and maximal-spacing theory. This yields a probabilistically tractable family of randomized algorithms with explicit upper contraction bounds B_m and rigorous convergence guarantees (in probability and almost surely) under the monotone setting.

Algorithmically, the methods are lightweight: each iteration requires only uniform sampling within the current bracket, m scalar function evaluations, and simple sorting plus sign-based interval selection. The bracket-preserving property makes them natural candidates for robust root-localization or interval-refinement modules in numerical libraries and derivative-free software.

From a computational perspective, the sample-size parameter m provides a transparent tuning mechanism under two distinct cost models. Under serial execution, larger m improves contraction at the expense of more function evaluations. Under an idealized batched model, however, increasing m reduces the number of update stages (synchronization depth) when the m interior evaluations can be performed concurrently. The numerical results therefore support a conditional computational advantage: the framework is not superior in raw serial evaluation count, but offers a clear reduction in stage depth when function evaluations are expensive or naturally batchable.

Accordingly, the proposed methods are best viewed as robust bracket-preserving tools for derivative-free black-box problems in which function evaluations are costly or naturally batchable, rather than as direct competitors to mature deterministic solvers in favorable serial environments.

Several directions remain open for future work. Theoretically, sharper analysis based directly on the root-containing spacing (rather than the maximal spacing) could yield tighter bounds. Methodologically, adaptive choice of m , hybrid deterministic–randomized updates, non-uniform sampling, extensions to noisy oracles, and non-monotone functions are natural next steps. It would also be valuable to study implementation-level performance in realistic parallel and distributed environments, as well as generalizations to multiple roots and higher-dimensional systems.

The present framework is intrinsically one-dimensional due to its reliance on interval ordering and spacing geometry. Extending these ideas to systems of nonlinear equations will require new notions of admissibility and contraction, and constitutes a promising direction for randomized numerical methods.

Limitations and scope

The present analysis is carried out under a deliberately restricted set of assumptions: continuity and strict monotonicity of f on a bounded interval, a valid initial bracket satisfying

$$f(a_0)f(b_0) < 0,$$

and exact sign information at all evaluation points. These conditions ensure a unique root, guarantee that admissibility can be characterized through endpoint sign information, and allow the spacing-driven contraction mechanism to be analyzed through the geometry of uniform order statistics.

The resulting theory should therefore be viewed as a foundational analysis of the randomized interval-contraction framework rather than a complete treatment of all root-finding settings. Extensions to multiple roots, nonmonotone functions, stochastic or unreliable sign information, nonuniform or adaptive sampling schemes, and higher-dimensional systems of equations require additional analysis and are left for future work.

A Supplementary numerical material

This appendix collects supplementary numerical material that supports, but is not essential to, the main conclusions of Section 5. It contains additional deterministic-function diagnostics for the monotone test set together with a separate exploratory noisy-evaluation study. The noisy results lie outside the formal assumptions used in the theoretical development and are included only as an empirical robustness check.

A.1 Supplementary deterministic-function diagnostics

For completeness, this subsection reports additional deterministic summaries for the five monotone test functions $f_1, \dots, f_5$. These include per-function contraction summaries for the randomized methods, midpoint-error trajectories, and a graphical efficiency comparison with deterministic benchmark solvers.

Table 7: Empirical contraction summaries for the randomized methods on the five monotone test functions $f_1, \dots, f_5$. For each test function and each value of m , the table reports the empirical contraction estimate $\hat{\rho}_m$, the mean final interval length at termination, and the mean number of scalar function evaluations per run.

Function	m	$\hat{\rho}_m$	Mean final interval length	Mean f -evals/run
f_1	2	0.50396	5.93×10^{-9}	72.447
f_1	3	0.40533	5.58×10^{-9}	74.108
f_1	5	0.32010	5.00×10^{-9}	83.718
f_1	10	0.24021	3.92×10^{-9}	110.294
f_2	2	0.50560	5.94×10^{-9}	72.285
f_2	3	0.40632	5.52×10^{-9}	74.004
f_2	5	0.31653	4.92×10^{-9}	83.478
f_2	10	0.25779	4.23×10^{-9}	109.843
f_3	2	0.50574	5.95×10^{-9}	72.579
f_3	3	0.40531	5.57×10^{-9}	74.288
f_3	5	0.32352	4.80×10^{-9}	83.346
f_3	10	0.29215	4.27×10^{-9}	110.481
f_4	2	0.50435	5.89×10^{-9}	72.225
f_4	3	0.40606	5.48×10^{-9}	74.616
f_4	5	0.32003	4.81×10^{-9}	82.866
f_4	10	0.25336	4.00×10^{-9}	110.437
f_5	2	0.50542	6.06×10^{-9}	72.720
f_5	3	0.40572	5.50×10^{-9}	74.444
f_5	5	0.31798	4.92×10^{-9}	83.640
f_5	10	0.28190	3.97×10^{-9}	110.723

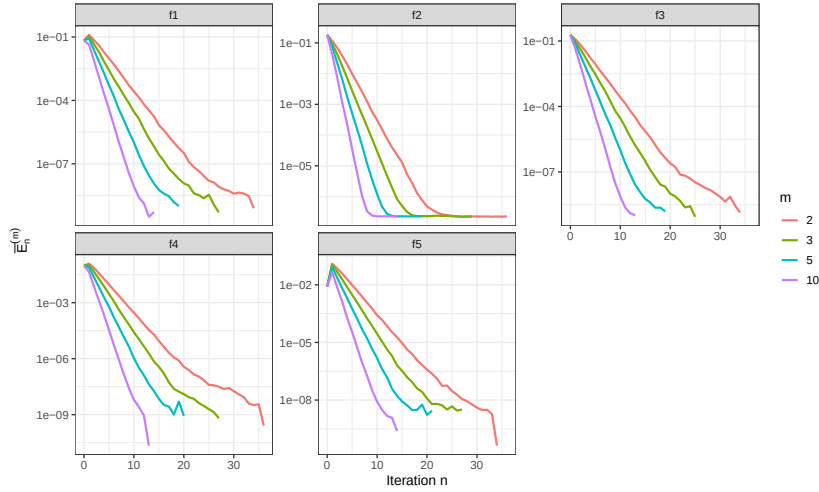


Fig. 4: Average midpoint root error $\bar{E}_n^{(m)}$ versus iteration number n for the randomized interval-contraction method with $m \in \{2, 3, 5, 10\}$ across the five monotone test functions. The error decay closely tracks the interval-length decay, consistent with the midpoint bound $|\hat{x}_n^{(m)} - x^*| \leq \frac{1}{2} L_n^{(m)}$.

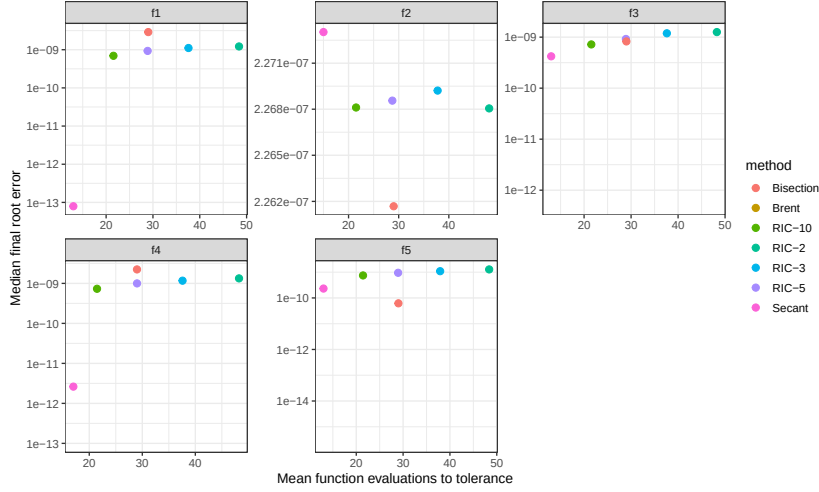


Fig. 5: Efficiency frontier for the randomized interval-contraction methods and deterministic benchmark solvers on the monotone test set. The horizontal axis shows the mean number of scalar function evaluations to tolerance, and the vertical axis shows the median final root error. Larger values of m tend to improve contraction and reduce iteration counts, but they also increase serial evaluation cost.

A.2 Exploratory noisy-evaluation study

This subsection reports a separate exploratory experiment under noisy function evaluations. The setting lies outside the formal assumptions of the theory because endpoint signs may be misclassified under additive noise, so the results should be interpreted only as an empirical robustness check of the sampling-and-selection mechanism.

We consider the monotone test function

$$g(x) = x - e^{-x}, \quad x \in [0, 1],$$

which has the unique root

$$x^* \approx 0.5671432904.$$

Instead of observing $g(x)$ exactly, we observe the perturbed value

$$\tilde{g}(x) = g(x) + \varepsilon,$$

where the evaluation noise is generated independently at each function call according to

$$\varepsilon \sim \mathcal{N}(0, \sigma^2), \quad \sigma = 10^{-3}.$$

The experiment uses the initial bracket $[a_0, b_0] = [0, 1]$, with $R = 1000$ independent replications and a maximum iteration budget of $N_{\max} = 100$. For the randomized interval-contraction procedures and for bisection, success is defined by attainment of the bracket-width tolerance $L_n^{(m)} \leq 10^{-6}$. For the secant and Brent-type benchmark methods, success is defined by the event $|\hat{x} - x^*| \leq 10^{-4}$ within the same iteration budget. We report results for the randomized methods with $m \in \{2, 3, 5, 10\}$.

Because bracketing and open methods return different objects, the noisy comparison is intentionally method-aware rather than based on identical stopping rules. In the presence of evaluation noise, interval-based criteria provide a more stable notion of convergence for bracketing methods.

Table 8: Exploratory performance under noisy function evaluations for $g(x) = x - e^{-x}$ on $[0, 1]$, with independent noise $\varepsilon \sim \mathcal{N}(0, 10^{-3})$ added at each evaluation (standard deviation $\sigma = 10^{-3}$). For each method, the table reports empirical success probability, median absolute root error, median iterations, and median *evaluation batches* over $R = 1000$ replications.

Method	Succ. Prob.	Med. Abs. Error	Med. Iter.	Med. Batches
RIC-2	1.000	0.000 303	13	15
RIC-3	1.000	0.000 286	10	12
RIC-5	1.000	0.000 327	8	10
RIC-10	1.000	0.000 329	6	8
Bisection	1.000	0.000 283	20	22
Secant	0.781	0.000 062 6	10	11
Brent	0.486	0.000 127	100	102

Table 8 indicates that the randomized interval-contraction methods remain empirically stable under the small independent evaluation noise considered here. All four randomized variants attained empirical success probability 1.000, matching bisection and exceeding the observed success rates of secant and Brent’s method under the method-specific stopping rules used in this experiment. Among the randomized procedures, increasing m reduced the median iteration count from 13 for RIC-2 to 6 for RIC-10, while the median absolute root errors remained of comparable magnitude.

Because the success criteria differ between bracket-preserving and open methods, this table should be interpreted as a qualitative robustness comparison rather than as a strict head-to-head accuracy contest. Its role is to illustrate one regime in which conservative, bracket-based updates can remain operationally attractive under noisy evaluations.

Data Availability

No external datasets were used in this study. All numerical results were generated from the test functions and simulation settings described in Section 5. A related public Zenodo item is: D. Kumar, A heuristic method for root-finding using random numbers, Zenodo, doi: [10.5281/zenodo.15328367](https://doi.org/10.5281/zenodo.15328367). Implementation scripts for the present manuscript are available from the corresponding author upon reasonable request.

Funding

Funding not applicable.

References

1. K. Atkinson, *An Introduction to Numerical Analysis*, 2nd ed., John Wiley & Sons, New York, 1989.
2. R. P. Brent, *Algorithms for Minimization without Derivatives*, Prentice-Hall, Englewood Cliffs, NJ, 1973.
3. R. L. Burden and J. D. Faires, *Numerical Analysis*, 9th ed., Brooks/Cole, Boston, 2011.

4. P. Deheuvels, Strong limit theorems for maximal spacings from a general univariate distribution, *Ann. Probab.* 12 (1984) 1181–1193.
5. P. Deheuvels, On the influence of the extremes of an i.i.d. sequence on the maximal spacings, *Ann. Probab.* 14 (1986) 194–208.
6. L. Devroye, Laws of the iterated logarithm for order statistics of uniform spacings, *Ann. Probab.* 9 (1981) 860–867.
7. G. E. Forsythe, M. A. Malcolm and C. B. Moler, *Computer Methods for Mathematical Computations*, Prentice-Hall, Englewood Cliffs, NJ, 1977.
8. D. Kumar, A heuristic method for root-finding using random numbers, Zenodo, 2025. doi: [10.5281/zenodo.15328367](https://doi.org/10.5281/zenodo.15328367).
9. H. J. Kushner and G. G. Yin, *Stochastic Approximation and Recursive Algorithms and Applications*, 2nd ed., Springer, New York, 2003.
10. W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, *Numerical Recipes: The Art of Scientific Computing*, 3rd ed., Cambridge University Press, Cambridge, 2007.
11. R. Pyke, Spacings, *J. Roy. Statist. Soc. Ser. B* 27 (1965) 395–449.
12. H. Robbins and S. Monro, A stochastic approximation method, *Ann. Math. Statist.* 22 (1951) 400–407.
13. G. R. Shorack and J. A. Wellner, *Empirical Processes with Applications to Statistics*, John Wiley & Sons, New York, 1986.
14. J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*, 3rd ed., Springer, New York, 2002.
15. R. Waeber, P. I. Frazier and S. G. Henderson, Bisection search for stochastic root-finding: The problem of unknown noise, *SIAM J. Control Optim.* 51 (2013) 2261–2279.