SHARP ROOT-EXPONENTIAL RATES FOR SMOOTH CONVEX OPTIMIZATION IN A SERIAL FINITE-BIT MODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

We establish the sharp convergence rate of deterministic smooth convex optimization under simultaneous limits on bit computation and live memory, providing a concrete benchmark for first-order algorithm design with finite resources. In fixed dimension d, we consider a finite-bit model in which useful numerical replies must be fully consumed in sequence. Each round includes one oracle call and permits at most $2^{12}d$ bit actions; live storage is bounded by $2^{20}\{A + d[p + \lceil \log_2(d+1) \rceil]\}$ bits, where A is the input bit length and p is the current numerical precision scale. We construct a single uniform algorithm with increasing precision whose reported point z_n satisfies $F(z_n) - F^* \leq C_0 \exp(-c_d\sqrt{n - n_0})$ at every sufficiently late round, uniformly over admissible rounding errors. The algorithm requires no prior knowledge of the smoothness constant, minimizer distance, target accuracy, or horizon. Conversely, for every deterministic program obeying the bit-work budget and reply protocol, even with unrestricted memory, we construct one fixed smooth convex objective whose run under a fixed nearest-grid oracle satisfies $F(z_{n_j}) - F^* \geq c \exp(-C\sqrt{n_j})$ along an unbounded subsequence of rounds. Together, these bounds establish $\exp[-\Theta(\sqrt{n})]$ as the sharp asymptotic order of eventual every-round guarantees in this protocol. With $O_d(1)$ bit actions per round and serial reply consumption, the same root-exponential order holds in total charged bit work.

1 INTRODUCTION

Finite numerical replies introduce costs beyond oracle calls: an optimizer must store, read, and compute with increasingly precise information. Classical first-order complexity counts calls (Bubeck, 2015), while communication complexity captures precision costs through square-logarithmic bounds for numerical-prefix protocols (Tsitsiklis & Luo, 1987). A sharp computational rate additionally requires implementable updates and a matching obstruction on one fixed objective throughout an infinite run.

We study deterministic smooth convex optimization in fixed dimension d, with $O(d)$ bit actions per round and live storage linear in working precision. Queries, replies, working state, and reports all count toward storage; useful value and full-gradient replies are consumed serially. The algorithm receives only the dimension and a finite anchor, and provides an answer every round without knowing smoothness, minimizer distance, target accuracy, or horizon.

Contributions. With $O_d(1)$ bit actions per round and serial reply consumption, the root-exponential order below also holds in total charged bit work. With (d, F, a) fixed before $n \to \infty$, our contributions are:

- **An upper construction.** To our knowledge, we give the first uniform ellipsoid implementation meeting these requirements and achieving $F(z_n) - F^* \leq C_0 \exp(-c_d\sqrt{n - n_0})$ at every sufficiently late round.
- **A fixed-instance lower construction.** For every deterministic program obeying the work budget and reply protocol, even with unrestricted memory, we construct one fixed

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

smooth convex objective whose fixed nearest-grid oracle run satisfies $F(z_{n_j}) - F^* \geq c\exp(-C\sqrt{n_j})$ on an unbounded subsequence. To our knowledge, this is the first fixed-instance root-exponential objective-gap lower bound under this serial finite-bit protocol.

- **The sharp rate in physical rounds.** Together, the constructions identify $\exp[-\Theta(\sqrt{n})]$ as the sharp order of universal eventual every-round guarantees: the lower subsequence excludes any faster exponent $\omega(\sqrt{n})$ on all fixed instances.

2 Related Work

We compare localization methods by their computational costs, and complexity lower bounds by their quantifier order and error criterion.

2.1 Upper Bounds

Classical ellipsoid methods developed factor updates and controlled rounding (Bland et al., 1981; Grötschel et al., 1988). Tsitsiklis & Luo (1987, Proposition 3.1) used center-of-gravity localization to obtain a square-logarithmic communication bound in inverse accuracy for fixed dimension, without charging local computation. Karabag et al. (2021) estimated shallow-cut directions from value probes in ellipsoid coordinates. The recursive cutting-plane algorithms of Blanchard et al. (2023) trade memory for calls: their lowest-memory endpoint uses $O(d\log(1/\varepsilon))$ bits, including computation memory, and $O_d(\log^d(1/\varepsilon))$ separation calls. Kerger et al. (2024) transferred exact-oracle algorithms to inexact first-order information with quantified degradation of the error guarantee.

To obtain a round bound in our model, we must also account for complete reply scans and arithmetic within the live-storage budget. A straightforward ellipsoid implementation at common b-bit precision uses long-word products, division, and square root, with per-update work $O_d(b^2)$ using schoolbook arithmetic or $O_d(\mathsf{M}(b))$ using fast arithmetic, where $\mathsf{M}(b)$ is the multiplication cost (Brent & Zimmermann, 2010). Our construction combines short directions, dimension-dependent coefficient precision, and controlled rounding to obtain $O_d(b)$ complete per-cut bit work. Expanding search regions and geometrically increasing precision, together with serial scheduling, yield an every-round guarantee without supplied problem constants or target accuracy.

2.2 Lower Bounds

Tsitsiklis & Luo (1987, Propositions 3.2 and 4.1) proved a square-logarithmic communication lower bound under a restricted numerical-prefix protocol, and a logarithmic upper bound in one dimension using comparison messages. Sikorski & Trojan (1990) proved asymptotic near-optimality of bisection for smooth zero finding with continuous linear information. Plaskota (2023) proved that recovering the full zero set in Hausdorff distance can converge arbitrarily slowly on a fixed function. Basu et al. (2026) obtained dimension-dependent lower bounds for continuous convex feasibility with selected-bit or inner-product-sign access to separating normals, assuming a bounded feasible region and an interior-ball promise when nonempty.

The numerical-prefix and binary-feasibility lower bounds allow the hard instance to depend on the prescribed accuracy or geometric scale. Our quantifier order requires, for each competing program, one objective fixed before $n \to \infty$: the same function must enforce the lower bound along an unbounded sequence of rounds. The zero-finding results already have fixed-function quantifiers, but control root location or zero-set distance. Such distance lower bounds do not themselves give an objective-gap lower bound after integration: a derivative can be arbitrarily flat near its zero. Our nested construction maintains a monotone Lipschitz derivative and a quantitative objective gap, while keeping all past replies consistent with fixed nearest rounding. Reply length is bounded below by a constant times the refinement depth, forcing quadratic cumulative scan work.

3 Problem Setup and Computational Model

Fix $d \geq 1$, a dyadic anchor a, and a convex $F \in C^1(\mathbb{R}^d)$ with $\arg\min F \neq \varnothing$, $F^* = \min F$, and

$$\|\nabla F(x) - \nabla F(y)\|_2 \leq L\|x - y\|_2 \qquad (x, y \in \mathbb{R}^d)$$

for some finite L. Only (d, a) is supplied, with total input bit length A including the dimension field; L, minimizer distance, F^*, accuracy, and horizon are unknown. The instance (d, F, a) is fixed before $n \to \infty$.

3.1 Resource Accounting Model

A deterministic uniform finite-control program pays

$$\mathsf{work} = N_{\mathrm{read}} + N_{\mathrm{write}} + N_{\mathrm{move}}, \qquad \mathsf{space} = \sum_{\text{resident regions } r} |r|.$$

Actions are single-bit reads/writes or one-cell head moves; all input, queries, replies, history, scratch, and reports count toward storage. For b-bit operands, reading, copying, addition, and comparison cost $O(b)$ bit actions; schoolbook multiplication costs $O(b^2)$, or $O_d(b)$ when one coefficient has dimension-dependent width. These operations use $O_d(b)$ live bits (Brent & Zimmermann, 2010, Sections 1.2–1.3). There is no random access or unit-cost long-word arithmetic. Oracle evaluation and delivery are uncharged; stored buffers and their reads are charged. Memory and unfinished work persist across rounds.

3.2 Reply Protocol

A *physical round* begins with one oracle call and delivery of its reply, followed by at most $2^{12}d$ charged bit actions. These actions may process a resident reply, continue arithmetic, or prepare a future query. The complete report available at the end of round n is denoted by z_n.

A query $(x, p) \in \mathbb{Q}^d \times \mathbb{N}_0$ may depend on previously read information; x and unary 1^p must be fully stored before issuance. Normal replies satisfy

$$(\widetilde{F}, \widetilde{g}) \in (\eta_p \mathbb{Z})^{d+1}, \qquad \max\{|\widetilde{F} - F(x)|, \|\widetilde{g} - \nabla F(x)\|_\infty\} \le \eta_p, \qquad \eta_p = 2^{-8p}.$$

The tolerance is one grid spacing; nearest rounding, with error at most $\eta_p/2$, is a permitted special case. Every reply, including overflow, has encoded length

$$R(p) = 2[2 + (d+1)(16p+4)] = \Theta(d(p+1)),$$

counting all fields and framing. Encoding ranges, ties, and overflow are specified in Appendix A.4. Legal replies may depend on history but must all be compatible with one fixed F.

The next useful reply may arrive only in a round after the current reply has been fully scanned, including unused fields and framing. If the round's work budget runs out first, a *padding call* $Q_{\mathrm{pad}} = (a, 0)$ starts the next round. Its $R(0) = 8d + 12$ bits arrive in a separate buffer and are scanned/discarded at charged cost $O(d)$, leaving the useful reply intact. Padding calls count toward n, so n rounds mean exactly n oracle calls; processing one useful reply or performing one numerical update may span many rounds.

We adopt this reply protocol to make the computational cost of numerical precision explicit. Counting oracle calls alone assigns the same cost to coarse and arbitrarily precise replies. Charging for reply consumption accounts for the additional work of handling higher-precision information.

3.3 Resource Constraints

With $W = 2^{12}$ and $S = 2^{20}$, admissibility requires

$$\begin{aligned} &\mathsf{work}_{\mathrm{round}} \le Wd, \\ &\mathsf{space} \le S \left\{ \underbrace{A}_{\text{input}} + \underbrace{dp}_{\text{numerical fields}} + \underbrace{d\lceil \log_2(d+1) \rceil}_{\text{coordinate labels}} \right\}. \end{aligned} \tag{1}$$

Here p is the highest working precision of stored useful replies or stage state. Coordinate labels include a sentinel; S allows working copies and scratch. A full reply itself occupies $R(p)$ live bits and requires at least $R(p)/(Wd) = \Omega(p+1)$ rounds to consume. Exact machine and startup conventions are in Appendix A.

4 MAIN RESULTS

The upper guarantee is uniform over legal rounding; the lower bound already holds for fixed nearest rounding, even without a memory limit.

Theorem 1 (Every-round upper bound)**.** *There is one deterministic uniform finite-control program satisfying Section 3 such that, for every fixed admissible (d, F, a), there are finite instance-dependent integers $n_0 \geq 0$, $n_* \geq n_0$, a constant $C_0 > 0$, and constants $\gamma_d, C_d > 0$ depending only on dimension and the fixed budgets, for which every legal transcript satisfies*

$$F(z_n) - F^* \leq C_0 \exp[-\gamma_d \sqrt{n - n_0}] \qquad (n \geq n_*).$$

By round n, calls equal n and charged bit work is at most $2^{12}dn$. Maintained accuracy $0 < \varepsilon \leq C_0$ is attained by $n_0 + C_d \lceil \log_+(C_0/\varepsilon) \rceil^2$ calls, where $\log_+(x) = \max\{1, \log x\}$. For sufficiently small ε, the maximum working precision through the completed report attaining this accuracy obeys

$$p \leq C_d[A + \lceil \log_+(C_0/\varepsilon) \rceil].$$

The target ε is not an input to the program. To support progressively smaller errors ε, Algorithm 1 is allowed to increase the working precision p epoch by epoch during the run, thereby enlarging the available memory budget.

Theorem 2 (Fixed-instance lower bound under nearest rounding)**.** *Remove only the live-storage bound from Section 3, retaining its reply protocol and $2^{12}d$ bit actions per round. Use the fixed nearest-grid oracle of Appendix A.4, including its tie and overflow rules. For every deterministic uniform finite-control program $\mathcal{A}$ in this enlarged class and every fixed $d \geq 1$ with zero anchor, there are one fixed admissible $F = F_{\mathcal{A},d}$, constants $c = c(F_{\mathcal{A},d}, d) > 0$, $C = C(\mathcal{A}, d) > 0$, and unbounded rounds n_j such that*

$$F(z_{n_j}) - F^* \geq c \exp(-C\sqrt{n_j}).$$

Consequently, no such program guarantees an eventual every-round decay exponent $\Phi(n) = \omega(\sqrt{n})$ on every fixed instance, even with this fixed nearest-grid oracle.

The bounds match in order in n, but not in constants or dimension dependence. Both are instancewise: F is fixed before $n \to \infty$. The lower objective may depend on the competing program; neither it nor the fixed oracle rule depends on a terminal horizon. The complete upper- and lower-bound proofs are in Appendices B and C, each beginning with an overview and a diagram of the proof dependencies. Numerical illustrations appear in Appendix D, Figure 6.

5 UPPER BOUND: FINITE-PRECISION LOCALIZATION

We construct the algorithm proving Theorem 1, proceeding from its epoch schedule to reliable cuts, rounded updates, and resource bounds. Proofs and implementation details are in Appendix B.

5.1 ALGORITHM OVERVIEW

Without knowing L, minimizer distance, or target accuracy, the algorithm uses epochs $m = 1, 2, 4, \ldots$, aiming for gap $O(4^{-m})$ in $O_d((A + m)^2)$ rounds. For $d \geq 2$, it maintains $E(c, B) = \{c + By : \|y\|_2 \leq 1\}$, where $c \in \mathbb{R}^d$ is the center and the invertible $B \in \mathbb{R}^{d \times d}$ maps the unit ball to the ellipsoid. Algorithm 1 resets to $E(a, 2^m I)$, makes at most K_m cuts, and publishes the least-observed-value candidate after fully copying it. The previous report remains public during computation. Expansion eventually contains a minimizer and its neighborhood; increasing precision then makes the cuts reliable. For $d = 1$, a noisy ternary variant of classical interval elimination (Brent, 2013, Chapter 5) has the same error and cost guarantees. Figure 1 shows one two-dimensional epoch.

5.2 RELIABLE CUTS FROM SHORT DIRECTIONS

Computing $B^T \nabla F(c)$ directly would multiply long stored numbers. Instead, value probes in ellipsoid coordinates (Karabag et al., 2021) give

$$\Delta_i = \widetilde{F}(c + \tau_m B e_i) - \widetilde{F}(c) \approx \tau_m (B^T \nabla F(c))_i.$$

Algorithm 1 Finite-precision localization: epoch phases and round costs

State: Public report z; epoch candidate w and its stored value $v = \widetilde{F}(w)$.
Init: Set $z \leftarrow a$, parse the input, and set $t_d = 16(\lceil \log_2(d+1) \rceil + 1)$. $O_d(A+1)$ rounds
for each epoch $m = 1, 2, 4, \ldots$ **do**
  Phase I: allocate and reset. $O_d((A+m)^2)$ rounds
  Set $q_m = A + 4d + m + \lceil \log_2(d+1) \rceil + 1$, $b_m = 2^{12} d^3 q_m$, and $\tau_m = 2^{-b_m}$.
  Set $P_m = 2^{16} d^4 q_m$, $K_m = 64d(d+1)(m+1)$; build schedules and $(c, B) \leftarrow (a, 2^m I)$.
  Mark w empty. **Phase II: localize.**
  for $k = 0, \ldots, K_m - 1$ **do** $O_d(A+m)$ physical rounds per cut
    Set $x_0 = c$, $x_i = c + \tau_m B e_i$ $(1 \le i \le d)$; e_i is the ith coordinate vector.
    for $i = 0, \ldots, d$ **do**
      Query (x_i, P_m); fully consume the reply and store $v_i = \widetilde{F}(x_i)$.
      On overflow, retain z, clean up, and start the next epoch.
      If w is empty or $v_i < v$, set $(w, v) \leftarrow (x_i, v_i)$; ties keep w.
    end for
    Set $\Delta_i = v_i - v_0$, $D = \max_i |\Delta_i|$; if $D = 0$, go to Phase III.
    Find h with $2^{h-1} \le D < 2^h$; set $u_i = \mathrm{sgn}(\Delta_i) 2^{-t_d} \lfloor 2^{t_d - h} |\Delta_i| \rfloor$.
    Set $\rho = 1/[32(d+1)]$; compute the rounded update of Appendix B.4 satisfying:
    Containment: $\{c + By : \|y\| \le 1,\ u^T y / \|u\| \le \rho\} \subseteq E(c_+, B_+)$.
    Volume contraction: $\mathrm{vol}(E(c_+, B_+)) \le e^{-1/[16(d+1)]} \mathrm{vol}(E(c, B))$.
    Once c_+, B_+ are fully computed, set $(c, B) \leftarrow (c_+, B_+)$.
  end for
  Phase III: publish and clean up. $O_d(A+m)$ rounds
  Fully copy w into an inactive report bank, set $z \leftarrow w$, and erase obsolete state.
end for
Epoch total: $O_d((A+m)^2)$ rounds; after the onset, gap $\le C_0 4^{-m}$.
For $d = 1$, use Algorithm 3: $8m$ noisy ternary decisions at precision m.

Binary rescaling and truncation produce a *quantized cut normal* u in ellipsoid coordinates, with $O(\log(d+1))$ bits per entry (Algorithm 1, lines 8–15). The shallow-cut criterion tolerates dimension-dependent directional error (Karabag et al., 2021, Lemma 1), so this width is independent of m.

Let $E_k = E(c_k, B_k)$, $V_k = \mathrm{vol}(E_k)$, and let $\mathcal{T}_m$ be the ball of radius 2^{-m-2} around a fixed minimizer. After an instance-dependent onset, three properties hold uniformly over legal rounding (Appendix B):

1. Initially $\mathcal{T}_m \subseteq E_0 = E(a, 2^m I)$, with $V_0 / \mathrm{vol}(\mathcal{T}_m) = 2^{2d(m+1)}$.
2. If $\mathcal{T}_m \subseteq E_k$ and $F(c_k) - F^* > (L/16)4^{-m}$, the shallow cut and rounded update retain $\mathcal{T}_m$.
3. Each update contracts volume by at least the factor $e^{-1/[16(d+1)]}$, as specified in Algorithm 1.

These properties yield accuracy by the classical volume argument (Bubeck, 2015, Section 2.2). If all K_m centers had gap greater than $(L/16)4^{-m}$, the target ball would survive every cut, whereas

$$V_{K_m} \le e^{-K_m/[16(d+1)]} V_0 < \mathrm{vol}(\mathcal{T}_m),$$

a contradiction. Some center c is therefore accurate, and selecting the least observed value adds at most twice the oracle error:

$$F(z^{(m)}) - F^* \le F(c) - F^* + 2\eta_{P_m} \le (L/16)4^{-m} + 2\eta_{P_m} \le C_0\, 4^{-m}.$$

If all probe differences vanish, a point with the required accuracy has already been queried or the current center is accurate; early publication obeys the same bound (Appendix B).

5.3 Rounded Updates and Error Control

Finite rounding can move the center and distort the shape enough to exclude part of the retained region. We combine the classical factor update (Bland et al., 1981, Section 6) with controlled inflation (Grötschel et al., 1988, Sections 3.2–3.3). First, use outward interval arithmetic (Moore

et al., 2009, Chapters 2–4) to approximate the local update coefficients by dyadics with $O(\log(d+1))$ bits and inflate to absorb their errors. Then round the long center and factor entries on a grid refined with the epoch and cut indices; a second inflation absorbs these storage errors, including the final rounding. The total perturbation preserves both containment and net volume contraction in Algorithm 1 (lines 16–19; Appendix B.4). The entries of c and B are stored at increasing precision across epochs, while update coefficients retain bit widths depending only on d. Standard shift-and-add multiplication (Brent & Zimmermann, 2010, Section 1.3) therefore multiplies a b-bit entry by an update coefficient using $O_d(b)$ bit actions.

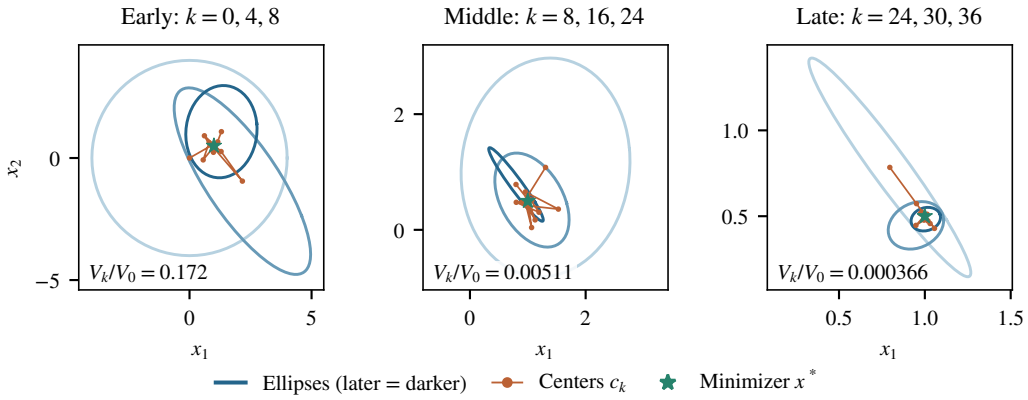

Figure 1: Ellipses during one two-dimensional epoch on a rotated quadratic. Panels zoom into one run; darker outlines mark later cuts, orange points are centers, and the green star is the minimizer. Labels give area ratios V_k/V_0. All 36 updates retain the minimizer and decrease area in this example; theoretical target-ball preservation is conditional as stated above. The reduced numerical schedule is given in Appendix D.

5.4 Resource Bounds and Anytime Convergence

The geometric guarantee becomes an every-round rate through the costs in Algorithm 1. Each cut uses $O_d(A+m)$ bit actions for query construction, $d+1$ complete scans, comparison, and update. Serial scheduling, including padding, respects the $2^{12}d$ work cap; P_m accommodates the live state. Including allocation and publication, epoch m lasts $H_m = O_d((A+m)^2)$ rounds (Appendix B.8).

Working precision is $p = P_m$ for $d \geq 2$ and $p = m$ for $d = 1$. After the onset, $p = \Theta_d(m)$ and $F(z^{(m)}) - F^* \leq C_0 e^{-\beta_d p}$ for some $\beta_d > 0$. Thus $p = O_d(A + \log(C_0/\varepsilon))$ suffices for small ε. An initial accuracy target is not permanent: later epochs raise p and attain successively smaller errors.

The epoch indices double, $1, 2, 4, \ldots$, applying geometric restarting to precision (Besson & Kaufmann, 2018, Sections 2–3). During epoch m, the algorithm keeps the completed report $z^{(m/2)}$ public until its replacement is fully written. For epoch indices $k \geq A$, the cost $H_k = O_d((A+k)^2)$ is $O_d(k^2)$. Absorbing startup and the finitely many earlier epochs into n_0, the cumulative cost through epoch m is therefore bounded by

$$O_d\left(m^2 + \left(\frac{m}{2}\right)^2 + \left(\frac{m}{4}\right)^2 + \cdots\right) = O_d(m^2).$$

Thus, at every round n in epoch m after the onset, the elapsed time and the accuracy of the public report satisfy

$$n - n_0 \leq C_d m^2, \qquad F(z_n) - F^* \leq C_0 4^{-m/2} \leq C_0 e^{-c_d\sqrt{n-n_0}}.$$

This proves Theorem 1, uniformly over legal rounding.

6 Lower Bound: A Fixed Smooth Instance

For each deterministic program, we construct one fixed smooth convex objective proving Theorem 2, even with unrestricted memory and fixed nearest rounding. Algorithm 2 describes the construction;

the following components of this section explain its geometry, compatibility with past oracle replies, and the time cost forced on the optimization algorithm. Proofs are in Appendix C. The nested-family strategy follows fixed-function adversary constructions (Sikorski & Trojan, 1990); we supply convex patches and exact nearest-rounding compatibility for the present oracle.

6.1 Construction Overview

Take $d = 1$ and anchor $a = 0$. After j refinements, only $I_j = [a_j, b_j]$ remains unresolved. Its length ℓ_j, endpoint slope magnitude s_j, and common endpoint value H_j satisfy

$$\ell_j = \tfrac{1}{2}\,64^{-j}, \quad s_j = \tfrac{1}{200}\,128^{-j}, \quad f(a_j) = f(b_j) = H_j, \quad f'(a_j) = -s_j, \quad f'(b_j) = s_j.$$

The exterior is fixed. Coarse interior queries receive a common reply without refining the interval. A precise interior query triggers an *informative event*: choose a child avoiding the query and preceding report, then fix the function outside it. Each event enforces a report gap and a longer reply scan. The oracle's construction is uncharged; the forced costs are receiver rounds.

Algorithm 2 Fixed nearest-grid construction and forced receiver costs ($d = 1$)

State: Depth j; active $I_j = [a_j, b_j]$ of length ℓ_j, endpoint slopes $\pm s_j$, and height H_j.
Init: Set $j \leftarrow 0$, $I_0 = [1/4, 3/4]$, $(\ell_0, s_0, H_0) = (1/2, 1/200, 0)$.
Rounding: RN_p is nearest rounding on $\eta_p\mathbb{Z}$, $\eta_p = 2^{-8p}$, ties toward zero.
Use the fixed normal/overflow rule of Appendix A.4 throughout.
for each useful query (x, p) with preceding report z **do**
  Phase I: test for refinement. — Uncharged oracle work
  if $x \notin I_j$ **then**
    Return the fixed nearest-grid reply from the exterior value and derivative.
  else if $\eta_p \geq 2s_j$ **then**
    Return $(\mathrm{RN}_p(H_j), 0)$; keep j.
  else
    Phase II: event j. Record delivery t_j, $n_j = t_j - 1$. — $p_j = p = \Omega(j+1)$
    Choose a child C with $x, z \notin N(C)$.
    Build g_C from Appendix C.3; set $f_C(t) = H_j + \int_{a_j}^{t} g_C(r)\,dr$.
    Freeze f_C on $I_j \setminus C$; leave the child interior unresolved.
    Return the fixed nearest-grid reply for $(f_C(x), g_C(x))$.
    Set $I_{j+1} \leftarrow C$, $\ell_{j+1} \leftarrow \ell_j/64$, $s_{j+1} \leftarrow s_j/128$.
    Set $H_{j+1} \leftarrow H_j - s_j\ell_j/4$ and $j \leftarrow j + 1$.
  end if
  Phase III: full receipt. — $\Omega(p+1)$ receiver rounds
  The next useful reply may arrive only in a round after full consumption.
end for
Before event j: $n_j \geq \sum_{i<j} \Omega(i+1) = \Omega(j^2)$.
Pre-event gap: $F(z_{n_j}) - F^* \geq s_j\ell_j/12800 = \Theta(2^{-13j})$.

6.2 Nested Patches and Report Separation

The geometric task is to build four candidates on the whole parent, matching its exterior while placing their minimizers in separated children. Set

$$C_{j,r} = a_j + \ell_j[c_r - 1/128, c_r + 1/128], \qquad (c_1, c_2, c_3, c_4) = (3/8, 11/24, 13/24, 5/8).$$

For $C = [u, v]$, its neighborhood $N(C) = [u - \ell_j/100, v + \ell_j/100]$ includes two isolation bands. The four neighborhoods are disjoint (Figure 2).

We use the characterization of differentiable convex functions by monotone derivatives (Bubeck, 2015, Section 1.2). A candidate's derivative joins $-s_j$ to $-s_j/128$ on $[a_j, u]$ and $s_j/128$ to s_j on $[v, b_j]$, using constant segments and linear ramps. Their integrals are $-s_j\ell_j/4$ and $s_j\ell_j/4$. The child interior has zero derivative integral and remains available for recursive refinement. Table 1 summarizes these derivative constraints.

The construction gives three properties (Appendix C.3):

Table 1: Derivative constraints for a candidate child $C = [u, v]$, with $s' = s_j/128$. The status column describes what is fixed after C is selected; its interior is reserved for recursive refinement.

Region	Status	Derivative data	Integral
$[a_j, u]$	Fixed	$-s_j \nearrow -s'$	$-s_j\ell_j/4$
$[u, v]$	Unfixed interior	$g_C(u) = -s',\ g_C(v) = s'$	0
$[v, b_j]$	Fixed	$s' \nearrow s_j$	$s_j\ell_j/4$

1. Each candidate preserves the parent's endpoint values and slopes. The child endpoints have equal value $H_{j+1} = H_j - s_j\ell_j/4$ and slopes $\pm s_j/128$. Preserving these data in later refinements leaves the exterior function and derivative unchanged.
2. The derivative is nondecreasing and 1-Lipschitz; the ratio s_j/ℓ_j halves with depth, preserving the global smoothness bound.
3. Every completion has its minimizer in C, and
$$t \notin N(C) \quad \Longrightarrow \quad f(t) - f^* \geq s_j\ell_j/12800.$$
Indeed, the path to the minimizer crosses a fixed segment of length $\ell_j/100$ with derivative magnitude at least $s_j/128$.

The query and previous report exclude at most two neighborhoods, so another child can always be selected. Freezing its exterior protects the current reply and gives the preceding report the stated gap. Figure 8 in Appendix D.5 illustrates three successive refinements and their compatible functions.

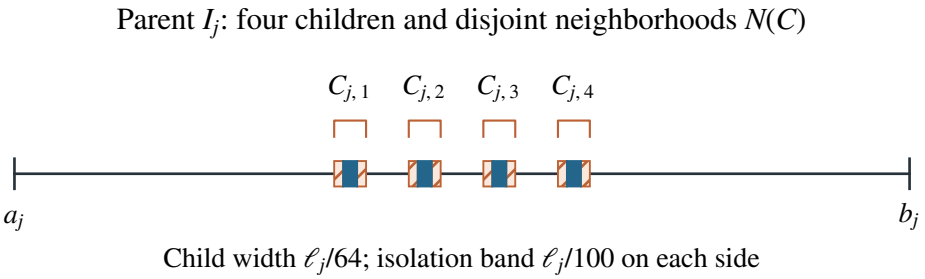

Figure 2: The parent interval I_j, four children $C_{j,r}$ (blue), and isolation bands (hatched). Each bracket marks the neighborhood $N(C)$ formed by one child and its two bands. The four neighborhoods are disjoint, so the query and preceding report exclude at most two choices.

6.3 Compatible Replies and a Fixed Objective

Later refinements must preserve every earlier reply under the same nearest-grid rule. At the time of a query (x, p), denote the active depth by k and let h be any later completion. Set $\widehat{H}_k = \mathrm{RN}_p(H_k)$. Algorithm 2 distinguishes three cases: $x \notin I_k$; $x \in I_k$ with $\eta_p \geq 2s_k$; and $x \in I_k$ with $\eta_p < 2s_k$.

For an exterior query, $h(x)$ and $h'(x)$ are already fixed. For a precise interior query, selecting a child that avoids x freezes these data before replying. In both cases, all later refinements leave the true data, and hence their fixed rounded reply, unchanged (lines 4 and 13).

For a coarse interior query, the common reply is $(\widehat{H}_k, 0)$ (line 7). Convexity and the preserved endpoint data give $h(x) \in [H_k - s_k\ell_k, H_k]$ and $|h'(x)| \leq s_k$. The constants ensure that this entire value interval lies strictly inside the rounding cell centered at $\widehat{H}_k$, with $s_k < \eta_p/2$ (Lemma 19, Appendix C.6). Consequently,

$$|\widehat{H}_k - h(x)| < \eta_p/2, \qquad |h'(x)| < \eta_p/2,$$
$$\mathrm{RN}_p(h(x)) = \widehat{H}_k, \qquad \mathrm{RN}_p(h'(x)) = 0.$$

Thus the reply remains exactly the oracle's nearest-grid answer even when x lies in later unresolved intervals.

If refinement continues, I_k shrinks to a point α and $s_k \to 0$. The derivative patches converge uniformly to a nondecreasing 1-Lipschitz derivative, whose integral is a fixed smooth convex f minimized at α (Appendix C.5). If refinement stops, any remaining completion realizes the transcript. The compatibility argument applies to the entire adaptive run; for general d, embed $F(x) = f(x_1)$. The objective is fixed independently of any terminal horizon.

6.4 Precision Costs and the Rate Lower Bound

As in numerical-prefix communication lower bounds (Tsitsiklis & Luo, 1987, Proposition 3.2), we charge the precision needed for informative observations. At event j, the condition $\eta_{p_j} < 2s_j$ requires $p_j > (7j + \log_2 100)/8 = \Omega(j+1)$. Its $R(p_j)$-bit reply takes $\Omega(p_j + 1)$ receiver rounds under the $2^{12}d$ work cap. Before round $n_j = t_j - 1$, all earlier event replies have been fully consumed. Their disjoint scan costs and the report separation therefore give (Appendix C.8)

$$n_j \geq \sum_{i<j} \Omega(i+1) = \Omega(j^2), \qquad F(z_{n_j}) - F^* \geq \frac{2^{-13j}}{5{,}120{,}000} \geq ce^{-C\sqrt{n_j}}.$$

Infinitely many events give an unbounded subsequence. If only J events occur, four completions induce the same infinite transcript and, by determinism, the same reports. Each report misses at least three minimizer neighborhoods, so one fixed completion has gap at least $s_J \ell_J / 12800 > 0$ infinitely often (Appendix C.9). Both cases prove Theorem 2.

7 Discussion and Limitations

The upper and lower bounds identify $\exp[-\Theta(\sqrt{n})]$ as the sharp order of eventual convergence guarantees for each fixed dimension under the stated computational model. This root-exponential order remains sharp in cumulative charged bit work $\mathcal{B}_n$ (Proposition 24, Appendix C.11). The lower bound holds even with unrestricted working memory and a fixed nearest-grid oracle. Its time obstruction relies on serial consumption of complete replies: increasing precision forces $R(p)/(Wd) = \Omega(p)$ receiver rounds. Allowing algorithms to skip fields, overwrite unread buffers, or stop reading a reply before its end can change this cost; whether the same sharp order persists under these interfaces remains open.

The results concern eventual behavior, with an instance-dependent onset. Their constants and dimension dependence do not match, and the upper construction incurs substantial storage and arithmetic costs as dimension grows. Improving these costs may require sharper numerical invariants or a more compact localization representation. Sharper dimension dependence, the full time–space tradeoff, and extensions from deterministic to randomized programs remain open.

AI Use Statement

The initial research was conducted by a fully autonomous in-house research agent system powered by GPT-5.6 Sol and GPT-6 Astra. The human authors' initial contribution was to pose the task of finding and proving sharp convergence rates for first-order optimization algorithms in the finite-bit source setting used in this paper. Starting from this task, the agent carried out the entire initial literature search, formulated the mathematical claims, developed and wrote the proofs, and produced the manuscript draft. The human authors subsequently used GPT-6 Astra to assist in verifying and correcting the results, refining the arguments, and preparing the final manuscript. The human authors take responsibility for the final content of this work, including all AI-generated text, claims, proofs, and code.

Reproducibility Statement

To support independent verification of the theoretical results, we specify the computational model and resource accounting conventions in Section 3 and Appendix A, and provide complete proofs

of the upper and lower bounds in Appendices B and C. The provided supplementary materials include experiment code, recorded data, pinned dependency versions, and scripts for rerunning the experiments and regenerating the figures. We also provide certificates and independent checkers for validating finite experimental runs using exact arithmetic. The experimental settings and cost conventions are documented in Appendix D. Appendix E introduces the supplementary materials, and the accompanying README files provide setup, validation, and reproduction commands.

REFERENCES

Amitabh Basu, Phillip Kerger, and Marco Molinaro. Tight lower bounds for binary first-order oracles for convex optimization, 2026. URL https://arxiv.org/abs/2511.02082.

Lilian Besson and Emilie Kaufmann. What doubling tricks can and can't do for multi-armed bandits, 2018. URL https://arxiv.org/abs/1803.06971.

Moïse Blanchard, Junhui Zhang, and Patrick Jaillet. Memory-constrained algorithms for convex optimization via recursive cutting-planes, 2023. URL https://arxiv.org/abs/2306.10096.

Robert G. Bland, Donald Goldfarb, and Michael J. Todd. Feature article—the ellipsoid method: A survey. *Operations Research*, 29(6):1039–1091, 1981. doi: 10.1287/opre.29.6.1039. URL https://pubsonline.informs.org/doi/10.1287/opre.29.6.1039.

Richard P. Brent. *Algorithms for minimization without derivatives*. Courier Corporation, 2013.

Richard P. Brent and Paul Zimmermann. *Modern Computer Arithmetic*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2010.

Sébastien Bubeck. Convex optimization: Algorithms and complexity, 2015. URL https://arxiv.org/abs/1405.4980.

Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, 1988. ISBN 3-540-13624-X, 0-387-13624-X (U.S.).

Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, second edition, 2002. ISBN 0-89871-521-0.

Mustafa O. Karabag, Cyrus Neary, and Ufuk Topcu. Smooth convex optimization using sub-zeroth-order oracles, 2021. URL https://arxiv.org/abs/2103.00667.

Phillip Kerger, Marco Molinaro, Hongyi Jiang, and Amitabh Basu. A universal transfer theorem for convex optimization algorithms using inexact first-order oracles. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 23532–23546. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/kerger24a.html.

Ramon E. Moore, R. Baker Kearfott, and Michael J. Cloud. *Introduction to Interval Analysis*. Society for Industrial and Applied Mathematics, 2009. doi: 10.1137/1.9780898717716. URL https://epubs.siam.org/doi/abs/10.1137/1.9780898717716.

Leszek Plaskota. In the search for all zeros of smooth functions. *Journal of Applied and Numerical Analysis*, 1:72–79, 2023. doi: 10.30970/ana.2023.1.72. URL https://api.semanticscholar.org/CorpusID:267225093.

K. Sikorski and G. M. Trojan. Asymptotic near optimality of the bisection method. *Numerische Mathematik*, 57(1):421–433, Dec 1990. ISSN 0945-3245. doi: 10.1007/BF01386421. URL https://doi.org/10.1007/BF01386421.

John N. Tsitsiklis and Zhi Quan Luo. Communication complexity of convex optimization. *Journal of Complexity*, 3(3):231–243, September 1987. ISSN 0885-064X. doi: 10.1016/0885-064X(87)90013-6.

A Encoding and Machine Conventions

This appendix defines how data are represented, how oracle records are exchanged, and how storage and computation are charged. The material is organized in the order needed to assemble a complete execution:

1. Define general encodings for integers, rational numbers, dyadic numbers, and vectors (Appendix A.1).
2. Use these formats for the input, queries, reports, and numerical replies (Appendices A.2–A.4), including the exact reply length $R(p)$.
3. Specify the machine budgets and reply scheduling, then prove that the upper algorithm's input conversion has linear work and storage bounds (Appendices A.5–A.6).

The upper proof uses these rules to implement its algorithm; the lower proof uses mandatory complete-reply scans to force elapsed time.

A.1 General Data Encodings

All programs use the same parsing rules for the binary formats below, so each permitted record can be decoded unambiguously. Sign bits and markers for the binary point, coordinates, and vectors have constant size. Every bit, including markers and length prefixes, counts toward record length and storage; reading or writing it is charged. For a binary string w, $|w|$ denotes its length in bits.

1. *Integer encodings.* For a signed integer $N \in \mathbb{Z}$ and a positive integer $D \in \mathbb{Z}_{\geq 1}$, define

$$\mathsf{Int}(N) = \sigma 1^{\ell} 0 \,\mathrm{bin}_{\ell}(|N|), \quad |\mathsf{Int}(N)| = 2\ell + 2,$$
$$\mathsf{Pos}(D) = 1^{r} 0 \,\mathrm{bin}_{r}(D), \quad |\mathsf{Pos}(D)| = 2r + 1,$$

   where

$$\sigma = \mathbf{1}_{\{N<0\}}, \qquad \ell = \lceil \log_2(|N|+1) \rceil, \qquad r = \lceil \log_2(D+1) \rceil.$$

   Here juxtaposition denotes concatenation of bit strings, $\mathrm{bin}_k(q)$ is the k-bit unsigned binary representation of q, and 1^k is a string of k ones. The prefix 1^k0 tells the reader to consume exactly k subsequent magnitude bits. The sign bit σ is one for negative integers and zero otherwise; the encoding Pos omits the sign bit. These canonical integer encodings omit redundant leading zeroes. Zero has $\sigma = \ell = 0$, an empty magnitude field, and $\mathsf{Int}(0) = 00$.

2. *Rational numbers and vectors.* A rational number N/D is encoded by concatenating its signed numerator and positive denominator:

$$C = \mathsf{Int}(N)\mathsf{Pos}(D), \qquad |C| = (2\ell+2) + (2r+1) = 2\ell + 2r + 3.$$

   For $d \geq 1$ such records $C_i = \mathsf{Int}(N_i)\mathsf{Pos}(D_i)$, the vector encoding is

$$\mathsf{Vec}(N, D) = 10\, C_1\, 01\, C_2\, 01 \cdots 01\, C_d\, 11.$$

   It represents $(N_i/D_i)_{i=1}^{d}$, with start marker 10, separator 01 between adjacent coordinates, and end marker 11.

3. *Explicit binary-fraction representation.* A dyadic number has the form $N2^{-b}$, with $N \in \mathbb{Z}$ and $b \in \mathbb{N}_0$. Its explicit binary-fraction format is

   TAG SIGN INTEGER RADIX FRACTION END.

   If the sign bit is s, the integer field represents I, and the fractional field represents the integer f in b bits, the value is

$$(-1)^s (I + f2^{-b}).$$

   Each integer or fractional binary digit is stored as a two-bit token, $0 \mapsto 00$ and $1 \mapsto 01$; SIGN is one bit immediately after TAG. The number b counts fractional digits, so the fractional field occupies $2b$ stored bits. The digits are explicit: an exponent cannot replace a string of zeroes. Zero may have empty integer and fractional fields.

For a dyadic vector $x = (x_1, \ldots, x_d)$, let X_i be the complete coordinate record above for x_i. The vector is encoded in coordinate order as

$$\mathsf{DyadVec}(x) = \mathsf{VSTART}\ X_1\ \mathsf{SEP}\ X_2\ \cdots\ \mathsf{SEP}\ X_d\ \mathsf{VEND}.$$

Here VSTART and VEND mark the vector's beginning and end, and SEP separates adjacent coordinate records. The fixed marker codes are

$$\mathsf{TAG} = 1000, \quad \mathsf{RADIX} = 1001, \quad \mathsf{END} = 1010,$$
$$\mathsf{VSTART} = 1011, \quad \mathsf{SEP} = 1100, \quad \mathsf{VEND} = 1101.$$

Outside the single sign bit, an initial zero starts a two-bit digit token and an initial one starts a four-bit marker. A sequential finite-state parser therefore distinguishes digits and markers without knowing field lengths or using lookahead beyond the current token. Every token and marker bit counts toward length and storage.

The vector formats above specify coordinate records and delimiters; the dimension is read from its separate input record. Finite binary-fraction records represent dyadic values, whereas numerator-denominator records represent arbitrary rational values. Both formats permit redundant leading-zero digits and either sign for zero; every additional bit counts toward stored length and incurs the same read and write costs as other bits.

A.2 Input Encoding

The input supplies a dimension $d \geq 1$ and a dyadic anchor $a \in \mathbb{Q}^d$. These are stored as the two records $\mathsf{Pos}(d)$ and $\mathsf{DyadVec}(a)$, using the general encodings in Appendix A.1.

Both records count toward input length and live storage, and their reads are charged. Their lengths are

$$A_d = |\mathsf{Pos}(d)| = 2\lceil \log_2(d+1) \rceil + 1,$$
$$A_{\mathrm{src}} = |\mathsf{DyadVec}(a)|, \qquad A = A_{\mathrm{src}} + A_d.$$

For example, $d = 3$ has binary digits 11, so $\mathsf{Pos}(3) = 110\,11$ and the dimension field occupies $A_d = 5$ bits.

A.3 Query and Report Encodings

A query specifies a rational point $x \in \mathbb{Q}^d$ and a precision $p \in \mathbb{N}_0$; a report specifies a rational point z_n. Their vector records use the formats in Appendix A.1: dyadic vectors may use either format, and vectors with non-dyadic coordinates use the numerator-denominator format. The dimension d is obtained from the input and is not repeated in these vector records.

1. *Queries.* A query consists of a fully stored rational vector and a unary precision field 1^p, a string of p ones (empty for $p = 0$). In the numerator-denominator format, its record is

$$Q = \mathsf{Vec}(N, D)\,10\,1^p 0\,11.$$

For $\ell_i^Q = \lceil \log_2(|N_i| + 1) \rceil$ and $r_i^Q = \lceil \log_2(D_i + 1) \rceil$, its exact length is

$$|Q| = \sum_{i=1}^{d} (2\ell_i^Q + 2r_i^Q + 3) + 2(d-1) + p + 9.$$

This exact length applies to the displayed format without additional padding.

2. *Reports.* A report consists of the complete vector record for z_n in one of the permitted formats. It has no query precision field. Initially, the anchor record is selected as the report, so $z_0 = a$. The rules for replacing this record are specified in Appendix A.5.

A.4 Numerical Reply Encoding

The numerical interface has three parts.

1. *Accuracy and range.* A normal reply approximates $F(x)$ and all d gradient coordinates. For each true scalar field y, the allowed rounded values are

$$\eta_p = 2^{-8p}, \quad B_p = 2^{8p}, \qquad \mathcal{R}_p(y) = \{q \in \eta_p\mathbb{Z} \cap [-B_p, B_p] : |q - y| \le \eta_p\}.$$

   Selection from this set may depend on history. Nearest rounding has error at most $\eta_p/2$ in range and is one allowed choice; directed rounding may use the larger tolerance (Brent & Zimmermann, 2010, Section 3.1.9).

2. *Reply format and length.* Logical bits are framed by $0 \mapsto 00$, $1 \mapsto 01$; the terminal marker is 11. The logical payload contains

$$\underbrace{\mathsf{status}}_{1\text{ bit}} \quad \underbrace{\mathsf{value}, \mathsf{gradient}_1, \ldots, \mathsf{gradient}_d}_{(d+1)(16p+4)\text{ bits}}.$$

   Each scalar field has one sign bit and $16p + 3$ unsigned magnitude bits. A normal field encodes $m2^{-8p}$ with $|m| \le 2^{16p}$. Status zero means normal; status one means overflow and has all-zero fields. Both records have the same length:

$$R(p) = \underbrace{2[1 + (d+1)(16p+4)]}_{\text{framed payload}} + \underbrace{2}_{\text{terminal}} = 2[2 + (d+1)(16p+4)]. \tag{2}$$

3. *Overflow.* If $M = \max\{|F(x)|, |\partial_1 F(x)|, \ldots, |\partial_d F(x)|\}$, then

$$\text{overflow permitted} \implies M \ge B_p - \eta_p, \qquad M > B_p + \eta_p \implies \text{overflow required.}$$

   Every reply, including history-dependent rounding and overflow, must remain compatible with the same objective F.

The lower-bound construction in Section 6 proves Theorem 2 using the following fixed nearest-grid oracle. Let $\mathrm{RN}_p(y)$ be the nearest multiple of η_p, with ties toward zero:

$$\text{reply} = \begin{cases} \text{normal, with fields } \mathrm{RN}_p(y), & \text{if all rounded fields lie in } [-B_p, B_p], \\ \text{overflow, with all-zero fields,} & \text{otherwise.} \end{cases}$$

Zero uses the positive-sign encoding. This specialization is legal because

$$\begin{aligned} |\mathrm{RN}_p(y) - y| &\le \eta_p/2 && \text{for every normal field,} \\ |\mathrm{RN}_p(y)| > B_p &\implies |y| > B_p + \eta_p/2, \\ |y| > B_p + \eta_p &\implies |\mathrm{RN}_p(y)| > B_p. \end{aligned}$$

It depends only on the true value and gradient and on p.

A.5 Machine Resources and Reply Scheduling

The resource measures count different objects:

$$\begin{aligned} &\text{calls through round } n = n, \\ &\quad \text{charged work} = \text{bit reads} + \text{bit writes} + \text{one-cell head moves}, \\ &\quad \text{live storage} = \text{all resident bits, including unread replies.} \end{aligned}$$

One physical round is

$$\underbrace{\text{one call and complete delivery}}_{\text{external}} \quad \longrightarrow \quad \underbrace{\text{at most } Wd \text{ bit actions}}_{\text{charged}}, \qquad W = 2^{12}.$$

Memory and unfinished computation persist between rounds, so one arithmetic operation or reply scan may span many rounds.

For a deterministic uniform finite-control program, the budgets are

$$\begin{aligned} &\mathcal{B}_n = N_{\mathrm{read}}(n) + N_{\mathrm{write}}(n) + N_{\mathrm{move}}(n), \quad \mathcal{B}_n - \mathcal{B}_{n-1} \le Wd, \\ &S_{\mathrm{live}} \le S\{A + d[p + \lceil \log_2(d+1) \rceil]\}, \qquad W = 2^{12}, \quad S = 2^{20}. \end{aligned}$$

The accounting rules are:

- The machine has a fixed number of sequential binary tapes or tracks and heads, independent of d and the epoch index. Fixed selector bits choose the query and report banks; changing a selector is charged. Selecting a tape does not move its head, and every actual head move is charged separately.
- Each bit read, bit write, and one-cell head move costs one action. Validation, copying, and arithmetic expand into such actions; there is no random access or unit-cost long-word arithmetic.
- The working precision p is the largest precision of a resident useful buffer or precision-stage state. It may increase during the run; a padding call does not lower it while that state remains stored. A stage may declare its precision by a complete unary ruler 1^t0. Its declared precision becomes t only after the terminal zero is written; the ruler, its construction, and any replacement all count.
- All buffers, rulers, history, scratch, addressing state, stale capacity, and simultaneous old/new state count as live storage.
- Query, precision, recursion, and report choices may use all legally read information. The model prescribes neither a search subspace nor a precision schedule.

Table 2 gives standard serial arithmetic costs (Brent & Zimmermann, 2010, Sections 1.2–1.3). Appendix B.8 also counts head movement and region selection.

Table 2: Bit work and live storage for standard serial operations on b-bit integers. Constants in $O_d(\cdot)$ may depend on dimension.

Operation on b-bit integers	Bit actions	Live bits
Read, copy, add, or compare	$O(b)$	$O(b)$
Multiply (schoolbook)	$O(b^2)$	$O(b)$
Multiply by a coefficient of width depending only on d	$O_d(b)$	$O_d(b)$

The lifetime of a reply and report follows three rules.

1. *Delivery and consumption.* Each round delivers one complete first-order reply. Evaluation and delivery writing are external, but the full buffer immediately counts as storage. A useful reply remains resident until its entire framed word, including the terminal marker, is consumed. Scanned storage is erased after processing; the next useful reply can arrive only in a later round.
2. *Intervening rounds.* If the round's work budget is exhausted before the current useful reply has been fully scanned, the next round must start with the padding query $(a, 0)$. Its short reply uses a separate buffer and cannot overwrite the useful reply. Resume the suspended work after handling that padding reply. Padding calls may also provide further rounds for local computation after the useful scan is complete.
3. *Startup and publication.* At time zero the immutable anchor is both selected query and report: $z_0 = a$. The first round calls $(a, 0)$ before any input-sized copy. Subsequent input processing, including decoding $\mathsf{Pos}(d)$, is charged under the same rules. Keep a complete stored report selected until its replacement has been fully written; the anchor remains the permanent padding query. Appendix A.6 bounds the startup conversion cost for the algorithm constructed to prove Theorem 1.

A.6 Format Conversion

The algorithm constructed to prove Theorem 1 first decodes the dimension record and then converts the input anchor from its explicit binary-fraction representation to the numerator-denominator vector representation before localization. This subsection bounds the converted length, work, and simultaneous storage for this startup.

For an anchor coordinate $a_i = N_i 2^{-b_i}$ with b_i fractional digits, the converted record stores the denominator $D_i = 2^{b_i}$ explicitly in binary. With $\ell_i^a = \lceil \log_2(|N_i| + 1) \rceil$ and $r_i^a = b_i + 1$, the

converted anchor has length

$$A_{\text{list}} = \underbrace{2}_{\text{start}} + \underbrace{\sum_{i=1}^{d}(2\ell_i^a + 2r_i^a + 3)}_{\text{coordinate records}} + \underbrace{2(d-1)}_{\text{separators}} + \underbrace{2}_{\text{end}}.$$

The lemma below proves the bound

$$A_{\text{list}} \le 4(A_{\text{src}} + d) \le 4(A + d) \le 4[A + d\lceil \log_2(d+1)\rceil].$$

When the original and converted records coexist, their combined length $A_{\text{src}} + A_{\text{list}}$ counts toward live storage. The dimension record, both anchor records, and all parsing and conversion scratch are charged under the memory budget in Appendix A.5, with the original input-length parameter A and coefficient $S = 2^{20}$.

Lemma 3 (Charged input conversion)**.** *Let $d \ge 1$ and let the input consist of valid records $\mathsf{Pos}(d)$ and $\mathsf{DyadVec}(a)$ in the formats of Appendix A.1, with exactly d anchor coordinates. Write $A = A_d + A_{\text{src}}$ for their combined bit length. A serial finite-control transducer decodes the dimension and converts the anchor to the canonical rational-list representation without changing its value. Let T_{convert} count its charged input processing, excluding the separately charged padding-reply scans. The output length and combined decoding and conversion work satisfy*

$$A_{\text{list}} \le 4(A_{\text{src}} + d) \le 4(A + d), \qquad T_{\text{convert}} \le 128(A + d).$$

The source, output, and conversion scratch use $O(A + d)$ live bits. Let N_{startup} be the number of physical rounds from the first oracle call through completion of dimension decoding and anchor conversion, including all padding rounds. After each padding reply has been fully processed, the program executes one charged transition of the conversion procedure. Allowing one additional charged handover transition at its end gives $N_{\text{startup}} \le 128(A + d) + 1$.

Proof. First parse $\mathsf{Pos}(d) = 1^r 0\,\text{bin}_r(d)$, where $r = \lceil \log_2(d+1)\rceil$ and $A_d = 2r + 1$. We specify the layout and count its primitive operations explicitly. Use three groups of sequential binary tracks:

- The input group has a data track containing $\mathsf{Pos}(d)$ and a track marking its current head position.
- The ruler group has a unary-mark track and a head-position track.
- The output group has a binary-digit track, a validity track, and a head-position track. A validity bit of one marks a stored digit; zero marks a boundary.

Each track has one head, so there are seven heads, independently of d. Within each group, heads occupy the same index between operations. To move a group by one position, clear its old head mark, move each of its heads by one cell, and write the new head mark. Thus a group move costs

$$M_{\text{input}} = M_{\text{ruler}} = 2 \text{ writes} + 2 \text{ moves} = 4,$$
$$M_{\text{output}} = 2 \text{ writes} + 3 \text{ moves} = 5.$$

These are sequences of charged primitives, not unit-cost moves. The fixed controller records which primitive to resume if a round ends during such a sequence.

The parser performs the following steps.

1. All heads start at index zero. Write the three head marks, a zero ruler sentinel, and a zero output-validity boundary. Advance the ruler and output groups to index one.
2. For each prefix one, read it, write a ruler mark, and advance both the input and ruler groups. On reading the prefix's terminating zero, advance the input group to the first magnitude bit and move the ruler group left to its last mark.
3. Read the current ruler mark. If it is one, read and copy one input magnitude bit, set its output-validity bit to one, and erase the ruler mark. Advance input and output by one position, move the ruler left by one, and repeat.

4. The first zero read on the ruler is its initial sentinel. Write a zero output-validity boundary after the last digit and move the output group left once, leaving it at the least significant digit.

The controller holds at most one copied bit; no numerical counter or uncharged jump is used. Table 3 counts every read, write, and individual head move in these steps.

Table 3: Primitive operation counts for decoding $\mathsf{Pos}(d)$. Reads, writes, moves, and actions are per repetition; writes include head marks, validity bits, and ruler erasure.

Operation	Repetitions	Reads	Writes	Moves	Actions
Initialize marks and boundaries; advance ruler and output	1	0	9	5	14
Process one prefix one	r	1	5	4	10
Process the prefix's terminating zero	1	1	4	4	9
Copy one magnitude bit and consume a ruler mark	r	2	9	7	18
Read the sentinel, close the output, and move back one position	1	1	3	3	7

For example, a prefix-one step uses one read, one ruler write, and two group moves, hence $1 + 1 + 4 + 4 = 10$ actions. A magnitude step uses two reads, three data/validity writes, and three group moves, hence $2 + 3 + 4 + 4 + 5 = 18$ actions. Initialization uses five writes and two group moves, $5 + 4 + 5 = 14$; termination uses one read, one write, and an output group move, $1 + 1 + 5 = 7$. Summing the table gives

$$N_{\mathrm{read}}^{\mathrm{dim}} = 3r + 2, \qquad N_{\mathrm{write}}^{\mathrm{dim}} = 14r + 16, \quad N_{\mathrm{move}}^{\mathrm{dim}} = 11r + 12,$$
$$T_{\mathrm{dim}} = 28r + 30 \le 32(2r + 1) = 32A_d.$$

The input heads visit indices $0, \ldots, 2r + 1$, including one cell beyond the input record; ruler and output heads visit $0, \ldots, r + 1$. Counting every visited cell on all seven tracks, including head marks and blank boundary cells, gives

$$S_{\mathrm{dim}}^{\mathrm{tracks}} \le 2(2r + 2) + (2 + 3)(r + 2) = 9r + 14.$$

The fixed parsing/resumption flags are counted once in the 64-bit allowance in the combined storage calculation below. The table totals can also be checked column by column:

$$\begin{aligned} N_{\mathrm{read}}^{\mathrm{dim}} &= 0 + r + 1 + 2r + 1, \\ N_{\mathrm{write}}^{\mathrm{dim}} &= 9 + 5r + 4 + 9r + 3, \\ N_{\mathrm{move}}^{\mathrm{dim}} &= 5 + 4r + 4 + 7r + 3, \\ 32A_d - T_{\mathrm{dim}} &= 32(2r + 1) - (28r + 30) = 36r + 2 \ge 0. \end{aligned}$$

After j magnitude iterations, the output contains exactly the first j binary digits of d and the ruler has exactly $r - j$ marks left. This invariant holds at $j = 0$; one iteration copies the next digit and erases exactly one mark. At $j = r$ the sentinel ends the loop. Thus the completed binary record represents d and supplies the dimension used by the anchor conversion below.

The input encoding is the explicit dyadic grammar specified in Appendix A.1. Each coordinate contains an explicit signed integer payload and fractional payload, separated by fixed delimiters. The program scans those delimiters to obtain payload lengths; it is not supplied with an uncharged length ruler. A coordinate with sign s, integer payload I, fractional payload f and fractional length b represents

$$(-1)^s(I + f2^{-b}).$$

The conversion below preserves the numerical value even when the input contains leading zeroes or signed zero. Every marker and payload bit is read, written and moved over at charged cost.

To compare the source and converted lengths, let u_i, b_i be the numbers of integer and fractional binary digits of source coordinate i. Their two-bit tokens occupy $2u_i$ and $2b_i$ stored bits. Its sign and

coordinate markers occupy c_i bits, and the vector markers occupy v_d bits, where

$$c_i = |\mathsf{TAG}| + |\mathsf{SIGN}_i| + |\mathsf{RADIX}| + |\mathsf{END}| = 13,$$
$$v_d = |\mathsf{VSTART}| + (d-1)|\mathsf{SEP}| + |\mathsf{VEND}| = 4d + 4,$$
$$L_i = 2u_i + 2b_i + c_i.$$

Thus the two record lengths, before comparing their payloads, are

$$A_{\text{src}} = \sum_{i=1}^{d} (2u_i + 2b_i + c_i) + v_d,$$

$$A_{\text{list}} = 2 + \sum_{i=1}^{d} (2\ell_i^a + 2r_i^a + 3) + 2(d-1) + 2.$$

Here ℓ_i^a and r_i^a are the binary magnitude lengths of the converted numerator and denominator, respectively.

The anchor conversion in Table 4 uses four groups of tracks, in addition to the stored dimension:

- The source group has three tracks: input data, a coordinate-start flag, and a head-position mark. Its one-position moves cost 2 writes plus 3 head moves, hence 5 actions.
- Each of two reusable rulers has a mark track and a head-position track. They record ℓ_i^a and b_i using unary marks, with a zero sentinel at index zero. Each ruler move costs 4 actions.
- The output group has data and head-position tracks. Its moves cost 4 actions; emitting one bit costs $1 + 4 = 5$. The output is delimited by its own vector and integer encodings.

All nine heads start at index zero. Initialization writes four head marks and two ruler sentinels, costing 6 actions. At the start of each coordinate, advance both rulers to index one and write a zero there on the b_i ruler, costing $4 + 4 + 1 = 9$ actions. The controller stores the sign and parsing phase in fixed finite control. It reads both bits of each digit token and all four bits of each marker at charged cost. A ruler mark or output digit is produced only after the corresponding two-bit source token has been read.

Here is the primitive accounting behind the coordinate rows.

1. First pass and return. During PASS1, read each source bit, write its start flag (one only on the first bit of the coordinate), and advance the source: $1 + 1 + 5 = 7$ actions per bit. Appending an ℓ_i^a-ruler mark costs $1 + 4 = 5$; appending a b_i-ruler mark also writes a new zero right boundary, costing $1 + 4 + 1 = 6$. The latter happens once per fractional digit; the former happens once per payload digit from the first one onward, skipping markers. The source ends one position past the coordinate. In REWIND, move left first and then read the start flag, $5 + 1 = 6$ actions per position, until the flag is one.

2. Numerator header and payload. After the first pass, the length-ruler head is one position past its last mark. Emit the canonical sign, move that ruler left once, and repeat: read a mark, emit one, erase the mark, and move left. Each repetition costs $1 + 5 + 1 + 4 = 11$. At the zero sentinel, read it and emit the prefix's terminating zero. The fixed cost is therefore $5 + 4 + 1 + 5 = 15$, and the ruler is reset at index zero. The payload pass reads and advances over all L_i source bits, costing $6L_i$, and emits the ℓ_i^a significant digits, costing $5\ell_i^a$.

3. Denominator and ruler cleanup. Keep the b_i marks for two traversals: a backward traversal emits the remaining b_i ones of the denominator-length prefix; a forward traversal emits its b_i trailing magnitude zeroes. A final backward traversal erases the marks and returns to the left sentinel. The respective per-mark costs are

$$(1 + 5 + 4)b_i + (1 + 5 + 4)b_i + (1 + 1 + 4)b_i = 26b_i.$$

The fixed 30 actions consist of emitting three bits (the extra prefix one, its terminating zero, and the leading magnitude one), three initial ruler moves, and three boundary reads:

$$3 \cdot 5 + 3 \cdot 4 + 3 \cdot 1 = 30.$$

Table 4: Anchor-conversion transitions and additive bit-action budgets, with the dimension-decoding cost from Table 3. Sum each budget over the executions indicated; *each* i means $i = 1, \ldots, d$. Internal scans repeat to their stated boundary, and SEP returns to TAG for the next coordinate. Every return traversal and ruler cleanup is included.

State	Operation / next state	Executions	Bit actions
DIM	Decode Pos(d); INIT	1	$\leq 32A_d$
INIT	Initialize tracks; consume vector start and emit 10; TAG	1	$6\lvert\mathsf{VSTART}\rvert + 16$
TAG	Prepare the two coordinate rulers; PASS1	each i	9
PASS1	Scan the whole coordinate, recording sign, start flag, and both lengths; repeat to coordinate end; HEADER	each i	$7L_i + 5\ell_i^a + 6b_i$
HEADER	Emit sign and $1^{\ell_i^a}0$ while consuming the length ruler; REWIND	each i	$11\ell_i^a + 15$
REWIND	Move left and test the start flag until the coordinate start; PAYLOAD	each i	$6L_i$
PAYLOAD	Rescan to coordinate end, copying only the significant payload suffix; DENOM	each i	$6L_i + 5\ell_i^a$
DENOM	Emit $1^{b_i+1}0\,10^{b_i}$; clear and reset the b_i ruler; SEP	each i	$26b_i + 30$
SEP	On a coordinate separator, consume it and emit 01; return to TAG	$d-1$	$6\lvert\mathsf{SEP}\rvert + 10$
SEP	On vector end, consume it and emit 11; DONE	1	$6\lvert\mathsf{VEND}\rvert + 10$
DONE	Halt	1	0

Both rulers now point to their zero sentinels and can be reused for the next coordinate. These steps also apply when $b_i = 0$ or $\ell_i^a = 0$.

Outside coordinates, each vector-marker bit costs one source read and one source-group move, hence 6 actions; each emitted framing bit costs 5. The vector contributes v_d input-marker bits and $2d + 2$ output-framing bits.

For coordinate i, concatenating the decoded integer and fractional digits produces the magnitude $I2^{b_i} + f$ and hence

$$N_i = (-1)^{s_i}(I2^{b_i} + f), \qquad N_i 2^{-b_i} = (-1)^{s_i}(I + f2^{-b_i}).$$

Deleting leading zeroes gives $\ell_i^a \leq u_i + b_i$, and the literal denominator 2^{b_i} has $r_i^a = b_i + 1$ bits. Consequently

$$\begin{aligned} A_{\text{list}} &= \sum_{i=1}^{d}(2\ell_i^a + 2r_i^a + 3) + 2d + 2 \\ &\leq \sum_{i=1}^{d}(2u_i + 4b_i + 5) + 2d + 2 \\ &\leq 4\sum_{i=1}^{d}(u_i + b_i + c_i) + 4d \\ &\leq 4(A_{\text{src}} + d). \end{aligned}$$

For the penultimate inequality, each coordinate contributes

$$4(u_i + b_i + c_i) - (2u_i + 4b_i + 5) = 2u_i + 4c_i - 5 \geq 0,$$

and $4d - (2d + 2) = 2(d - 1) \geq 0$. For the last inequality, the explicit source encoding gives $A_{\text{src}} = \sum_i (2u_i + 2b_i + c_i) + v_d \geq \sum_i (u_i + b_i + c_i)$. Thus the length comparison also covers empty numerator payloads, leading zeroes, and zero fractional length.

It remains to account for the implementation and startup:

1. Scanning and integer encoding. The first pass records b and the first nonzero digit. Concatenating the decoded digits gives $|N| = I2^b + f$ without long-word arithmetic. The output sign and numerator length are

$$(\sigma, \ell) = \begin{cases} (0,0), & I = f = 0, \\ (s, \text{length of the suffix starting at the first one}), & \text{otherwise.} \end{cases}$$

Each payload and marker bit is visited only a bounded number of times.

2. Work and simultaneous storage. Summing Table 4 except its DIM row gives the anchor work. Since $\ell_i^a, b_i \le L_i$ and $A_{\mathrm{src}} = \sum_i L_i + v_d$,

$$\begin{aligned} T_{\mathrm{anchor}} &\le \sum_{i=1}^{d} (19L_i + 21\ell_i^a + 32b_i + 54) + 6v_d + 10d + 16 \\ &\le 72 \sum_{i=1}^{d} L_i + 6v_d + 64d + 16 \\ &\le 128(A_{\mathrm{src}} + d), \\ T_{\mathrm{convert}} &= T_{\mathrm{dim}} + T_{\mathrm{anchor}} \\ &\le 32A_d + 128(A_{\mathrm{src}} + d) \le 128(A + d). \end{aligned}$$

The coefficient sums used above are

$$7 + 6 + 6 = 19, \qquad 5 + 11 + 5 = 21, \qquad 6 + 26 = 32, \qquad 9 + 15 + 30 = 54.$$

The vector-marker rows contribute $6v_d + 10d + 16$. Since $A_{\mathrm{src}} = \sum_i L_i + v_d$, the slack in the displayed anchor-work bound is explicitly

$$128(A_{\mathrm{src}} + d) - \big(72 \textstyle\sum_i L_i + 6v_d + 64d + 16\big) = 56 \displaystyle\sum_i L_i + 122v_d + 64d - 16 \ge 0.$$

Likewise $32A_d \le 128A_d$ proves the combined conversion bound.

Let $\ell_{\max} = \max_i \ell_i^a$ and $b_{\max} = \max_i b_i$. For storage, first count tracks and then reserve 64 fixed flag bits for the parser and its suspended primitive. The dimension tracks occupy $9r + 14$ cells, as counted above. Therefore

$$\begin{aligned} S_{\mathrm{source+output+scratch}} &\le 3(A_{\mathrm{src}} + 1) + 2(A_{\mathrm{list}} + 1) \\ &\quad + 2(\ell_{\max} + 2) + 2(b_{\max} + 2) + (9r + 14) + 64 \\ &\le 15A_{\mathrm{src}} + 8d + 9r + 91. \end{aligned}$$

The last line uses $A_{\mathrm{list}} \le 4(A_{\mathrm{src}} + d)$ and $\ell_{\max}, b_{\max} \le A_{\mathrm{src}}$: the source-length coefficient is $3 + 2 \cdot 4 + 2 + 2 = 15$, and the fixed terms sum to $3 + 2 + 4 + 4 + 14 + 64 = 91$. The rulers are reused, so their capacity depends on the largest coordinate and is not summed over all coordinates.

3. Scheduling. The work T_{convert} counts input-processing primitives; padding reply processing is charged separately within each round. Use a padding-buffer group with data, a left-boundary flag, and a head-position mark, so a group move costs 5 actions. For each reply bit, read it, write its boundary flag (one only at the first bit), erase the data bit, and advance, costing $1 + 1 + 1 + 5 = 8$. The two-bit reply framing identifies the terminal 11 without knowing d. Then move left and test the boundary flag until back at the start, costing 6 actions per position. Table 5 includes initialization and the suspended input-processing or handover primitive. Its last column is directly additive. Consequently each padding round uses at most

$$14R(0) + 2 = 112d + 170 \le 282d < 2^{12}d.$$

The reusable padding buffer occupies $3[R(0) + 1] = 24d + 39$ cells. Adding it to the preceding storage bound, in which the 64 flags have already been counted, gives

$$\begin{aligned} S_{\mathrm{live,start}} &\le 15A_{\mathrm{src}} + 32d + 9r + 130 \\ &\le 16A + 256d < 2^{20}(A + d) \le 2^{20}\{A + d\lceil \log_2(d+1) \rceil\}. \end{aligned}$$

Table 5: Charged work in a padding round. The same reusable handler is used during startup and all later epochs. Here $R_0 = R(0) = 8d + 12$.

Step	Repetitions	Actions each	Total actions
Initialize the padding-buffer head mark	1	1	1
Read, mark, erase, and advance one reply bit	R_0	8	$8R_0$
Move back one cell and test its boundary flag	R_0	6	$6R_0$
Execute one suspended charged primitive	1	1	1
Total			$14R_0 + 2$

Indeed, $A = A_{\mathrm{src}} + 2r + 1$, and the difference in the second step is

$$16A + 256d - (15A_{\mathrm{src}} + 32d + 9r + 130) = A_{\mathrm{src}} + 23r + 224d - 114 \geq 0,$$

because $A_{\mathrm{src}} \geq 21$ and $d, r \geq 1$. The final step uses $r = \lceil \log_2(d+1) \rceil \geq 1$. The round-work comparison is also uniform in dimension: $282d - (112d + 170) = 170(d-1) \geq 0$. Thus startup is legal already at working precision zero. Executing one conversion primitive after each complete padding scan gives

$$N_{\mathrm{startup}} \leq T_{\mathrm{convert}} + 1 \leq 128(A + d) + 1 = O(A + d).$$

□

B Proof of the Upper Bound

This chapter constructs the program in Theorem 1. After an overview and the elementary inequalities, the proof has three parts:

1. *Geometry for $d \geq 2$.* Define the epoch schedule, certify the rounded update, and show that short directions retain a target ball whenever the center is inaccurate. Volume contraction then gives the completed-epoch error.
2. *The scalar branch.* Use interval elimination when $d = 1$, where the matrix update is undefined.
3. *Implementation and convergence.* Count stored regions and charged operations, schedule them in physical rounds, and combine epoch cost with accuracy.

The estimates to be combined are

$$\text{epoch cost} = O_d((A + m)^2), \qquad \text{completed-epoch gap} = O(4^{-m}).$$

They yield the root-exponential rate and the precision needed for a target error.

B.1 Overview and Proof Dependencies

An epoch is one complete localization attempt. Its index m runs through $1, 2, 4, \ldots$ and controls both the expanding initial search region and the numerical precision. Fix a minimizer x^* for the analysis and set

$$\mathcal{T}_m = \{x : \|x - x^*\|_2 \leq 2^{-m-2}\}.$$

The program does not know this ball. The proof uses it as a witness: if every queried center were inaccurate, every cut would retain $\mathcal{T}_m$, but the enclosing ellipsoid would eventually have smaller volume than the ball itself. Therefore a good queried point must exist.

On a finite machine, two implications need separate proofs:

1. Short coefficients and storage rounding must preserve containment and volume decrease for any supplied short direction.
2. Noisy value probes must produce a valid direction; this estimate uses the size bounds established in the first step.

Resource bounds are then combined with accuracy, as shown below.

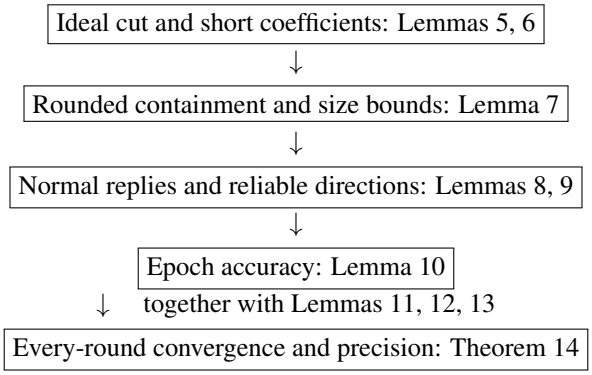

Figure 3: Dependencies in the upper-bound proof. The geometric branch handles $d \geq 2$; Lemma 11 supplies the $d = 1$ branch. Resource accounting and serial scheduling are needed for both branches.

B.2 Elementary Analytic and Matrix Facts

We first recall the problem assumptions and notation, then introduce matrix quantities and prove the inequalities used in localization.

The unknown continuously differentiable objective satisfies

$$F : \mathbb{R}^d \to \mathbb{R}, \qquad F \in C^1, \qquad F^* = F(x^*) = \min_x F(x),$$
$$F((1-t)x + ty) \leq (1-t)F(x) + tF(y) \quad (0 \leq t \leq 1),$$
$$\|\nabla F(x) - \nabla F(y)\|_2 \leq L\|x - y\|_2 \quad \text{for some finite } L.$$

The last two inequalities state convexity and gradient Lipschitz continuity. Here ∇F is the vector of first partial derivatives. The program receives neither L, x^*, F^*, a target error, nor a stopping horizon. If z_n denotes the report after round n, its error is $F(z_n) - F^*$.

We use the following notation:

- $\|v\|_2 = (\sum_i v_i^2)^{1/2}$ and $\|v\|_\infty = \max_i |v_i|$. An unmarked vector norm is Euclidean; e_i has coordinate i equal to one and all other coordinates zero.
- $\lceil t \rceil$ and $\lfloor t \rfloor$ round up and down to integers. $\log_2$ is the binary logarithm and $\log$ the natural logarithm.
- Constants in $O_d(\cdot)$ and $\Omega_d(\cdot)$ may depend on dimension and the fixed budgets, but not on epochs, rounds, or the unknown objective.

For a matrix B, its operator norm is $\|B\|_{\mathrm{op}} = \sup_{\|y\|_2=1} \|By\|_2$. Its largest and smallest singular values are, equivalently,

$$\sigma_{\max}(B) = \sup_{\|y\|_2=1} \|By\|_2, \qquad \sigma_{\min}(B) = \inf_{\|y\|_2=1} \|By\|_2.$$

If $\sigma_{\min}(B) > 0$, then B is invertible and $\|B^{-1}\|_{\mathrm{op}} = 1/\sigma_{\min}(B)$. An ellipsoid in factor form is $E(c, B) = \{c + By : \|y\|_2 \leq 1\}$; its volume is $v_d|\det B|$, where v_d is the volume of the unit ball. These definitions describe the geometry without requiring a positive-definite matrix representation of the ellipsoid.

Lemma 4 (Inequalities used in localization)**.** *Let $F : \mathbb{R}^d \to \mathbb{R}$ be convex and differentiable, let its gradient be L-Lipschitz for $L \geq 0$, and let x^* attain its global minimum. For all $x, y, h \in \mathbb{R}^d$,*

$$F(y) \geq F(x) + \langle \nabla F(x), y - x \rangle, \tag{3}$$
$$|F(x+h) - F(x) - \langle \nabla F(x), h \rangle| \leq \tfrac{L}{2}\|h\|_2^2, \tag{4}$$
$$0 \leq F(x) - F^* \leq \tfrac{L}{2}\|x - x^*\|_2^2. \tag{5}$$

For nonzero $v, w \in \mathbb{R}^d$ and matrices $M, E \in \mathbb{R}^{d\times d}$, we also have

$$\left\| \frac{v}{\|v\|} - \frac{w}{\|w\|} \right\| \le \frac{2\|v-w\|}{\|w\|}, \tag{6}$$

$$\sigma_{\min}(M+E) \ge \sigma_{\min}(M) - \|E\|_{\mathrm{op}}, \tag{7}$$

$$|\det(I+E)| \le (1+\|E\|_{\mathrm{op}})^d. \tag{8}$$

Proof. For $0 < t \le 1$, convexity gives $[F(x+t(y-x)) - F(x)]/t \le F(y) - F(x)$. Letting $t \downarrow 0$ proves (3). The fundamental theorem of calculus and gradient Lipschitz continuity give

$$F(x+h) - F(x) - \langle \nabla F(x), h\rangle = \int_0^1 \langle \nabla F(x+th) - \nabla F(x), h\rangle \, dt,$$

$$\left| \int_0^1 \langle \nabla F(x+th) - \nabla F(x), h\rangle \, dt \right| \le \int_0^1 Lt\|h\|^2 \, dt = \tfrac{L}{2}\|h\|^2.$$

For each coordinate i and every $t > 0$, minimality gives

$$\frac{F(x^* + te_i) - F(x^*)}{t} \ge 0, \qquad \frac{F(x^* - te_i) - F(x^*)}{t} \ge 0.$$

Taking $t \downarrow 0$ yields $\partial_i F(x^*) \ge 0$ and $-\partial_i F(x^*) \ge 0$, hence $\nabla F(x^*) = 0$. Apply (4) with $x = x^*$ and $h = x - x^*$; the linear term is zero and $F(x) \ge F(x^*)$, so

$$0 \le F(x) - F(x^*) = |F(x) - F(x^*) - \langle \nabla F(x^*), x - x^*\rangle| \le \tfrac{L}{2}\|x - x^*\|^2.$$

This proves (5). These are the standard convexity and smoothness inequalities (Bubeck, 2015, Sections 1.2 and 3.2).

For normalization, add and subtract $v/\|w\|$ and use the reverse triangle inequality:

$$\left\| \frac{v}{\|v\|} - \frac{w}{\|w\|} \right\| \le \frac{|\|w\| - \|v\||}{\|w\|} + \frac{\|v-w\|}{\|w\|} \le \frac{2\|v-w\|}{\|w\|}.$$

For every unit vector y, $\|(M+E)y\| \ge \|My\| - \|Ey\| \ge \sigma_{\min}(M) - \|E\|_{\mathrm{op}}$, which proves (7) by taking an infimum. Finally, the volume of a parallelepiped is at most the product of its edge lengths. Applying this to the columns of $I+E$ gives $|\det(I+E)| \le \prod_i \|(I+E)e_i\| \le (1+\|E\|_{\mathrm{op}})^d$. For background on matrix norms and perturbation theory, see Higham (2002, Chapters 6–7). □

B.3 The Epoch Algorithm and Its Parameters

For every dimension write $\ell_d = \lceil \log_2(d+1) \rceil$. For $d \ge 2$, define the following integers and dyadic scales:

$$t_d = 16(\ell_d + 1),$$

$$q_m = A + 4d + m + \ell_d + 1, \qquad b_m = 2^{12} d^3 q_m,$$

$$P_m = 2^{16} d^4 q_m, \qquad \tau_m = 2^{-b_m},$$

$$K_m = 64d(d+1)(m+1), \qquad \rho = \frac{1}{32(d+1)}.$$

The index m runs through powers of two. The quantities b_m and P_m control probe displacement and oracle precision; K_m is the maximum number of cuts. All loop bounds and precision headers are stored as unary rulers, whose construction is included in the resource count.

The schedule underlying Algorithm 1 has four steps:

1. *Initialize.* Set $(c, B) = (a, 2^m I)$ and empty the candidate bank.
2. *Probe and compare, for $k = 0, \dots, K_m - 1$.* Query in order

$$x_0 = c, \qquad x_i = c + \tau_m B e_i \quad (1 \le i \le d)$$

at precision P_m. Consume every reply completely, and store the query with least observed value, retaining the earlier stored candidate on a tie. An overflow in a useful query aborts

the epoch and preserves the previous report. Lemma 8 shows that, for each fixed objective, all useful replies in all sufficiently large epochs are normal, uniformly over legal rounding choices; hence such aborts occur in only finitely many epochs. For normal replies write $v_i = \widetilde{F}(x_i)$ and

$$\Delta_i = v_i - v_0, \qquad D = \max_i |\Delta_i|.$$

3. *Form the cut and update.* If $D = 0$, proceed directly to the publication step. Otherwise choose $h \in \mathbb{Z}$ with $2^{h-1} \le D < 2^h$ and form the quantized cut normal

$$u_i = \operatorname{sgn}(\Delta_i) 2^{-t_d} \left\lfloor 2^{t_d - h} |\Delta_i| \right\rfloor. \tag{9}$$

To implement this formula, subtract the observed values on their common 2^{-8P_m} grid and scan D for its highest nonzero bit. If that bit has weight 2^{h-1}, copy from each $|\Delta_i|$ the t_d bit positions with weights $2^{h-1}, \dots, 2^{h-t_d}$, padding missing positions with zero, and attach its sign. These copied bits are the fractional digits of $|u_i|$; no multiplication by a growing-length scaling factor is needed. Thus $1/2 \le \max_i |u_i| < 1$ and $u \ne 0$. Apply the dyadic construction in Lemma 7 to compute a new center c_+ and factor B_+ satisfying

$$\{c + By : \|y\| \le 1,\ u^T y/\|u\| \le \rho\} \subseteq E(c_+, B_+),$$
$$\operatorname{vol}(E(c_+, B_+)) \le e^{-1/[16(d+1)]} \operatorname{vol}(E(c, B)).$$

Once both new records are fully written, replace $(c, B) \leftarrow (c_+, B_+)$. If $k + 1 < K_m$, return to the probing step with cut index $k + 1$; otherwise proceed to publication.

4. *Publish.* After at most K_m updates, copy the complete candidate to the inactive report bank and publish it. The previous report remains public until this copy is complete.

Start with $m = 1$. After either publication or an overflow abort, replace m by $2m$ and return to initialization. All arithmetic and scans can be paused between bit actions; intervening physical rounds use the padding call $(a, 0)$.

B.4 Rounded Ellipsoid Updates

The update is proved in three steps. First construct a real-arithmetic ellipsoid with a known volume decrease. Next approximate only its local coefficients at a precision depending on d. Finally, compensate for both coefficient error and storage rounding by two small inflations. The resulting containment statement holds for any short direction u; the accuracy of that direction is addressed afterward.

Lemma 5 (Ideal shallow update)**.** *Let $d \ge 2$, let $u \in \mathbb{R}^d \setminus \{0\}$ be any nonzero direction, and put $e = u/\|u\|$ and $\rho = 1/[32(d+1)]$. Set*

$$a_d = \frac{d}{\sqrt{d^2 - 1}}, \quad b_d = \frac{d}{d+1}, \quad v = -\frac{e}{d+1}, \quad M = \sqrt{\lambda}\,[a_d I + (b_d - a_d) e e^T], \quad \lambda = 1 + \frac{8\rho}{d}.$$

Then

$$\{y : \|y\| \le 1,\ e^T y \le \rho\} \subseteq E(v, M),$$
$$\log\det M \le -\frac{3}{8(d+1)}, \qquad \frac{2}{3} \le \sigma_{\min}(M) \le \sigma_{\max}(M) < 2.$$

Proof. Fix y with $\|y\| \le 1$ and $e^T y \le \rho$. We first prove the containment assertion by decomposing y along the unit vector e.

1. Orthogonal decomposition. Set $s = e^T y$ and $y_\perp = y - se$. Since $e^T e = 1$,

$$\begin{aligned} e^T y_\perp &= e^T y - s\, e^T e = s - s = 0, \\ \|y\|^2 &= (se + y_\perp)^T (se + y_\perp) \\ &= s^2 e^T e + 2s\, e^T y_\perp + \|y_\perp\|^2 \\ &= s^2 + \|y_\perp\|^2. \end{aligned}$$

Consequently $\|y_\perp\|^2 = \|y\|^2 - s^2 \le 1 - s^2$. Also $|s| \le \|e\|\|y\| \le 1$, so $-1 \le s \le \rho$.

2. Action of the central-cut factor. For the classical factor $M_0 = a_d I + (b_d - a_d)ee^T$ (Bland et al., 1981, Section 6), direct multiplication gives

$$M_0 e = a_d e + (b_d - a_d)e(e^T e) = b_d e,$$
$$M_0 y_\perp = a_d y_\perp + (b_d - a_d)e(e^T y_\perp) = a_d y_\perp.$$

The second identity holds for every vector orthogonal to e. Since $\mathbb{R}^d = \mathrm{span}\{e\} \oplus e^\perp$ and $a_d, b_d > 0$, the factor M_0 is invertible. Applying its inverse to the two identities yields

$$M_0^{-1} e = b_d^{-1} e, \qquad M_0^{-1} y_\perp = a_d^{-1} y_\perp.$$

Since $v = -e/(d+1)$, it follows that

$$y - v = \left(s + \frac{1}{d+1}\right) e + y_\perp,$$
$$M_0^{-1}(y - v) = \frac{s + 1/(d+1)}{b_d} e + \frac{1}{a_d} y_\perp$$
$$= \frac{(d+1)s + 1}{d} e + \frac{\sqrt{d^2 - 1}}{d} y_\perp.$$

3. Squared norm and its upper bound. Expanding the square, and then using $e^T e = 1$ and $e^T y_\perp = 0$, gives

$$\|M_0^{-1}(y - v)\|^2 = \frac{((d+1)s+1)^2}{d^2}\|e\|^2 + \frac{2((d+1)s+1)\sqrt{d^2-1}}{d^2} e^T y_\perp + \frac{d^2-1}{d^2}\|y_\perp\|^2$$
$$= \frac{((d+1)s+1)^2}{d^2} + \frac{d^2-1}{d^2}\|y_\perp\|^2$$
$$\le \frac{((d+1)s+1)^2}{d^2} + \frac{d^2-1}{d^2}(1 - s^2).$$

The inequality uses $\|y_\perp\|^2 \le 1 - s^2$ and $d^2 - 1 > 0$. Expanding its right-hand side gives

$$\frac{((d+1)s+1)^2 + (d^2-1)(1-s^2)}{d^2}$$
$$= \frac{(d+1)^2 s^2 + 2(d+1)s + 1 + d^2 - 1 - (d^2-1)s^2}{d^2}$$
$$= \frac{d^2 + 2(d+1)s + 2(d+1)s^2}{d^2}$$
$$= 1 + \frac{2(d+1)}{d^2}(s + s^2).$$

For $-1 \le s \le 0$, $s + s^2 \le 0$; for $0 \le s \le \rho$, $s + s^2 \le \rho(1+\rho)$. Since $(d+1)/d \le 3/2$ and $\rho \le 1$,

$$\frac{2(d+1)(1+\rho)}{d} \le 2 \cdot \frac{3}{2} \cdot 2 = 6 \le 8.$$

It follows that

$$\|M_0^{-1}(y - v)\|^2 \le 1 + \frac{2(d+1)}{d^2}\rho(1+\rho) \le 1 + \frac{8\rho}{d} = \lambda.$$

Finally, $M = \sqrt{\lambda} M_0$ implies

$$\|M^{-1}(y - v)\|^2 = \lambda^{-1}\|M_0^{-1}(y - v)\|^2 \le 1.$$

By the definition of $E(v, M)$, this proves $y \in E(v, M)$.

We next compute the volume change and bound the singular values.

1. Determinant before inflation. The eigenvalue b_d occurs once, in direction e; the eigenvalue a_d occurs $d-1$ times, on $e^\perp$. Since the determinant is the product of these eigenvalues,

$$\det M_0 = b_d a_d^{d-1}.$$

Write the two factors as

$$b_d = (1+d^{-1})^{-1}, \qquad a_d = (1-d^{-2})^{-1/2}.$$

Taking logarithms and using $1-d^{-2} = (1-d^{-1})(1+d^{-1})$ gives

$$\begin{aligned}\log\det M_0 &= \log b_d + (d-1)\log a_d \\ &= -\log(1+d^{-1}) - \frac{d-1}{2}\log(1-d^{-2}) \\ &= -\frac{d}{2}\log(1-d^{-2}) + \frac{1}{2}\log(1-d^{-2}) - \log(1+d^{-1}) \\ &= -\frac{d}{2}\log(1-d^{-2}) + \frac{1}{2}\big[\log(1-d^{-1}) - \log(1+d^{-1})\big].\end{aligned}$$

2. Bounding the log determinant. For $0 < t < 1$, the geometric series for $1/(1-s)$ and $1/(1+s)$ converge uniformly on $[0,t]$: the absolute value of the kth term is at most t^k, and $\sum_{k\geq 0} t^k < \infty$. Consequently they may be integrated term by term, giving

$$-\log(1-t) = \int_0^t \frac{ds}{1-s} = \sum_{r=1}^{\infty} \frac{t^r}{r},$$

$$\log(1+t) = \int_0^t \frac{ds}{1+s} = \sum_{k=1}^{\infty} \frac{(-1)^{k+1} t^k}{k}.$$

Thus the even powers cancel in their difference:

$$\begin{aligned}\frac{1}{2}[\log(1-t) - \log(1+t)] &= -\frac{1}{2}\sum_{k=1}^{\infty} \frac{[1+(-1)^{k+1}]t^k}{k} \\ &= -\sum_{r=1}^{\infty} \frac{t^{2r-1}}{2r-1}.\end{aligned}$$

Since $d \geq 2$, these series converge absolutely at $t = d^{-1}$ and $t = d^{-2}$. Applying the first series at d^{-2} gives

$$-\frac{d}{2}\log(1-d^{-2}) = \frac{d}{2}\sum_{r=1}^{\infty}\frac{d^{-2r}}{r} = \sum_{r=1}^{\infty}\frac{d^{-(2r-1)}}{2r}.$$

Combining this with the odd-power series gives

$$\begin{aligned}\log\det M_0 &= \sum_{r=1}^{\infty}\left(\frac{1}{2r} - \frac{1}{2r-1}\right) d^{-(2r-1)} \\ &= -\sum_{r=1}^{\infty}\frac{d^{-(2r-1)}}{2r(2r-1)} \\ &= -\frac{1}{2d} - \sum_{r=2}^{\infty}\frac{d^{-(2r-1)}}{2r(2r-1)} \\ &\leq -\frac{1}{2d} \leq -\frac{1}{2(d+1)}.\end{aligned}$$

The first inequality drops a sum of nonpositive terms.

3. Inflation and net volume decrease. Multiplying a d-dimensional matrix by $\sqrt{\lambda}$ multiplies its determinant by $\lambda^{d/2}$. Therefore

$$\det M = \lambda^{d/2}\det M_0, \qquad \log\det M = \log\det M_0 + \frac{d}{2}\log\lambda.$$

For $x \geq 0$, $\log(1+x) \leq x$. Substituting $\lambda = 1 + 8\rho/d$ and $\rho = 1/[32(d+1)]$ gives

$$\frac{d}{2}\log\lambda = \frac{d}{2}\log\left(1+\frac{8\rho}{d}\right) \leq \frac{d}{2}\frac{8\rho}{d} = 4\rho = \frac{1}{8(d+1)}.$$

Consequently

$$\log\det M \leq -\frac{1}{2(d+1)} + \frac{1}{8(d+1)} = -\frac{3}{8(d+1)}.$$

Translation by v does not change volume, and M has positive determinant. Thus

$$\frac{\mathrm{vol}(E(v,M))}{\mathrm{vol}(E(0,I))} = \det M \leq e^{-3/[8(d+1)]}.$$

4. Singular-value bounds. The matrix M is symmetric positive definite, so its singular values equal its positive eigenvalues. Since $b_d < 1 < a_d$, the smallest and largest are $\sqrt{\lambda}b_d$ and $\sqrt{\lambda}a_d$, respectively. For $d \geq 2$,

$$b_d = \frac{1}{1+d^{-1}} \geq \frac{1}{1+1/2} = \frac{2}{3},$$
$$a_d^2 = \frac{1}{1-d^{-2}} \leq \frac{1}{1-1/4} = \frac{4}{3},$$
$$1 \leq \lambda = 1 + \frac{1}{4d(d+1)} \leq 1 + \frac{1}{4\cdot 2\cdot 3} = \frac{25}{24}.$$

It follows that

$$\sigma_{\min}(M) = \sqrt{\lambda}b_d \geq b_d \geq \frac{2}{3},$$
$$\sigma_{\max}(M) = \sqrt{\lambda}a_d \leq \sqrt{\frac{25}{24}\frac{4}{3}} = \sqrt{\frac{25}{18}} < 2.$$

Together these give the stated singular-value bounds.

□

Lemma 6 (Short, certified update coefficients)**.** *Let $d \geq 2$, $\ell_d = \lceil \log_2(d+1) \rceil$, and $t_d = 16(\ell_d + 1)$. Let u be a quantized direction satisfying*

$$u \in (2^{-t_d}\mathbb{Z})^d, \qquad \frac{1}{2} \leq \|u\|_\infty \leq 1.$$

The direction constructed in (9) *satisfies these conditions. Let v, M be the ideal update coefficients defined in Lemma 5 for $e = u/\|u\|$. Put*

$$r_\theta = \left\lceil \log_2(2^{12}d(d+1)) \right\rceil, \quad \theta_d = 2^{-r_\theta}, \quad \epsilon_d = \frac{\theta_d}{2^{12}d^4}, \quad N = 64(\ell_d+1).$$

There is a finite rational interval calculation producing dyadics $\bar{v}, \bar{M}$ such that

$$\|\bar{v} - v\|_\infty \leq \epsilon_d, \qquad \max_{i,j} |\bar{M}_{ij} - M_{ij}| \leq \epsilon_d.$$

Every operand and output coefficient has $O(\log(d+1))$ bits, uniformly over all quantized directions u specified above. All arithmetic operates on words of this length.

Proof. We first bound the exact data u, Q, then construct enclosing intervals, propagate their widths, and verify the word-length and work bounds.

First verify the conditions for the algorithm's direction. In (9), every component is an integer multiple of 2^{-t_d}. Since $D > 0$ and $2^{h-1} \leq D < 2^h$,

$$2^{t_d-1} \leq 2^{t_d-h}D < 2^{t_d}.$$

The endpoints 2^{t_d-1} and 2^{t_d} are integers. Taking floors therefore gives

$$2^{t_d-1} \leq \left\lfloor 2^{t_d-h}D \right\rfloor \leq 2^{t_d} - 1.$$

The maximum absolute component of u is obtained at a component attaining $|\Delta_i| = D$. Thus

$$\frac{1}{2} \le \|u\|_\infty = 2^{-t_d} \left\lfloor 2^{t_d - h} D \right\rfloor \le 1 - 2^{-t_d} < 1.$$

Now take any u satisfying the lemma's hypotheses and define $Q = \sum_{i=1}^{d} u_i^2$. Writing $u_i = k_i 2^{-t_d}$ with $k_i \in \mathbb{Z}$ gives

$$Q = 2^{-2t_d} \sum_{i=1}^{d} k_i^2,$$

so Q is an exactly representable dyadic number. Its bounds follow directly from the assumed maximum component:

$$Q \ge \max_i u_i^2 = \|u\|_\infty^2 \ge (1/2)^2 = \frac{1}{4},$$
$$Q = \sum_{i=1}^{d} u_i^2 \le \sum_{i=1}^{d} \|u\|_\infty^2 = d\|u\|_\infty^2 \le d.$$

The coefficients in Lemma 5 can therefore be expressed in terms of u and Q:

$$v_i = -\frac{u_i}{(d+1)\sqrt{Q}}, \qquad M_{ij} = \sqrt{\lambda}\left[\frac{d}{\sqrt{d^2-1}}\mathbf{1}_{i=j} + \left(\frac{d}{d+1} - \frac{d}{\sqrt{d^2-1}}\right)\frac{u_i u_j}{Q}\right].$$

1. Certified scalar intervals. The four quantities in Table 6 have the claimed initial brackets because

$$1 \le \lambda = 1 + \frac{1}{4d(d+1)} \le \frac{25}{24} < 4, \quad Q^{-1/2} \le 2, \quad (d^2-1)^{-1/2} \le 1, \quad Q^{-1} \le 4.$$

All four quantities are positive. We apply rational bisection (Brent, 2013, Chapter 4) to each of them. For a target $z \in [l, r]$, let $t = (l + r)/2$. The test in the table is equivalent to $t \le z$; thus replace $[l, r]$ by $[t, r]$ if the test holds and by $[l, t]$ otherwise. In both cases the new interval contains z and its width is $(r - l)/2$. Induction gives width $(r_0 - l_0)2^{-N}$ after N steps. Squaring is order-preserving here because $t, z \ge 0$; all denominators are positive, so clearing them also preserves the test's direction. All comparisons are therefore exact integer comparisons.

Table 6: The four bisection loops used to construct the short coefficients. Each loop has $N = 64(\ell_d + 1)$ iterations, where $\ell_d = \lceil \log_2(d + 1) \rceil$. Bracket widths in the last column are before final outward rounding.

Target	Initial bracket	Test for t below target	Final width
$\sqrt{\lambda}$	$[1, 2]$	$t^2 \le \lambda$	2^{-N}
$Q^{-1/2}$	$[0, 2]$	$Qt^2 \le 1$	2^{1-N}
$(d^2-1)^{-1/2}$	$[0, 1]$	$(d^2-1)t^2 \le 1$	2^{-N}
Q^{-1}	$[0, 4]$	$Qt \le 1$	2^{2-N}

Write the resulting intervals as $\mathcal{S}$, $\mathcal{R}_Q$, $\mathcal{R}_D$, and $\mathcal{R}_{Q,1}$, in table order. Define

$$G = 2N + 8\ell_d, \qquad \gamma = 2^{-G}, \qquad \mathrm{out}_\gamma([l, r]) = \left[\gamma \left\lfloor \frac{l}{\gamma} \right\rfloor, \gamma \left\lceil \frac{r}{\gamma} \right\rceil\right].$$

This outward rounding contains $[l, r]$ and enlarges its width by at most 2γ (Moore et al., 2009, Chapters 2–4). The initial bracket endpoints are integers, so after N bisections every endpoint is a multiple of 2^{-N}. Since $G \ge N$, these endpoints already lie on the γ mesh; rounding them leaves them unchanged. In particular

$$\mathcal{S} \subset [1, 2], \quad \mathcal{R}_Q \subset [0, 2], \quad \mathcal{R}_D \subset [0, 1], \quad \mathcal{R}_{Q,1} \subset [0, 4].$$

For subsequent calculations we allow the same 2γ rounding margin at every operation.

2. Interval operations and their valid ranges. For an interval $X = [l, r]$, put $\mathrm{w}(X) = r - l$ and $\mathrm{a}(X) = \max(|l|, |r|)$. Interval addition and multiplication mean the smallest enclosing interval of the corresponding sums and products. Then

$$\begin{aligned} \mathrm{w}(X + Y) &= \mathrm{w}(X) + \mathrm{w}(Y), \\ \mathrm{w}(cX) &= |c|\mathrm{w}(X), \\ \mathrm{w}(XY) &\le \mathrm{a}(X)\mathrm{w}(Y) + \mathrm{a}(Y)\mathrm{w}(X). \end{aligned}$$

For the last inequality, any $x, x' \in X$ and $y, y' \in Y$ satisfy

$$|xy - x'y'| = |x(y - y') + y'(x - x')| \le \mathrm{a}(X)\mathrm{w}(Y) + \mathrm{a}(Y)\mathrm{w}(X).$$

Apply this to the largest and smallest products. Also

$$\mathrm{w}(\mathrm{out}_\gamma(X)) \le \mathrm{w}(X) + 2\gamma,$$

since each endpoint moves outward by at most γ.

Construct the following intervals in the displayed order. A scalar in an interval expression denotes its singleton interval, and $\chi_{ij} = \mathbf{1}_{i=j}$ is either zero or one:

$$\begin{aligned} \mathcal{A} &= \mathrm{out}_\gamma(d\mathcal{R}_D) \cap [0, 2], \\ \mathcal{B} &= \mathrm{out}_\gamma\left(\frac{d}{d+1}\right) \cap [0, 1], \\ \mathcal{H} &= \mathrm{out}_\gamma(\mathcal{B} - \mathcal{A}) \cap [-2, 0], \\ \mathcal{G}_{ij} &= \mathrm{out}_\gamma(u_i u_j \mathcal{R}_{Q,1}) \cap [-1, 1], \\ \mathcal{P}_{ij} &= \mathrm{out}_\gamma(\mathcal{H}\mathcal{G}_{ij}), \\ \mathcal{J}_{ij} &= \mathrm{out}_\gamma(\chi_{ij}\mathcal{A} + \mathcal{P}_{ij}) \cap [-2, 2], \\ \mathcal{M}_{ij} &= \mathrm{out}_\gamma(\mathcal{S}\mathcal{J}_{ij}), \\ V_i &= \mathrm{out}_\gamma\left(-\frac{u_i}{d+1}\mathcal{R}_Q\right). \end{aligned}$$

To justify every intersection, the exact enclosed quantities satisfy

$$\begin{aligned} &a_d \in [0, 2], \qquad b_d \in [0, 1], \qquad b_d - a_d \in [-2, 0], \\ &\left|\frac{u_i u_j}{Q}\right| \le 1, \qquad |(M_0)_{ij}| = |e_i^T M_0 e_j| \le \|M_0\|_{\mathrm{op}} = a_d \le 2. \end{aligned}$$

The first line follows from Lemma 5 and $b_d < 1 < a_d$. For the ratio, $u_i^2, u_j^2 \le Q$ imply $|u_i u_j| \le Q$. Thus every intersection is nonempty, retains its intended exact quantity, and cannot enlarge its width. In order, these quantities are

$$a_d, \quad b_d, \quad b_d - a_d, \quad \frac{u_i u_j}{Q}, \quad (b_d - a_d)\frac{u_i u_j}{Q}, \quad (M_0)_{ij}, \quad M_{ij}, \quad v_i.$$

This proves enclosure before any error estimates are used.

3. Propagating all rounding errors. Set $w = 64(d + 1)2^{-N}$. Since $\gamma \le 2^{-N}$,

$$2\gamma \le w, \qquad 2^{2-N} + 2\gamma \le 6\,2^{-N} \le w.$$

The initial widths in Table 6 are at most 2^{2-N}, so all four scalar intervals have width at most w. We now apply the preceding width rules at every constructed interval:

$$\begin{aligned} \mathrm{w}(\mathcal{A}) &\le d\,\mathrm{w}(\mathcal{R}_D) + 2\gamma \le (d + 1)w, \\ \mathrm{w}(\mathcal{B}) &\le 2\gamma \le w, \\ \mathrm{w}(\mathcal{H}) &\le \mathrm{w}(\mathcal{B}) + \mathrm{w}(\mathcal{A}) + 2\gamma \le [1 + (d + 1) + 1]w = (d + 3)w, \\ \mathrm{w}(\mathcal{G}_{ij}) &\le |u_i u_j|\,\mathrm{w}(\mathcal{R}_{Q,1}) + 2\gamma \le 2w, \\ \mathrm{w}(\mathcal{P}_{ij}) &\le 2\,\mathrm{w}(\mathcal{G}_{ij}) + \mathrm{w}(\mathcal{H}) + 2\gamma \le [4 + (d + 3) + 1]w = (d + 8)w, \\ \mathrm{w}(\mathcal{J}_{ij}) &\le \chi_{ij}\mathrm{w}(\mathcal{A}) + \mathrm{w}(\mathcal{P}_{ij}) + 2\gamma \le [(d + 1) + (d + 8) + 1]w = (2d + 10)w, \\ \mathrm{w}(\mathcal{M}_{ij}) &\le 2\,\mathrm{w}(\mathcal{J}_{ij}) + 2\,\mathrm{w}(\mathcal{S}) + 2\gamma \le [2(2d + 10) + 2 + 1]w = (4d + 23)w, \\ \mathrm{w}(V_i) &\le \frac{|u_i|}{d+1}\mathrm{w}(\mathcal{R}_Q) + 2\gamma \le \left(\frac{1}{d+1} + 1\right) w \le 2w. \end{aligned}$$

Here the product bounds use $\mathrm{a}(\mathcal{H}) \le 2$, $\mathrm{a}(\mathcal{G}_{ij}) \le 1$, and $\mathrm{a}(\mathcal{S}), \mathrm{a}(\mathcal{J}_{ij}) \le 2$; the scalar bounds use $|u_i| \le 1$ and $|u_i u_j| \le 1$. As $4d + 23 \le 16(d+1)$ for $d \ge 2$, both output widths are bounded by

$$(4d+23)w \le 2^{10}(d+1)^2 2^{-N} \le 2^{12}(d+1)^4 2^{-N}.$$

To compare this width with ϵ_d, put $\ell = \ell_d$. The definition of the ceiling gives

$$2^{\ell-1} < d+1 \le 2^{\ell}, \qquad 2^{12} d(d+1) \le 2^{12+2\ell}, \qquad r_\theta \le 12 + 2\ell.$$

The last upper bound is an integer, so it remains an upper bound after taking the ceiling in r_θ. Therefore

$$\epsilon_d = \frac{2^{-r_\theta}}{2^{12} d^4} \ge 2^{-(12+2\ell)} 2^{-12} 2^{-4\ell} = 2^{-(24+6\ell)}.$$

On the other hand, $N = 64(\ell+1)$ gives

$$2^{12}(d+1)^4 2^{-N} \le 2^{12+4\ell-64(\ell+1)} = 2^{-52-60\ell} = 2^{-28-54\ell}\, 2^{-24-6\ell} < \epsilon_d.$$

Choose one endpoint of each enclosing interval, or zero when it belongs to that interval. Both the chosen dyadic and the exact coefficient belong to the same interval, so their difference is at most its width and hence less than ϵ_d. Entrywise bounds imply the norm bounds used in the next lemma:

$$\|\bar{v} - v\|_2^2 = \sum_i |\bar{v}_i - v_i|^2 \le d\epsilon_d^2,$$

$$\|(\bar{M} - M)y\|_2^2 \le \sum_i \left(\sum_j |\bar{M}_{ij} - M_{ij}|^2 \right) \left(\sum_j |y_j|^2 \right) \le d^2 \epsilon_d^2 \quad (\|y\|_2 = 1).$$

Taking square roots, and a supremum over unit y in the matrix case, gives $\|\bar{v} - v\|_2 \le \sqrt{d}\epsilon_d$ and $\|\bar{M} - M\|_{\mathrm{op}} \le d\epsilon_d$.

4. Word lengths and the arithmetic actually performed. We bound the integer words used for the preceding rational operations; a numerator and a denominator are separate integer words. Write $u_i = k_i 2^{-t_d}$ and $J = \sum_i k_i^2$, so that

$$|k_i| \le 2^{t_d}, \qquad 0 < J \le d 2^{2t_d} \le 2^{2t_d+\ell}, \qquad Q = J/2^{2t_d}.$$

Thus J has at most $2t_d + \ell + 1$ magnitude bits. For the other constants, $d^2 - 1 < 2^{2\ell}$, $4d(d+1)+1 < 2^{2\ell+3}$, and the denominator of ϵ_d is

$$2^{r_\theta + 12} d^4 \le 2^{24+6\ell}.$$

Thus its length is at most $25 + 6\ell$; the denominators defining ρ and θ_d are shorter still. The initial numerical constants and exact dyadic data fit in $G+1$ bits because

$$\max\{2t_d + \ell + 1, 25 + 6\ell\} = 33\ell + 33 < G = 2N + 8\ell = 136\ell + 128.$$

Before range clipping, all interval endpoint magnitudes are at most $d + 5 \le 4(d+1) \le 2^{\ell+2}$. For example, the only endpoint multiplication by a growing dimension is $d\mathcal{R}_D \subset [0, d]$; the other products have factors in the fixed ranges displayed above. A rounded endpoint $Z2^{-G}$ therefore has at most $G + \ell + 4$ signed numerator bits, and its common denominator 2^G has $G+1$ bits.

Here are explicit integer comparisons for the bisection tests. Put $c_\lambda = 4d(d+1)$ and write a midpoint as $t = T/2^G$; this is exact because all bisection endpoints lie on that mesh. The four tests are

$$c_\lambda T^2 \le (c_\lambda + 1) 2^{2G}, \quad JT^2 \le 2^{2G+2t_d},$$
$$(d^2 - 1)T^2 \le 2^{2G}, \qquad\qquad JT \le 2^{G+2t_d}.$$

They use only integer multiplication, binary shifts, and comparison. An interval product uses two mesh numerators and denominator 2^{2G}; rounding it to the mesh divides its numerator by 2^G, with directed rounding determined by the remainder. For a rational scalar such as $-u_i/(d+1)$, keep the additional positive factor $d+1$ in that divisor. Keep $u_i u_j$ as $k_i k_j / 2^{2t_d}$, so its denominator is not enlarged before multiplication by $\mathcal{R}_{Q,1}$. Let b_{test} be the largest integer-word length in a bisection

comparison and b_{round} the largest in an endpoint product or its directed rounding. The displayed formulas give

$$b_{\mathrm{test}} \le 3G + 3\ell + 12, \qquad b_{\mathrm{round}} \le 3G + 4\ell + 16.$$

Indeed, a product adds the operand bit lengths; the displayed bisection tests use two factors of at most $G + \ell + 4$ bits and at most one $G + 1$-bit factor. Endpoint products use two such numerators; clearing a mesh denominator adds at most G bits, and an extra factor $d + 1$ adds at most $\ell + 1$ bits. These estimates also allow one carry bit per addition. Including signs and boundary carries, all integer operands fit the more generous bound

$$4G + 8\ell + 32 = 552\ell + 544 < 2^{10}(\ell + 1) =: U.$$

A schoolbook multiplication may use its temporary $2U + 2$-bit accumulator, whose storage and work are counted in Lemma 12. Every word still has $O(\log(d + 1))$ bits. No square-root or division on a growing state word is used.

For the work count, a *short macro* is an addition, subtraction, comparison, shift, copy, integer product, or quotient-with-remainder on words of at most U bits. Directed rounding to the common mesh uses a quotient and its remainder when division is needed. An interval product uses four endpoint products, six comparisons for its extrema, and two directed roundings. An interval sum or difference uses two endpoint operations and two directed roundings; intersecting with a known range uses two comparisons. The constants in Table 7 allow these operations and their scalar special cases.

Table 7: Additive short-macro budget for one coefficient construction. The local bit cost is the sum of the last column multiplied by the short-macro bound $C_{\mathrm{sh}}(U)$ in Lemma 12. That lemma separately charges operand selection in the full layout.

Step	Repetitions	Total short macros
Build dimension constants, mesh, and four brackets	1	64
Square the u_i and sum to obtain Q	d squares, $d - 1$ sums	$2d - 1$
Midpoint, exact test, comparison, and endpoint replacement	$4N$	$48N$
Form and clip $\mathcal{A}, \mathcal{B}, \mathcal{H}$	1	32
Form V_i, round, and select $\bar{v}_i$	d	$16d$
Form $\mathcal{G}_{ij}, \mathcal{P}_{ij}, \mathcal{J}_{ij}, \mathcal{M}_{ij}$ and select $\bar{M}_{ij}$	d^2	$64d^2$
Ruler reset, output copying, and scratch cleanup	1	$32(d^2 + d + 1)$
Total		$127 + 48N + 50d + 96d^2$

In one bisection iteration, midpoint construction takes at most two short macros; squaring, cross-multiplication, and denominator shifts take at most four; comparison and endpoint replacement take one each; and ruler/loop bookkeeping takes at most four. This explains its 12-macro allowance. These are macro calls, not unit-cost bit actions. The cleanup row also covers the final eight endpoint roundings. Therefore

$$T^{\mathrm{local}}_{\mathrm{coefficients}} \le [127 + 48N + 50d + 96d^2] C_{\mathrm{sh}}(U).$$

□

Lemma 7 (Finite shallow update)**.** *Let $d \ge 2$, let $m \ge 1$ and $0 \le k < K_m$ be integers with the parameters of Appendix B.3, and let $c \in \mathbb{Q}^d$ and $B \in \mathbb{Q}^{d \times d}$ have finite dyadic entries. Assume that $u \in (2^{-t_a}\mathbb{Z})^d$ and $1/2 \le \|u\|_\infty \le 1$, as in Lemma 6. Suppose*

$$2^m 4^{-k} \le \sigma_{\min}(B) \le \sigma_{\max}(B) \le 2^m 4^k.$$

There is a dyadic update (c_+, B_+), using the coefficients of Lemma 6, such that

$$\{c + By : \|y\| \le 1,\ u^T y/\|u\| \le \rho\} \subseteq E(c_+, B_+), \tag{10}$$

$$\mathrm{vol}(E(c_+, B_+)) \le e^{-1/[16(d+1)]} \mathrm{vol}(E(c, B)), \tag{11}$$

$$2^m 4^{-(k+1)} \le \sigma_{\min}(B_+) \le \sigma_{\max}(B_+) \le 2^m 4^{k+1}. \tag{12}$$

Every multiplication of a growing state entry is by a short coefficient. If the stored state, intermediate accumulators, and rounded outputs fit in b-bit words, their arithmetic costs $O_d(b)$ bit actions, in addition to the dimension-only coefficient construction. In the epoch layout of Lemma 12, $b = O(b_m)$, so the complete update costs $O_d(b_m)$.

Proof. We combine the classical factor representation (Bland et al., 1981, Section 6) with controlled inflation (Grötschel et al., 1988, Sections 3.2–3.3). The proof separates parameter bounds, the stored rounding errors, containment, and volume/size control.

1. Inflation and storage mesh. By the definition of r_θ in Lemma 6,

$$2^{r_\theta - 1} < 2^{12} d(d+1) \le 2^{r_\theta}, \qquad 0 < \theta_d = 2^{-r_\theta} \le \frac{1}{2^{12} d(d+1)}.$$

Let δ_d be the smallest power of two at least $16\theta_d/d$. This power can be computed without real logarithms. If $j_d = \lfloor \log_2 d \rfloor$, locating the leading bit of d gives $2^{j_d} \le d < 2^{j_d+1}$ and

$$\delta_d = 2^{-(r_\theta + j_d - 4)}, \qquad \frac{16\theta_d}{d} \le \delta_d < \frac{32\theta_d}{d}.$$

The inequalities follow by dividing $2^{j_d} \le d < 2^{j_d+1}$ by $d2^{j_d}$ and multiplying by $16\theta_d$. Set $\alpha = 1+\delta_d$. Its bounds are

$$\delta_d < \frac{32\theta_d}{d} \le \frac{1}{128 d^2(d+1)} < \frac{1}{8}, \qquad 1 < \alpha < \frac{9}{8} < 2.$$

In particular, multiplication by α is an exact copy, binary shift, and addition.

For the storage mesh define

$$r_d = \left\lceil \log_2(2^{r_\theta + 4} d^2) \right\rceil, \qquad h = h_{m,k} = 2^{-(m+2k+r_d)}, \qquad s = 2^m 4^{-k}.$$

The ceiling gives $2^{r_d} \ge 2^{r_\theta+4} d^2$, so

$$\frac{h}{s} = \frac{2^{-m-2k-r_d}}{2^{m-2k}} = 2^{-2m-r_d} \le \frac{1}{4}\,\frac{\theta_d}{16 d^2} = \frac{\theta_d}{64 d^2}. \tag{13}$$

Here $m \ge 1$ gives $2^{-2m} \le 1/4$. The assumed singular bound also gives $\|B^{-1}\|_{\mathrm{op}} \le 1/s$.

2. The stored update and its normalized errors. Define rounding away from zero by

$$\mathcal{R}_h(t) = \mathrm{sgn}(t) h \lceil |t|/h \rceil, \qquad \mathcal{R}_h(0) = 0.$$

Because $0 \le \lceil a \rceil - a < 1$ for every real a, its error satisfies $|\mathcal{R}_h(t) - t| \le h$. Here $\bar{v}$ and $\bar{M}$ are the dyadic approximations to the ideal coefficients v and M constructed in Lemma 6. The stored update is exactly

$$c_+ = \mathcal{R}_h(c + B\bar{v}), \qquad \widehat{D} = \mathcal{R}_h(B\alpha\bar{M}), \qquad B_+ = \mathcal{R}_h(\alpha\widehat{D}). \tag{14}$$

Rounding is entrywise. Both new banks are completed using the old c, B before their selectors change. Containment will follow from the error bounds below, not from entrywise rounding by itself.

Define e_c, E_1, E_2 by

$$c_+ = c + B\bar{v} + e_c, \qquad \widehat{D} = B\alpha\bar{M} + E_1, \qquad B_+ = \alpha\widehat{D} + E_2.$$

Every error entry has absolute value at most h. Hence

$$\|e_c\|_2^2 \le dh^2, \qquad \|E_i y\|_2^2 \le \sum_{r=1}^{d} \left(\sum_{j=1}^{d} (E_i)_{rj}^2 \right) \left(\sum_{j=1}^{d} y_j^2 \right) \le d^2 h^2 \quad (\|y\|_2 = 1),$$

where the matrix inequality is Cauchy–Schwarz in each row. Thus $\|e_c\|_2 \le \sqrt{d}\, h$ and $\|E_i\|_{\mathrm{op}} \le dh$. Introduce the normalized matrix and center errors

$$D = \alpha^2(\bar{M} - M) + \alpha B^{-1} E_1 + B^{-1} E_2, \qquad f = v - \bar{v} - B^{-1} e_c.$$

Expanding the actual update shows precisely why these are the useful errors:

$$\begin{aligned} B_+ &= \alpha(B\alpha\bar{M} + E_1) + E_2 = B\alpha^2\bar{M} + \alpha E_1 + E_2 \\ &= B\big[\alpha^2 M + \alpha^2(\bar{M} - M) + \alpha B^{-1}E_1 + B^{-1}E_2\big] = B(\alpha^2 M + D). \end{aligned}$$

Using $\alpha < 2$, the coefficient bounds, and (13), we obtain

$$\begin{aligned} \|D\|_{\mathrm{op}} &\le \alpha^2\|\bar{M} - M\|_{\mathrm{op}} + (\alpha+1)\|B^{-1}\|_{\mathrm{op}}\, dh \\ &\le 4d\epsilon_d + 3d\frac{h}{s} \\ &\le \frac{\theta_d}{d}\left(\frac{1}{1024d^2} + \frac{3}{64}\right) \le \frac{49\theta_d}{1024d} < \frac{\theta_d}{16d}, \\ \|f\|_2 &\le \|v - \bar{v}\|_2 + \|B^{-1}\|_{\mathrm{op}}\|e_c\|_2 \le \sqrt{d}\left(\epsilon_d + \frac{h}{s}\right) \\ &\le \frac{\theta_d}{d}\left(\frac{1}{4096d^{5/2}} + \frac{1}{64\sqrt{d}}\right) \le \frac{65\theta_d}{4096d} < \frac{\theta_d}{32d}. \end{aligned}$$

The numerical comparisons use $d \ge 1$, $49 < 64$, and $65 < 128$.

3. Invertibility and containment. Lemma 5 gives $\|M^{-1}\|_{\mathrm{op}} \le 3/2$. Set $K = \alpha^2 I + M^{-1}D$. For every unit vector z,

$$\|Kz\|_2 \ge \alpha^2 - \|M^{-1}D\|_{\mathrm{op}} \ge \alpha^2 - \frac{3}{2}\|D\|_{\mathrm{op}}.$$

The last quantity is positive: $\alpha^2 \ge 1$, $\theta_d \le 1$, and $d \ge 2$ give

$$\alpha^2 - \frac{3}{2}\|D\|_{\mathrm{op}} \ge 1 - \frac{3\theta_d}{32d} \ge 1 - \frac{3}{64} > 0.$$

Consequently K is invertible and

$$\|K^{-1}\|_{\mathrm{op}} \le \left(\alpha^2 - \frac{3}{2}\|D\|_{\mathrm{op}}\right)^{-1}.$$

As $\alpha^2 M + D = MK$, the identity $B_+ = BMK$ also proves that B_+ is invertible before its inverse is used.

Take a point in the ideal enclosing ellipsoid, written as $x = c + Bv + BMy$ with $\|y\|_2 \le 1$. Then

$$\begin{aligned} x - c_+ &= B(v - \bar{v} + My) - e_c = B(My + f), \\ B_+^{-1}(x - c_+) &= K^{-1}M^{-1}B^{-1}B(My + f) = K^{-1}(y + M^{-1}f). \end{aligned}$$

Therefore

$$\|B_+^{-1}(x - c_+)\|_2 \le \frac{1 + (3/2)\|f\|_2}{\alpha^2 - (3/2)\|D\|_{\mathrm{op}}}.$$

The denominator is at least the numerator, because

$$\begin{aligned} \alpha^2 - 1 &= 2\delta_d + \delta_d^2 \ge 2\delta_d \ge \frac{32\theta_d}{d}, \\ \frac{3}{2}(\|D\|_{\mathrm{op}} + \|f\|_2) &\le \frac{3}{2}\left(\frac{\theta_d}{16d} + \frac{\theta_d}{32d}\right) = \frac{9\theta_d}{64d} \le \frac{32\theta_d}{d}. \end{aligned}$$

Thus $\|B_+^{-1}(x - c_+)\|_2 \le 1$, proving $E(c + Bv, BM) \subseteq E(c_+, B_+)$. By Lemma 5, every retained point $c + By$ with $\|y\| \le 1$ and $u^T y/\|u\| \le \rho$ belongs to that ideal enclosing ellipsoid. This proves (10).

4. Volume decrease after all rounding. Set $T = \alpha^{-2}M^{-1}D$. Factoring $\alpha^2 M + D = \alpha^2 M(I + T)$ gives

$$\frac{|\det B_+|}{|\det B|} = \alpha^{2d}\det M\,|\det(I + T)|.$$

All factors in the ratio are nonzero by the preceding invertibility argument. Lemma 4 gives $|\det(I + T)| \le (1 + \|T\|_{\mathrm{op}})^d$. For $x \ge 0$, $\log(1 + x) = \int_0^x (1 + t)^{-1}\, dt \le x$; hence

$$\begin{aligned}\log \frac{|\det B_+|}{|\det B|} &\le \log\det M + 2d\log(1 + \delta_d) + d\log(1 + \|T\|_{\mathrm{op}}) \\ &\le \log\det M + 2d\delta_d + d\|T\|_{\mathrm{op}} \\ &\le \log\det M + 2d\delta_d + \frac{3}{2}d\|D\|_{\mathrm{op}}.\end{aligned}$$

The last line uses $\alpha^{-2} \le 1$ and $\|M^{-1}\|_{\mathrm{op}} \le 3/2$. The complete additional log-volume cost is bounded by

$$\begin{aligned}2d\delta_d + \frac{3}{2}d\|D\|_{\mathrm{op}} &< 64\theta_d + \frac{3\theta_d}{32} < 65\theta_d \\ &\le \frac{65}{4096d(d+1)} \le \frac{65}{8192(d+1)} < \frac{1}{32(d+1)}.\end{aligned}$$

For the last inequality, $65 \cdot 32 = 2080 < 8192$. Combining with $\log\det M \le -3/[8(d+1)]$ gives

$$\log \frac{|\det B_+|}{|\det B|} \le -\frac{3}{8(d+1)} + \frac{1}{32(d+1)} = -\frac{11}{32(d+1)} \le -\frac{1}{16(d+1)}.$$

Since ellipsoid volume is v_d times the absolute determinant of its factor, exponentiation proves (11).

5. Singular values for the next cut. The earlier bounds give $\alpha < 9/8$ and

$$\|D\|_{\mathrm{op}} \le \frac{\theta_d}{16d} \le \frac{1}{65536d^2(d+1)} < \frac{1}{64}.$$

For $C = \alpha^2 M + D$, the triangle and reverse triangle inequalities therefore give

$$\begin{aligned}\sigma_{\min}(C) &\ge \alpha^2\sigma_{\min}(M) - \|D\|_{\mathrm{op}} \ge \frac{2}{3} - \frac{1}{64} = \frac{125}{192} > \frac{1}{4}, \\ \sigma_{\max}(C) &\le \alpha^2\sigma_{\max}(M) + \|D\|_{\mathrm{op}} < 2(9/8)^2 + \frac{1}{64} = \frac{163}{64} < 4.\end{aligned}$$

Finally, for every unit vector z,

$$\|BCz\|_2 \ge \sigma_{\min}(B)\|Cz\|_2 \ge \sigma_{\min}(B)\sigma_{\min}(C), \qquad \|BCz\|_2 \le \sigma_{\max}(B)\sigma_{\max}(C).$$

Taking an infimum or supremum and using $B_+ = BC$ yields

$$\sigma_{\min}(B_+) \ge \tfrac{1}{4}\, 2^m 4^{-k} = 2^m 4^{-(k+1)}, \qquad \sigma_{\max}(B_+) \le 4\, 2^m 4^k = 2^m 4^{k+1}.$$

This proves (12).

6. Implementation of the displayed update. The rounding operation itself is a bit scan. Write an intermediate entry as $t = J2^{-b_{\mathrm{f}}}$, where $J \in \mathbb{Z}$ and $b_{\mathrm{f}} \ge 0$ is its number of fractional bit positions, and put $r = m + 2k + r_d$, so $h = 2^{-r}$. If $b_{\mathrm{f}} \le r$, the entry already lies on the mesh. Otherwise let $\nu = b_{\mathrm{f}} - r$ and split its magnitude into retained and discarded bits:

$$|J| = q2^\nu + \zeta, \qquad 0 \le \zeta < 2^\nu.$$

Then the exact stored rounding is

$$\mathcal{R}_h(t) = \operatorname{sgn}(J)\big(q + \mathbf{1}_{\zeta>0}\big)2^{-r}.$$

Copy the retained prefix q, scan the discarded suffix for a one, and increment the retained magnitude if that suffix is nonzero. The increment uses one carry scan. Thus this formula implements rounding away from zero for either sign without a division on a long word.

The mesh is dyadic and each coefficient has $O(\log(d+1))$ bits. To multiply a b-bit state entry by a t-bit coefficient, scan the coefficient and add at most t shifted copies of the state entry. Each addition, copy, and final mesh rounding is a linear scan. Thus one product costs $O(tb)$ bit actions; there are $O(d^3)$ such products and $t = O(\log(d+1))$. For fixed d and $b = O(b_m)$, this is $O_d(b_m)$ (Brent &

Zimmermann, 2010, Sections 1.2–1.3). The explicit layout, addressing, and scan costs are included in Lemma 12. To make the actual loops explicit, first form the scratch entries $B^{(0)}_{i\ell} = \alpha B_{i\ell}$ by a binary shift and addition, keeping the old factor intact. Then compute

$$(c_+)_i = \mathcal{R}_h\left(c_i + \sum_{\ell=1}^{d} B_{i\ell}\bar{v}_\ell\right), \quad 1 \le i \le d,$$
$$\widehat{D}_{ij} = \mathcal{R}_h\left(\sum_{\ell=1}^{d} B^{(0)}_{i\ell}\bar{M}_{\ell j}\right), \quad 1 \le i, j \le d,$$
$$(B_+)_{ij} = \mathcal{R}_h(\alpha\widehat{D}_{ij}), \quad 1 \le i, j \le d.$$

For each output entry, clear an accumulator and visit the summation index ℓ in increasing order. The center computation has two nested loops and d^2 long-by-short products; the matrix computation has three nested loops and d^3 such products. Round only after the sum is complete. Check all output words before switching the center and factor selectors, then erase the old scratch records by charged scans. Table 13 gives the directly additive cost of these steps, including inflation, rounding, and cleanup. □

B.5 Query Bounds and Reliable Short Directions

The goal of this subsection is to prove that, in all sufficiently large epochs, the finite-precision value probes yield a reliable shallow cut whenever the queried center is not yet accurate. We first use the rounded update bounds to control the centers, factors, and query locations; these bounds do not assume that the cuts preserve a minimizer. Smoothness then bounds the function values and gradients at those query locations, allowing Lemma 8 to rule out overflow for all non-padding replies in sufficiently large epochs. With normal replies established, Lemma 9 bounds the relative finite-difference error and the direction-quantization error. Together, these estimates show that the resulting shallow cut retains a small ball around a minimizer whenever the center is still inaccurate.

Lemma 8 (A uniform query envelope)**.** *Use the epoch algorithm of Appendix B.3 with $d \ge 2$ and a valid encoded anchor a. For every epoch and every sequence of normal replies up to an abort, its queries satisfy*

$$\|x\|_2 \le R^{\mathrm{qry}}_m := 2^{A+d+m+2K_m+4}.$$

For each fixed objective satisfying Appendix B.2, all useful replies in all sufficiently large epochs are necessarily normal, uniformly over legal rounding choices.

Proof. We bound, in order, the state, the query locations, and the oracle fields. The state bounds do not require the cuts to be useful for optimization.

1. Factor and center bounds. The initial factor is $B_0 = 2^m I$. Inductively, Lemma 7 gives $\|B_k\|_{\mathrm{op}} \le 2^m 4^k$ for every update taken. The ideal translation has $\|v\| = 1/(d+1)$, so Lemma 6 implies

$$\|\bar{v}\| \le \frac{1}{d+1} + \sqrt{d}\,\epsilon_d \le \frac{2}{d+1} < 1.$$

Here $\theta_d \le 1$ and $\sqrt{d}\,\epsilon_d \le 1/(4096d^{7/2}) \le 1/(d+1)$; the last comparison follows from $d+1 \le 2d \le 4096d^{7/2}$. Also, the definition of r_d gives

$$\sqrt{d}\,h_{m,k} \le \frac{\theta_d}{16d^{3/2}}\,2^{-m}4^{-k} \le \frac{\theta_d}{32d^{3/2}} < 1.$$

Thus the actual rounded center update satisfies $\|c_{k+1} - c_k\| \le 2^m 4^k + 1$. Summing from $c_0 = a$,

$$\|c_k - a\| \le \sum_{i=0}^{k-1}(2^m 4^i + 1) = 2^m\frac{4^k - 1}{3} + k$$
$$\le \frac{1}{3}\,2^m 4^k + 2^m 4^k < 2^{m+2k+1}.$$

We used $k \le 4^k \le 2^m 4^k$ for integers $k \ge 0$; at $k = 0$ the sum is empty and the same bound holds.

2. Query locations. Since $0 < \tau_m \le 1$, each probe has displacement $\tau_m \|B_k e_i\| \le 2^{m+2k}$ from its center. The anchor's explicit binary digits give $|a_i| < 2^{A+1}$ for every coordinate. Hence

$$\|a\| \le \sqrt{d}\, 2^{A+1} \le 2^{A+d+1},$$

using $\sqrt{d} \le 2^d$. Both a center query and a displaced probe therefore satisfy

$$\begin{aligned}\|x\| &\le 2^{A+d+1} + 3 \cdot 2^{m+2k} \\ &\le 5 \cdot 2^{A+d+m+2K_m} < 2^{A+d+m+2K_m+4} = R_m^{\mathrm{qry}}.\end{aligned}$$

The middle inequality uses $k \le K_m$, $m \ge 1$, and $A + d \ge 1$. It also bounds the query whose reply first causes an abort: only earlier completed updates determine that query.

3. Field magnitudes and overflow exclusion. Let $C_F = 1 + |F(a)| + \|\nabla F(a)\| + L$ and put $r = \|x - a\|$. Lipschitz continuity and (4) yield

$$\begin{aligned}\|\nabla F(x)\| &\le \|\nabla F(a)\| + Lr \le C_F(1 + r), \\ |F(x)| &\le |F(a)| + \|\nabla F(a)\| r + \tfrac{L}{2} r^2 \le C_F(1 + r + r^2/2).\end{aligned}$$

Write $R = R_m^{\mathrm{qry}} \ge 1$. Since $\|a\| \le R$, $r \le 2R$, and consequently

$$1 + r \le 1 + 2R \le 3R^2, \qquad 1 + r + r^2/2 \le 1 + 2R + 2R^2 \le 5R^2.$$

Every scalar field is thus bounded by the convenient common bound $8C_F R^2$; each gradient coordinate is bounded by its Euclidean norm.

For $E_m = A + d + m + 2K_m + 4$, the parameter definitions give

$$\begin{aligned}K_m &= 64d(d+1)(m+1) \le 96d^2 q_m, \\ A + d + m + 4 &\le q_m, \\ 2E_m &\le (2 + 384d^2) q_m \le 400 d^2 q_m \le P_m.\end{aligned}$$

Here $d + 1 \le 3d/2$, $m + 1 \le q_m$, and $q_m - (A + d + m + 4) = 3d + \ell_d - 3 \ge 0$. The final comparison uses $P_m = 2^{16} d^4 q_m$. In particular,

$$\frac{8C_F 2^{2E_m}}{2^{8P_m - 1}} \le 16 C_F 2^{-7P_m} \longrightarrow 0.$$

Choose the epoch index large enough that $7P_m > \log_2(16C_F)$. Then every true field has magnitude less than $B_{P_m}/2$. Since $P_m \ge 1$, $\eta_{P_m} = 2^{-8P_m} < 2^{8P_m - 1} = B_{P_m}/2$, so

$$B_{P_m}/2 < B_{P_m} - \eta_{P_m}.$$

The interface therefore forbids overflow. Applying this reasoning successively to each useful query rules out a first overflow in every such epoch.

All bounds depend on the fixed objective through C_F only; none uses the particular rounding choices. □

Lemma 9 (Directional shallow cut)**.** *Let $d \ge 2$ and let the fixed objective satisfy Appendix B.2 with a positive gradient Lipschitz bound $L > 0$. Fix a minimizer x^* and recall the target ball and the shallow-cut allowance:*

$$\mathcal{T}_m = \{x : \|x - x^*\|_2 \le 2^{-m-2}\}, \qquad \rho = \frac{1}{32(d+1)}.$$

There is a finite instance-dependent m_0 with the following property. Consider any epoch $m \ge m_0$ and any cut index $0 \le k < K_m$ reached by the algorithm of Appendix B.3, with its stored state (c, B), its probes at precision P_m, and arbitrary legal replies. Suppose $\mathcal{T}_m \subseteq E(c, B)$ and $F(c) - F^* > (L/16)4^{-m}$. Then the direction u in (9) *is well defined, nonzero, and satisfies*

$$\frac{u^T y}{\|u\|} \le \rho \quad \textit{for every } c + By \in \mathcal{T}_m.$$

The conclusion holds uniformly over all legal rounding errors.

Proof. Increase the onset index, if necessary, so that Lemma 8 makes every useful reply normal. Let $g = B^T \nabla F(c)$ and let $r = (\Delta_1, \dots, \Delta_d)$ be the observed difference vector, where $\Delta_i = v_i - v_0$ and $v_i = \widetilde{F}(x_i)$ is the stored value at the algorithm's probe x_i. Set $\delta = 1/[1024(d+1)]$ as the relative-error tolerance. We compare the directions of g, r, and u in four steps.

1. The true direction is separated from zero. For $x \in \mathcal{T}_m$, the smoothness bound (5) gives

$$F(x) - F^* \le \frac{L}{2}(2^{-m-2})^2 = \frac{L}{32}4^{-m}.$$

Initialization and (12) imply $\sigma_{\min}(B) \ge 2^m 4^{-k} > 0$, so B is invertible. The containment hypothesis therefore gives the unique coordinate $y = B^{-1}(x-c)$ with $\|y\| \le 1$. First-order convexity (3) and the assumed center gap imply

$$g^T y = \langle \nabla F(c), x - c \rangle \le F(x) - F(c) \le -\frac{L}{32}4^{-m}.$$

The target ball is nonempty. Applying Cauchy–Schwarz to any of its points yields

$$\frac{L}{32}4^{-m} \le -g^T y \le \|g\|\,\|y\| \le \|g\|.$$

Thus $g \neq 0$, and the relative errors below have nonzero denominators.

2. Value errors give a small relative vector error. Write $v_i = F(x_i) + \xi_i$ for $0 \le i \le d$, where normal replies satisfy $|\xi_i| \le \eta_{P_m} = 2^{-8P_m}$ by Appendix A.4. For $1 \le i \le d$, the ith component of g is $g_i = \langle \nabla F(c), Be_i \rangle$. With $x_i = c + \tau_m B e_i$ and $x_0 = c$,

$$\Delta_i - \tau_m g_i = F(c + \tau_m B e_i) - F(c) - \langle \nabla F(c), \tau_m B e_i \rangle + \xi_i - \xi_0.$$

Applying (4) and the triangle inequality gives

$$|\Delta_i - \tau_m g_i| \le \frac{L}{2}\tau_m^2 \|Be_i\|^2 + 2\eta_{P_m}. \tag{15}$$

Initialization at $B_0 = 2^m I$ and the singular-value bound (12) in Lemma 7 give

$$\|Be_i\| \le \|B\|_{\mathrm{op}} \le 2^m 4^k \le 2^m 4^{K_m}.$$

Using $\|w\|_2 \le \sqrt{d}\,\|w\|_\infty$ for a vector w, the coordinate bounds therefore combine as

$$\|r - \tau_m g\| \le \sqrt{d}\left[\frac{L}{2}\tau_m^2\, 2^{2m+4K_m} + 2\eta_{P_m}\right].$$

Divide by $\tau_m\|g\| \ge \tau_m (L/32) 4^{-m}$:

$$\begin{aligned}\frac{\|r - \tau_m g\|}{\tau_m \|g\|} &\le 16\sqrt{d}\,\tau_m 2^{4m+4K_m} + 64\sqrt{d}\,L^{-1}\eta_{P_m}\tau_m^{-1}2^{2m} \\ &= 16\sqrt{d}\,2^{-b_m+4m+4K_m} + 64\sqrt{d}\,L^{-1}2^{-8P_m+b_m+2m}.\end{aligned}$$

For the first term, $m \le q_m$ and $K_m \le 96d^2 q_m$ imply

$$4m + 4K_m \le (4 + 384d^2)q_m \le 400d^2 q_m \le b_m/2.$$

The last inequality uses $b_m/2 = 2048 d^3 q_m$. Moreover, $\ell_d \le d$ for $d \ge 2$, so

$$b_m/2 \ge 16d \ge 15 + \ell_d + \tfrac{1}{2}\log_2 d \ge 15 + \log_2(d+1) + \tfrac{1}{2}\log_2 d.$$

Here $2^d \ge d+1$ implies $\ell_d \le d$, and $\log_2 d \le d$ makes the middle comparison immediate from $16d \ge 15 + 3d/2$. It follows that

$$16\sqrt{d}\,2^{-b_m/2} \le \frac{1}{2048(d+1)}.$$

For the second term, $b_m \ge m$ and $P_m = 16 d b_m \ge 3 b_m$ give $P_m \ge b_m + 2m$. Hence $8P_m - b_m - 2m \ge 7P_m$. Because $L > 0$ is fixed and $P_m \to \infty$, we may also require

$$7P_m \ge 17 + \log_2(d+1) + \tfrac{1}{2}\log_2 d - \log_2 L.$$

This yields $64\sqrt{d}\,L^{-1}2^{-7P_m} \le 1/[2048(d+1)]$. This threshold and the normal-reply threshold from Lemma 8 are independent of k and the rounding history. Since P_m increases without bound, one finite m_0 enforces both thresholds in every subsequent epoch. Adding the two error terms proves

$$\|r - \tau_m g\| \le \delta\,\tau_m \|g\|. \tag{16}$$

In particular, $\|r\| \ge (1-\delta)\tau_m\|g\| > 0$. Since $\tau_m > 0$, the vectors $\tau_m g$ and g have the same unit direction. The normalization bound (6), with $v = r$ and $w = \tau_m g$, gives

$$\left\| \frac{r}{\|r\|} - \frac{g}{\|g\|} \right\| \le 2\delta.$$

3. Truncation adds at most δ to the directional error. Since $r \neq 0$, its largest absolute component $D = \max_i |\Delta_i| = \|r\|_\infty$ is positive. Choose the integer rescaling exponent h with $2^{h-1} \le D < 2^h$, as in (9), and put $\widehat{r} = 2^{-h} r$. Then

$$\frac{1}{2} \le \|\widehat{r}\|_\infty = 2^{-h} D < 1.$$

The truncation formula and $0 \le s - \lfloor s \rfloor < 1$ imply, for each coordinate,

$$|u_i - \widehat{r}_i| = 2^{-t_d}\left(2^{t_d}|\widehat{r}_i| - \left\lfloor 2^{t_d}|\widehat{r}_i| \right\rfloor\right) < 2^{-t_d}.$$

Summing the squared coordinate errors gives

$$\|u - \widehat{r}\|^2 = \sum_{i=1}^{d} |u_i - \widehat{r}_i|^2 \le d\,2^{-2t_d}, \qquad \|u - \widehat{r}\| \le \sqrt{d}\,2^{-t_d}.$$

For a largest-magnitude coordinate i_*, $2^{t_d}|\widehat{r}_{i_*}| \ge 2^{t_d-1}$. Since the right side is an integer, its floor is still at least 2^{t_d-1}; thus $|u_{i_*}| \ge 1/2$ and $u \neq 0$. Using (6) again,

$$\begin{aligned} \left\| \frac{u}{\|u\|} - \frac{r}{\|r\|} \right\| &= \left\| \frac{u}{\|u\|} - \frac{\widehat{r}}{\|\widehat{r}\|} \right\| \\ &\le \frac{2\|u - \widehat{r}\|}{\|\widehat{r}\|} \le 4\sqrt{d}\,2^{-t_d}. \end{aligned}$$

The parameter choice $t_d = 16(\ell_d + 1)$ makes this at most δ:

$$4\sqrt{d}\,2^{-t_d} \le 2^{-14-(31/2)\ell_d} \le 2^{-10-\ell_d} \le \frac{1}{1024(d+1)}.$$

The first and last comparisons use $d \le 2^{\ell_d}$ and $d + 1 \le 2^{\ell_d}$, respectively; the middle comparison uses $\ell_d \ge 0$.

4. The stored normal retains the target ball. By the triangle inequality,

$$\left\| \frac{u}{\|u\|} - \frac{g}{\|g\|} \right\| \le \delta + 2\delta = 3\delta < \frac{1}{32(d+1)} = \rho.$$

For any $x = c + By \in \mathcal{T}_m$, step 1 gives $g^T y < 0$ and $\|y\| \le 1$. Therefore

$$\begin{aligned} \frac{u^T y}{\|u\|} &= \frac{g^T y}{\|g\|} + \left\langle \frac{u}{\|u\|} - \frac{g}{\|g\|}, y \right\rangle \\ &\le \frac{g^T y}{\|g\|} + \left\| \frac{u}{\|u\|} - \frac{g}{\|g\|} \right\| \|y\| \le 3\delta < \rho. \end{aligned}$$

The onset conditions involve only the fixed instance and the prescribed parameters, so the conclusion is uniform over legal rounding. This is the approximate-direction shallow-cut principle of Karabag et al. (2021, Lemma 1), with the finite-difference and truncation errors included explicitly. □

B.6 Accuracy of a Completed Epoch

Lemma 10 (Completed-epoch accuracy)**.** *Let $d \geq 2$, assume the objective conditions of Appendix B.2, and run the epoch algorithm of Appendix B.3 from the encoded anchor a. There are a finite instance-dependent epoch index and a constant $C_F > 0$ such that every later epoch finishes without overflow and publishes a candidate $z^{(m)}$ satisfying*

$$F(z^{(m)}) - F^* \leq \frac{L}{16} 4^{-m} + 2\eta_{P_m} \leq C_F 4^{-m}.$$

Here $z^{(m)}$ is the report published at the end of epoch m, and $\eta_{P_m} = 2^{-8P_m}$ is the value-error allowance of a normal reply. The index and constant work for all legal rounding transcripts.

Proof. We first obtain a good queried center, then account for selection by noisy values. Fix a minimizer x^*. In the current epoch, let c_k, B_k be the stored center and factor after k completed updates, and write

$$E_k = E(c_k, B_k), \qquad \mathcal{T}_m = \{x : \|x - x^*\|_2 \leq 2^{-m-2}\}.$$

1. Choose an onset and initialize the containment invariant. Lemma 8 supplies an index beyond which no useful reply overflows. The epoch attempts at most K_m iterations, each with $d+1$ finite replies and the finite arithmetic constructions of Lemmas 6 and 7. Publication copies a finite record. Thus every such epoch finishes either after K_m updates or on the zero-difference branch. Both branches are reached after consuming a normal center reply, so the candidate bank is nonempty before publication.

If $L = 0$, stationarity at a minimizer from Lemma 4 and the Lipschitz condition give

$$\nabla F(x^*) = 0, \qquad \|\nabla F(x) - \nabla F(x^*)\| \leq 0,$$

so the gradient vanishes everywhere. Hence

$$F(x) - F^* = \int_0^1 \langle \nabla F(x^* + t(x - x^*)), x - x^* \rangle \, dt = 0.$$

Every published candidate is then exact. Since $P_m \geq m$, the claimed chain holds with $C_F = 2$:

$$0 = F(z^{(m)}) - F^* \leq 2\eta_{P_m} = 2\,2^{-8P_m} \leq 2\,4^{-m}.$$

Assume henceforth that $L > 0$.

Increase the index to satisfy Lemma 9 and $2^m \geq \|a - x^*\| + 1$. For $x \in \mathcal{T}_m$,

$$\|x - a\| \leq \|x - x^*\| + \|x^* - a\| \leq 2^{-m-2} + \|x^* - a\| \leq 2^m,$$

so $\mathcal{T}_m \subseteq E_0 = E(a, 2^m I)$. The last inequality uses $2^{-m-2} \leq 1$ and the chosen lower bound on 2^m. For every reached state with k completed updates, the induction invariant is

$$\big[F(c_i) - F^* > (L/16)4^{-m} \text{ for all } i < k\big] \implies \mathcal{T}_m \subseteq E_k.$$

It holds at $k = 0$. If it holds at k and the current center is also inaccurate, Lemma 9 supplies a nonzero normal whose shallow half contains $\mathcal{T}_m$. The difference vector is nonzero by (16), so the zero-difference branch does not stop the iteration. Lemma 7 then encloses that half in E_{k+1}, proving the next induction step.

2. A completed run of all cuts contains a good center. Consider an epoch that completes all K_m updates, and suppose its queried centers $c_0, \ldots, c_{K_m - 1}$ were all inaccurate. The invariant would give $\mathcal{T}_m \subseteq E_{K_m}$. Writing v_d for the unit-ball volume, the initial-to-target ratio is

$$\frac{v_d (2^m)^d}{v_d (2^{-m-2})^d} = 2^{2d(m+1)}.$$

The volume bound (11) in Lemma 7 holds at each update. Iterating it gives

$$\operatorname{vol}(E_{K_m}) \leq \left(\prod_{k=0}^{K_m - 1} e^{-1/[16(d+1)]} \right) \operatorname{vol}(E_0) = e^{-K_m/[16(d+1)]} \operatorname{vol}(E_0).$$

Divide by the target volume and substitute $K_m = 64d(d+1)(m+1)$:

$$\begin{aligned}\frac{\mathrm{vol}(E_{K_m})}{\mathrm{vol}(\mathcal{T}_m)} &\le e^{-K_m/[16(d+1)]}\, 2^{2d(m+1)} \\ &= \exp\left(2d(m+1)\log 2 - \frac{64d(d+1)(m+1)}{16(d+1)}\right) \\ &= \exp[(2\log 2 - 4)d(m+1)] < 1.\end{aligned}$$

The last exponent is negative because $\log 2 < 1$. Containment of a ball of positive volume requires the ratio to be at least one, a contradiction. Thus at least one queried center has gap at most $(L/16)4^{-m}$. This is the ellipsoid volume argument (Bubeck, 2015, Section 2.2).

3. Early termination also contains a good center. The only other termination after the chosen onset is $D = \max_i |\Delta_i| = 0$, equivalently $r = (\Delta_1, \ldots, \Delta_d) = 0$, where $\Delta_i = v_i - v_0$ are the observed probe differences at the stopping state (c_k, B_k). If an earlier center was accurate, the desired point has already been queried. Otherwise the invariant gives $\mathcal{T}_m \subseteq E_k$ at the stopping center. If that center were inaccurate, the reverse triangle inequality, the relative-error estimate (16), and the true-direction lower bound in step 1 of Lemma 9 would give

$$\begin{aligned}\|r\| &\ge \left(1 - \frac{1}{1024(d+1)}\right)\tau_m \|B_k^T \nabla F(c_k)\| \\ &\ge \left(1 - \frac{1}{1024(d+1)}\right)\tau_m \frac{L}{32} 4^{-m} > 0,\end{aligned}$$

contradicting $r = 0$. The stopping center is therefore accurate.

4. Observed-value selection preserves the accuracy bound. Let x_{good} be a good center from step 2 or 3. Let $\widetilde{v}_{\mathrm{good}}$ denote the value stored when it was queried, and let $(w, \widetilde{v}_w)$ be the selected candidate record. The algorithm retains the smallest observed value among all epoch queries, so $\widetilde{v}_w \le \widetilde{v}_{\mathrm{good}}$. Each record has error at most η_{P_m}, even when repeated queries at the same point receive different legal roundings. Hence

$$\begin{aligned}F(w) - F^* &\le \widetilde{v}_w + \eta_{P_m} - F^* \\ &\le \widetilde{v}_{\mathrm{good}} + \eta_{P_m} - F^* \\ &\le F(x_{\mathrm{good}}) - F^* + 2\eta_{P_m} \\ &\le (L/16)4^{-m} + 2\eta_{P_m}.\end{aligned}$$

Publication copies the candidate exactly, giving $z^{(m)} = w$. Finally $P_m \ge m$ implies $2^{-8P_m} \le 2^{-8m} \le 2^{-2m} = 4^{-m}$, so $C_F = L/16 + 2$ is a valid constant in the stated bound.

All onset conditions precede the choice of rounding transcript. □

B.7 The One-Dimensional Branch

The factor formula above contains $d^2 - 1$ in a denominator and is used only for $d \ge 2$. In one dimension an ellipsoid is simply an interval. This subsection gives the scalar algorithm, proves its completed-epoch accuracy, and then accounts for its exact rational implementation. We use a bounded-error version of classical interval elimination (Brent, 2013, Chapter 5), specified in Algorithm 3.

One *ternary decision* queries the two trisection points $x = l + (r-l)/3$ and $y = l + 2(r-l)/3$ of the current interval $[l, r]$. If the observed value at x is no larger, retain $[l, y]$; otherwise retain $[x, r]$. Thus either decision discards one outer third and retains two thirds of the interval. Epoch m starts from $[a - 2^m, a + 2^m]$ and attempts $8m$ such decisions at precision $p = m$. A completed epoch therefore makes $16m$ useful queries. A decision may require many physical rounds: queries are sequential, every reply is fully consumed, and padding calls allow the pending local work to continue under the per-round cap.

Because finite-precision comparisons may discard a minimizer, the algorithm also retains the query with the least observed value throughout the epoch. It publishes that candidate after all decisions

finish. An overflow instead aborts the epoch while preserving the previous report. The lemma below proves that either the final interval still contains a minimizer or a query made before losing it is already accurate.

Algorithm 3 One-dimensional localization by noisy ternary decisions

State: Public report z; interval $[l, r]$; candidate w and its observed value v.
Init: Set $z \leftarrow a$ and parse the input as in Lemma 3.
for each epoch $m = 1, 2, 4, \ldots$ **do**
  Phase I: allocate and reset. $O((A+m)^2)$ rounds
  Set $p \leftarrow m$, $(l, r) \leftarrow (a - 2^m, a + 2^m)$, and mark w empty.
  Phase II: interval elimination.
  for $k = 0, \ldots, 8m - 1$ **do** $O(A+m)$ rounds per decision
    Set $x \leftarrow l + (r-l)/3$ and $y \leftarrow l + 2(r-l)/3$.
    for $q = x$, then $q = y$ **do**
      Query (q, p) and fully consume its reply before any next useful query.
      On overflow, retain z, clean up, and start the next epoch.
      Store $v_q \leftarrow \widetilde{F}(q)$.
      If w is empty or $v_q < v$, set $(w, v) \leftarrow (q, v_q)$; ties keep w.
    end for
    if $v_x \le v_y$ **then**
      Set $r \leftarrow y$; retain $[l, y]$.
    else
      Set $l \leftarrow x$; retain $[x, r]$.
    end if
  end for
  Phase III: publish and clean up. $O(A+m)$ rounds
  Fully copy w into an inactive report bank, set $z \leftarrow w$, and erase obsolete state.
end for
Epoch total: $O((A+m)^2)$ rounds; detailed budgets are proved in Lemma 11.

Lemma 11 (Noisy ternary epochs)**.** *For $d = 1$, let F satisfy the objective assumptions in Appendix B.2, and let a be the encoded dyadic input anchor. For every integer $m \ge 1$, the epoch routine of Algorithm 3 starts from $[a - 2^m, a + 2^m]$, uses oracle precision $p = m$, and attempts $8m$ ternary decisions. Abort on a useful overflow and keep the previous report. Otherwise publish the query with least observed value, breaking ties by earlier storage order. After a finite instance-dependent onset, all $8m$ decisions finish and*

$$F(z^{(m)}) - F^* \le (4 + L/2)4^{-m}.$$

The complete epoch has an implementation with at most 2^{12} charged bit actions per round, legal live storage, and

$$H_m^{(1)} < 2^{32}(A + m + 6)^2.$$

Here $z^{(m)}$ is the report published by a successful epoch, and $H_m^{(1)}$ counts all physical rounds of the scalar epoch, including padding, publication, or abort cleanup. The resource bounds apply also to aborted epochs; the accuracy guarantee and its onset are uniform over all legal rounding transcripts.

Proof. We first bound the interval width in Algorithm 3, prove accuracy in both possible cases, and then bound the exact rational implementation and its charged costs. Throughout this proof, the precision is $p = m$, so normal value observations obey

$$|\widetilde{F}(q) - F(q)| \le \eta_m = 2^{-8m}, \qquad B_m = 2^{8m},$$

where B_m is the allowed magnitude range from Appendix A.4. Each $\widetilde{F}(q)$ denotes the particular stored observation from that query; repeated queries at the same point may have different observations. The symbol w denotes the final epoch candidate, while w_m below denotes an interval width.

1. Decisions and interval width. For a current interval $[l, r]$ of width $W = r - l > 0$, either branch of the decision in Algorithm 3 retains width $2W/3$. Let w_m be the width after all $8m$ decisions have

completed. Starting from width 2^{m+1}, we obtain

$$\begin{aligned} w_m &= 2^{m+1}(2/3)^{8m} = 2^{1-m}\left[4(2/3)^8\right]^m \\ &= 2^{1-m}(1024/6561)^m \le 2^{1-m}. \end{aligned}$$

Here $4(2/3)^8 = 1024/6561 < 1$ and $m \ge 1$. In the following accuracy argument, suppose the initial interval contains a minimizer and all replies are normal.

2. The first decision that loses all minimizers. If such a decision occurs, the pre-decision interval contains a minimizer t_*. There are two cases.

- If $[l, y]$ is retained, every minimizer in $[l, r]$ lies to the right of y, so choose $t_* \in (y, r]$. Write

  $$y = \frac{t_* - y}{t_* - x}x + \frac{y - x}{t_* - x}t_*.$$

  The two coefficients are nonnegative and sum to one. Convexity gives

  $$(t_* - x)F(y) \le (t_* - y)F(x) + (y - x)F^*,$$

  and rearranging gives

  $$(y - x)(F(y) - F^*) \le (t_* - y)(F(x) - F(y)).$$

  The left side is nonnegative, so $F(x) - F(y) \ge 0$. Since $0 < t_* - y \le r - y = y - x$, division by $y - x > 0$ yields

  $$\begin{aligned} F(y) - F^* &\le \frac{t_* - y}{y - x}[F(x) - F(y)] \le F(x) - F(y) \\ &\le \widetilde{F}(x) - \widetilde{F}(y) + 2\eta_m \le 2\eta_m. \end{aligned}$$

  The last inequality is precisely the decision rule.

- If $[x, r]$ is retained, choose $t_* \in [l, x)$. Now

  $$x = \frac{y - x}{y - t_*}t_* + \frac{x - t_*}{y - t_*}y.$$

  Applying convexity and rearranging gives

  $$(y - x)(F(x) - F^*) \le (x - t_*)(F(y) - F(x)).$$

  Again nonnegativity of the left side gives $F(y) - F(x) \ge 0$. Use $0 < x - t_* \le x - l = y - x$ and $\widetilde{F}(y) < \widetilde{F}(x)$ to conclude

  $$\begin{aligned} F(x) - F^* &\le \frac{x - t_*}{y - x}[F(y) - F(x)] \le F(y) - F(x) \\ &\le \widetilde{F}(y) - \widetilde{F}(x) + 2\eta_m \le 2\eta_m. \end{aligned}$$

Thus a queried point $q \in \{x, y\}$ has gap at most $2\eta_m$. For the final candidate w, its selected observation is no larger than the observation at q, regardless of subsequent interval decisions. The two observation errors give

$$F(w) \le \widetilde{F}(w) + \eta_m \le \widetilde{F}(q) + \eta_m \le F(q) + 2\eta_m, \qquad F(w) - F^* \le 4\eta_m.$$

3. No decision loses all minimizers. Choose a minimizer t_* in the final interval. The interval just before the last decision has width $3w_m/2$. Write this preceding interval as $[l, r]$, so $r = l + 3w_m/2$. Its probes are $x = l + w_m/2$ and $y = l + w_m$, and

$$[l, r] \subseteq [x - w_m/2, x + w_m/2] \cup [y - w_m/2, y + w_m/2].$$

One of these probes, say q, therefore satisfies $|q - t_*| \le w_m/2$. By (5),

$$F(q) - F^* \le \tfrac{L}{2}(w_m/2)^2 = Lw_m^2/8.$$

Candidate selection adds at most $2\eta_m$ as above. The two exhaustive cases are consequently bounded by

$$\begin{aligned} F(z^{(m)}) - F^* &\le \max\{4\eta_m,\ Lw_m^2/8 + 2\eta_m\} \\ &\le 4\eta_m + Lw_m^2/8 \\ &\le 4\,2^{-8m} + \tfrac{L}{8}\,2^{2-2m} \le (4 + L/2)4^{-m}. \end{aligned}$$

4. Both onset conditions eventually hold. The initial interval contains a fixed minimizer once $2^m \ge |a - t_*|$. Every later query remains in this interval, so $|x - a| \le 2^m$. Set $C_F = 1 + |F(a)| + |F'(a)| + L$. The smoothness estimate (4) and the Lipschitz condition on F' give

$$\begin{aligned} |F(x)| &\le |F(a)| + |F'(a)|2^m + \tfrac{L}{2}\,2^{2m} \le \tfrac{5}{2}C_F 2^{2m}, \\ |F'(x)| &\le |F'(a)| + L2^m \le 2C_F 2^m \le 2C_F 2^{2m}. \end{aligned}$$

Both fields are bounded by $3C_F 2^{2m}$. For $6m > \log_2(6C_F)$ this is less than $2^{8m-1} = B_m/2$, which is strictly below $B_m - \eta_m$ because $m \ge 1$. The overflow rule in Appendix A.4 therefore forbids overflow in every sufficiently large epoch. Both onset conditions depend only on the fixed instance, not on the observed rounding errors. Each query record, reply scan, and integer update is finite, as detailed below; hence all $8m$ decisions finish after this onset. Their first normal reply fills the candidate bank, and publication copies its final record exactly. Any earlier overflow is fully consumed, aborts that epoch, and preserves the preceding public report.

5. Exact endpoint updates and record lengths. Write the dyadic anchor as $a = N_a/2^{b_a}$ using its input digits, where $N_a \in \mathbb{Z}$ and $b_a \in \mathbb{N}_0$ is the number of fractional binary digits in the input. The initial integer records are

$$L_0 = N_a - 2^{m+b_a}, \qquad R_0 = N_a + 2^{m+b_a}, \qquad D_0 = 2^{b_a}.$$

At decision k, the endpoints are L_k/D_k and R_k/D_k. The two trisection queries and subsequent state are

$$x = \frac{2L_k + R_k}{3D_k}, \qquad y = \frac{L_k + 2R_k}{3D_k},$$

$$(L_{k+1}, R_{k+1}, D_{k+1}) = \begin{cases} (3L_k, L_k + 2R_k, 3D_k), & \widetilde{F}(x) \le \widetilde{F}(y), \\ (2L_k + R_k, 3R_k, 3D_k), & \widetilde{F}(x) > \widetilde{F}(y). \end{cases}$$

Thus $D_k = 2^{b_a}3^k$ by induction. Since every endpoint remains in $[a - 2^m, a + 2^m]$,

$$\max\{|L_k|, |R_k|\} \le D_k(|a| + 2^m) = 3^k(|N_a| + 2^{m+b_a}).$$

Only integer additions and multiplication by two or three are needed. At decision k, both queries have denominator $D_{k+1} = 3D_k$ and remain in the initial interval. Thus the following record-length bounds also apply to their numerators, with index $k + 1 \le 8m$, before the next endpoint state is committed.

Set

$$X = A + m + 6, \qquad \Lambda_m = 64X.$$

The quantity Λ_m bounds the data length of one scalar record; L_k continues to denote a left-endpoint numerator. Since $3^5 < 2^8$, for $0 \le k \le 8m$, $3^k \le 2^{8k/5} \le 2^{64m/5} < 2^{13m}$. The input gives $|N_a| < 2^A$ and $b_a \le A$, so

$$\begin{aligned} D_k &= 2^{b_a}3^k < 2^{A+13m}, \\ \max(|L_k|, |R_k|) &\le 3^k(|N_a| + 2^{m+b_a}) \\ &< 2^{13m}(2^A + 2^{m+A}) \le 2^{A+14m+1}. \end{aligned}$$

Here L_k, R_k are the current endpoint numerators. Thus they and the common denominator need at most $A + 14m + 2$ and $A + 13m + 1$ bits, respectively. A complete one-coordinate vector record in the numerator-denominator format of Appendix A.1, with numerator and denominator magnitude lengths ℓ_N, r_D, has length $2\ell_N + 2r_D + 7$: the integer encodings contribute $2\ell_N + 2r_D + 3$ and the two vector markers contribute four bits. It therefore needs at most

$$2(A + 14m + 2) + 2(A + 13m + 1) + 7 = 4A + 54m + 13 < \Lambda_m.$$

The full query adds $m + 5$ bits for query/precision framing as in Appendix A.3, so

$$|Q| \leq 4A + 55m + 18 < 64(A + m + 6) = \Lambda_m.$$

The complete reply length (2) at $d = 1$ is $R(m) = 2[2 + 2(16m + 4)] = 64m + 20 \leq \Lambda_m$.

6. Simultaneous storage and charged work.

Use the twenty data regions in Table 8. Each region has data, a left-boundary flag, and a head-position track; its three heads move together. The region length is at most $\Lambda_m + 1$, including a boundary cell. The common denominator is stored once; the two next-numerator regions first hold the two trisection numerators. After the decision, overwrite the discarded numerator by three times the retained old endpoint numerator. The new denominator $3D_k$ is held in the third arithmetic scratch region until this replacement is complete. The other two arithmetic regions supply reusable temporaries; each signed sum is written into its next-numerator region before the temporaries are reused. Replies are consumed sequentially, so the same useful buffer holds each reply in turn, while the two observed values have separate regions.

Table 8: Simultaneous data regions for the scalar branch. Every row is multiplied by three tracks; regions are reused only after charged cleanup.

Stored objects	Number of regions
Current/next endpoint numerators and common denominator	5
Two complete query records	2
Epoch candidate and old/new public reports	3
Two observed values and candidate value	3
Useful and padding buffers	2
Arithmetic accumulators, including the next denominator	3
Precision and iteration rulers	2
Total	20

The immutable input, converted anchor, and retained parser regions cost at most $16A + 256$ further cells by Lemma 3 at $d = 1$. Writing S_{live} for the total live storage in bits, we obtain

$$S_{\text{live}} \leq 60(\Lambda_m + 1) + 16A + 256 \leq 64\Lambda_m = 2^{12}(A + m + 6) \leq 2^{20}(A + m + 1).$$

For the middle comparison,

$$64\Lambda_m - [60(\Lambda_m + 1) + 16A + 256] = 240A + 256m + 1220 > 0.$$

For the budget comparison, $A + m + 6 \leq 256(A + m + 1)$, since the difference of the right and left sides is $255(A + m) + 250 > 0$. At an epoch change, erase scratch before reusing and extending its regions; keep the old public report in its listed region until publication. That report comes from an earlier epoch $m' \leq m$, or is the initial anchor. The preceding record-length bound is increasing in m, so it also bounds this retained report.

For clarity, a scalar scan macro operates on at most three selected data regions. During incremental query emission, the output head may stay at its marked append position; other head groups return to their boundaries. The bound below allows all three head groups to return. A group move costs two head-mark writes and three head moves, or 5 actions. A forward position uses at most three such moves, three reads, and three writes, or 21 actions. A backward position uses at most three moves and three boundary reads, or 18. Including at most 16 entry and exit actions, a macro costs

$$21\Lambda_m + 18\Lambda_m + 16 \leq 64\Lambda_m.$$

The finitely many scalar regions are selected by the finite controller; there is no variable-length random address. Copying, comparison, addition with a carry, and shifting by one bit use these scans. A signed addition can compare magnitudes before its arithmetic scan and uses at most three macros. Multiplication by three uses a shift and such an addition, hence at most four macros.

Table 9: Additive scalar-epoch work budget in scan macros. Multiply the last column by $64\Lambda_m$ to obtain bit actions; allocation is charged separately below. Decision rows are executed at most $8m$ times.

Step	Repetitions	Total macros
Initialize endpoint words, rulers, and headers	1	64
Form $2L_k + R_k$, $L_k + 2R_k$, and $3D_k$	$8m$	$12(8m)$
Emit two query records and their unary precision fields	$8m$	$24(8m)$
Consume both complete replies and retain their values	$8m$	$8(8m)$
Compare the two values, update the candidate twice if needed	$8m$	$14(8m)$
Write retained endpoints and commit the common denominator	$8m$	$24(8m)$
Reset scratch/rulers and advance the decision counter	$8m$	$16(8m)$
Publish or abort, erase scratch, and reset the layout	1	128
Total		$192 + 98(8m)$

The arithmetic row uses four macros for each of its three scaled sums. For each query, its 12-macro allowance includes scanning the numerator and denominator lengths, emitting their unary headers and payloads, writing vector and precision markers, and rewinding. Each receipt uses at most four passes for complete scanning, value copying, erasure, and head reset. The comparison row allows two macros for each of three comparisons and four for each of two possible candidate point/value replacements, giving $3 \cdot 2 + 2 \cdot 4 = 14$. The endpoint row allows two 12-macro paths to write and reset the selected numerators, denominator, and validity markers. All regions used for these steps are in Table 8.

A conservative allocator writes one layout cell after a full traversal; for at most $64\Lambda_m$ cells its cost is at most $16(64\Lambda_m)^2$. Let $T^{(1)}_{\mathrm{main}}$ be the charged work of the scalar epoch excluding padding-reply handling and useful-delivery control. Summing allocation with Table 9 gives

$$\begin{aligned} T^{(1)}_{\mathrm{main}} &\le 16(64\Lambda_m)^2 + [192 + 98(8m)]64\Lambda_m \\ &= 2^{28}X^2 + 3{,}211{,}264\, mX + 786{,}432\, X \\ &< 2^{28}X^2 + 2^{22}mX + 2^{20}X \\ &\le (2^{28} + 2^{22} + 2^{20})X^2 < 2^{29}X^2. \end{aligned}$$

Here $m \le X$ and $X \ge 1$, and the decision coefficients sum to $12 + 24 + 8 + 14 + 24 + 16 = 98$. After each padding reply, execute one suspended main primitive. The scan, erasure, and rewind of a padding buffer, followed by that primitive, take at most

$$14R(0) + 2 = 14 \cdot 20 + 2 = 282 < 2^{12}$$

charged actions. A useful-delivery round restores the padding selector; the receipt then advances in subsequent padding rounds. No new useful call is issued before the previous receipt is complete. There are at most $16m$ useful calls, so the physical rounds satisfy

$$H^{(1)}_m \le T^{(1)}_{\mathrm{main}} + 16m + 2 < 2^{32}(A + m + 6)^2.$$

Indeed, $16m + 2 \le 18X \le 18X^2$, and $2^{29} + 18 < 2^{32}$. Keep the old report selected until its replacement is fully copied.

□

B.8 Bit Complexity and Storage Bounds

The goal of this subsection is to show that the localization algorithm fits the prescribed storage and per-round work budgets, and that a complete epoch takes $O_d((A + m)^2)$ physical rounds. Let H_m denote the total number of physical rounds used by epoch m, from allocation and initialization through final cleanup, whether the epoch publishes a new report or aborts. Both useful-call and padding-call rounds are counted. Lemma 12 proves this by the following chain of bounds. Recall $q_m = A + 4d + m + \lceil \log_2(d + 1) \rceil + 1 = \Theta_d(A + m)$.

1. The query envelope bounds the integer parts of stored numbers, and the rounding mesh bounds their fractional parts. Together with $b_m, P_m = O_d(q_m)$, these estimates bound the lengths of state words, query records, and complete replies by $O_d(q_m)$.
2. Sum all simultaneously resident regions, including old and new matrix banks, reply buffers, reports, scratch, and control tracks. Their total length $\mathcal{L}_m$ is $O_d(q_m)$ and satisfies the explicit storage budget. In particular, $P_m = 16db_m$ gives $d^2b_m = dP_m/16$, allowing the matrix banks to fit within that budget.
3. Bound a complete cut, including query emission, full receipt, candidate selection, and the rounded update. The update coefficients have only $O(\log(d+1))$ bits, so applying them to long state entries uses linear scans for fixed d. Counting addressing and head moves as well as arithmetic gives $O_d(\mathcal{L}_m) = O_d(q_m)$ bit actions per cut.
4. Let T_{main} count the epoch's charged main-program bit actions, including complete useful-reply scans but excluding padding handling and the separate useful-delivery and boundary-control actions. Allocation and initialization cost $O_d(q_m^2)$, the at most $K_m = O_d(m)$ cuts cost $K_m O_d(q_m)$, and publication and cleanup cost $O_d(q_m)$. Since $m \le q_m$,
$$T_{\mathrm{main}} = O_d(q_m^2) + K_m O_d(q_m) + O_d(q_m) = O_d(q_m^2).$$
5. Schedule one pending main-program bit action after processing each padding reply. Such a round uses at most $112d + 170 < 2^{12}d$ charged actions. Useful-delivery and boundary-control rounds add at most $(d+1)K_m + 2$, so
$$H_m \le T_{\mathrm{main}} + (d+1)K_m + 2 = O_d((A+m)^2).$$

The proof below makes each bound explicit through additive accounting tables, including reply fields unused by the geometric algorithm. The numerical constants are conservative and verify the fixed budgets.

Lemma 12 (Epoch resource bounds)**.** *Let $d \ge 2$, let $m \in \{1, 2, 4, \ldots\}$, and use the epoch parameters and construction of Appendix B.3, with the valid input records of Appendix A.2. The construction is a uniform finite-control program with at most $2^{12}d$ charged bit actions after each physical reply and peak storage at most*
$$2^{20}\{A + d[P_m + \lceil \log_2(d+1) \rceil]\}.$$
The epoch round count H_m includes construction, full normal/overflow receipt, all probes and updates, report copying, abort, erasure, and resize. It satisfies
$$H_m \le 2^{80}(d+1)^{22}q_m^2. \tag{17}$$
Exactly one complete first-order reply occurs per round. These bounds hold for every legal reply transcript, including epochs before the no-overflow onset.

Proof. We count query emission, resident storage, arithmetic, and round scheduling in that order.

1. Query emission. Write the query coordinates as $x_i = N_i/D_i$, with $N_i \in \mathbb{Z}$ and $D_i \in \mathbb{Z}_{\ge 1}$, and set
$$\ell_i^Q = \lceil \log_2(|N_i| + 1) \rceil, \qquad r_i^Q = \lceil \log_2(D_i + 1) \rceil.$$
Here $p = P_m$ for a useful query; the stored padding query has $p = 0$. The canonical grammar in Appendix A.1 gives
$$|\mathsf{Int}(N_i)| = 2\ell_i^Q + 2, \qquad |\mathsf{Pos}(D_i)| = 2r_i^Q + 1,$$
$$|Q| = \sum_{i=1}^{d}(2\ell_i^Q + 2r_i^Q + 3) + 2(d-1) + p + 9.$$
Generate each unary length by a scan and emit the fields in order:
$$\mathsf{header} \rightarrow \mathsf{coordinate\ records\ and\ separators} \rightarrow \mathsf{vector\ end} \rightarrow \mathsf{precision\ frame}.$$
The final marker is written after all payload bits. A query selector can therefore point only to a complete valid buffer.

2. Resident storage. Recall the mesh parameters from Lemmas 6 and 7:

$$r_\theta = \lceil \log_2(2^{12} d(d+1)) \rceil, \qquad r_d = \lceil \log_2(2^{r_\theta+4} d^2) \rceil.$$

Lemma 8 and the mesh $h_{m,k} = 2^{-(m+2k+r_d)}$ give

$$\begin{aligned} b_{\mathrm{int}} &\le A + d + m + 2K_m + 8, \\ b_{\mathrm{frac}}^{\mathrm{state}} &\le \max\{A, m + 2K_m + r_d\}, \\ b_{\mathrm{frac}}^{\mathrm{probe}} &\le b_m + \max\{A, m + 2K_m + r_d\}. \end{aligned}$$

Here the maximum with A covers the unrounded initial center $c = a$; subsequent centers and factors have the stated mesh precision. To turn these estimates into record lengths, use $d \ge 2$ and $q_m = A + 4d + m + \ell_d + 1$ to obtain

$$\begin{aligned} K_m &= 64d(d+1)(m+1) \le 96d^2(m+1) \le 96d^2 q_m, \\ r_\theta &\le 12 + 2\ell_d, \qquad r_d \le 16 + 4\ell_d \le 16 + 4d \le 12d \le 3q_m. \end{aligned}$$

The first inequality uses $d + 1 \le 3d/2$; the bound $\ell_d \le d$ follows from $d + 1 \le 2^d$. Moreover $A + d + m + 8 \le q_m$, because $3d + \ell_d - 7 \ge 0$. Define the integer and state-fractional length bounds explicitly by

$$I_m = A + d + m + 2K_m + 8, \qquad f_m = \max\{A, m + 2K_m + r_d\}.$$

Then

$$\begin{aligned} I_m &\le (1 + 192d^2) q_m \le 193 d^2 q_m, \\ f_m &\le \max\{q_m, (192d^2 + 4) q_m\} \le 193 d^2 q_m, \\ I_m, f_m &\le b_m/32, \qquad 32 \cdot 193 = 6176 \le 4096d. \end{aligned}$$

Thus a probe has at most I_m integer bits and $f_m^{\mathrm{probe}} \le b_m + f_m \le 33 b_m/32$ fractional bits. For its dyadic representation $N/2^f$, where $0 \le f \le b_m + f_m$, the numerator and denominator magnitude lengths satisfy

$$\ell_N \le I_m + f \le 17 b_m/16, \qquad r_D = f + 1 \le 33 b_m/32 + 1.$$

Consequently the integer framing, even with a coordinate separator, uses at most

$$2\ell_N + 2r_D + 5 \le 2I_m + 4f + 7 \le 67 b_m/16 + 7 < 8 b_m.$$

The final inequality holds since $b_m \ge 4096 d^3 q_m \ge 2^{15}$. Including the vector's start and end markers gives

$$|\mathsf{Vec}| \le d(67 b_m/16 + 7) + 2 < 8 d b_m.$$

The additional P_m precision bits are counted in the control row below. Raw signed words, exact short-times-long products, and their sums fit in $4b_m$ bits: an additional short coefficient contributes at most $U = 2^{10}(\ell_d + 1) \le 2^{11} d \le b_m/32$ bits, and summing at most d terms contributes at most $\ell_d + 1 \le b_m/32$ further carry bits. Indeed, $b_m/32 = 128 d^3 q_m$ and $q_m \ge 4d$ imply $b_m/32 \ge 512 d^4 \ge 2048 d$ for $d \ge 2$. There are at most two successive short factors before a mesh rounding in (14); even allowing four such widths and one carry allowance gives $17 b_m/16 + 4(b_m/32) + b_m/32 = 39 b_m/32 < 4 b_m$. Thus exact temporary products do not require a wider long-word slot. The old report comes from the anchor or an earlier epoch $m' \le m$; the same monotone length bounds therefore cover it. Table 10 lists all simultaneously resident banks. Let $\mathcal{L}_m$ denote their peak total allocated length, including all auxiliary tracks and stale or resize capacity.

Table 10: Storage inventory for an epoch. The aggregate is multiplied by sixteen to include delimiters, boundary and position tracks, adjacent head marks, and finite flags. Old, new, stale, and scratch capacity count simultaneously.

Resident regions	Cells before the common factor
Original input and startup parser/scratch	$9A + 64d$
Converted anchor list	$4(A + d)$
Two active and two resize/stale factor banks, with scratch	$36 d^2 b_m$
Centers, queries, candidates, and both report banks	$128 d b_m$
Unary rulers and query precision headers	$8 P_m$
Useful delivery, conversion scratch, scalar fields, padding delivery	$192(d + 1) P_m$

The factor row allocates nine factor-shaped regions, each with d^2 words of at most $4b_m$ bits: four current/next or resize/stale banks, and five scratch regions for the inflated factor, intermediate factor, short coefficients, products/sums, and rounding checks. The vector row allows sixteen regions of at most $8db_m$ bits: four current/next or resize/stale center banks, two query-coordinate banks, two candidate banks, two report banks, and six temporary vector regions. The two query precision headers are counted separately:

$$\underbrace{2P_m}_{\text{query headers}} + \underbrace{2P_m}_{\text{current/replacement precision rulers}} + \underbrace{32(4b_m)}_{\text{reusable control regions}} \le 8P_m,$$

because $P_m = 16db_m \ge 32b_m$. The control regions provide eight pairs of loop rulers and eight pairs of operand-length or shift rulers; each pair retains an old ruler while its replacement is written. At most eight loop levels are active, and these regions are reused between allocation, probe, and update phases. Apart from the separately counted precision rulers and scalar-field offsets below, their lengths are bounded by $4b_m$: q_m, m, K_m, the mesh exponents, and the short-word loop bounds are all smaller than this word capacity. Dimension constants remain short binary words until a loop needs a ruler. Erase obsolete query records and headers before resizing their banks, so only two query headers coexist. Each completed query consists of its coordinate record followed by its precision header.

Construct coefficient entries one at a time: keep the selected $\bar{M}$ in its factor-shaped region and $u, \bar{v}$ in temporary vector regions. Bisection brackets and intermediate interval endpoints use the products/sums and rounding-check scratch, which is reused for the long update after coefficient construction ends.

For the last row, a useful delivery occupies at most $48(d+1)P_m$ bits. At most $2d+6$ scalar records cover the $d+1$ probe values, d differences, two candidate values, and three decoding/arithmetic temporaries. Each needs at most $16P_m + 8 \le 24P_m$ bits. Padding uses $R(0) = 8d + 12 \le 12(d+1)P_m$ bits. Their sum is bounded by

$$\big[48(d+1) + 24(2d+6) + 12(d+1)\big]P_m = (108d + 204)P_m \le 192(d+1)P_m.$$

The inequality is equivalent to $12 \le 84d$. Any ruler for an offset within a long scalar field uses one of the three scalar decoding/arithmetic scratch records, rather than the $8P_m$ control allowance: its length is at most $16P_m + 8 \le 24P_m$. Complete replies are scanned using their framing, terminal marker, and stored progress marks; no additional ruler of length $R(P_m)$ is needed.

The startup row excludes the converted list, which is accounted for separately. Indeed, with $r = \lceil \log_2(d+1) \rceil$, the source tracks, reusable rulers, dimension parser, and 64 fixed flag bits from Lemma 3 occupy at most

$$7A_{\mathrm{src}} + 9r + 89 \le 9A + 64d.$$

The slack in this inequality is

$$9A + 64d - (7A_{\mathrm{src}} + 9r + 89) = 2A_{\mathrm{src}} + 9r + 64d - 80 \ge 0,$$

using $A = A_{\mathrm{src}} + 2r + 1$, $A_{\mathrm{src}} \ge 21$, and $d, r \ge 1$. The common factor sixteen also covers the converted list's head-position track and the fixed number of boundary/position tracks for every region.

Thus the visited extent satisfies

$$\begin{aligned}\mathcal{L}_m &\le 16\{9A + 4(A+d) + 64d + 36d^2 b_m \\ &\qquad + 128db_m + 8P_m + 192(d+1)P_m\} \\ &< 2^{32}(d+1)^5 q_m.\end{aligned}$$

For the strict numerical bound in the last line, substitute $b_m = 2^{12}d^3 q_m$ and $P_m = 2^{16}d^4 q_m$. The terms $128db_m$ and $8P_m$ together equal $2^{20}d^4 q_m$. Also $13A + 68d \le 30q_m$, since $A \le q_m$ and $4d \le q_m$. Bounding d^5, d^4, and $(d+1)d^4$ by $(d+1)^5$ therefore gives

$$\begin{aligned}\mathcal{L}_m &\le 16[30 + 36 \cdot 2^{12} + 2^{20} + 192 \cdot 2^{16}](d+1)^5 q_m \\ &= 220{,}463{,}584\,(d+1)^5 q_m < 2^{28}(d+1)^5 q_m < 2^{32}(d+1)^5 q_m.\end{aligned}$$

To compare this with the prescribed budget, use $q_m \ge A + 4d$ and

$$dP_m = 2^{16}d^5 q_m, \qquad d^2 b_m = 2^{12}d^5 q_m, \qquad (d+1)^5 \le 2^3 d^5 \quad (d \ge 2).$$

Then

$$\mathcal{L}_m < 2^{32}(d+1)^5 q_m \le 2^{35} d^5 q_m = 2^{19} d P_m$$
$$\le 2^{20}\{A + d[P_m + \lceil \log_2(d+1) \rceil]\}.$$

Here $(d+1)^5/d^5 = (1+1/d)^5 \le (3/2)^5 = 243/32 < 8$, which explains the factor 2^3. Also $2^{35} d^5 q_m = 2^{19} d(2^{16} d^4 q_m) = 2^{19} d P_m$. This calculation applies to $d \ge 2$; startup at $p = 0$ was bounded in Lemma 3, and $d = 1$ uses the separate inventory of Lemma 11. The input length A always includes $\mathsf{Pos}(d)$. Blank capacity in visited regions also counts; neither unused cells nor head positions provide uncharged storage.

3. Serial arithmetic and addressing. Write $\ell = \ell_d$, recall $N = 64(\ell+1)$ from Lemma 6, and set $U = 2^{10}(\ell+1)$ and $\mathsf{D}_\ell = 2^{2N+8\ell}$. Short coefficients use the common dyadic denominator D_ℓ; their numerators and temporary comparison operands have at most U bits, with a $2U+2$-cell product accumulator. Center, factor, and probe words have at most $4b_m$ bits; observed scalar fields and their differences use at most $16P_m + 8$. The inventory includes accumulators, unary length rulers, carry tracks, and head-position marks.

We use two explicit ways of accessing those records.

1. A *traversal-equivalent* is an allowance of $16\mathcal{L}_m$ charged actions. It covers a marked pass over the layout: per visited position allow four reads, four writes, and eight one-cell moves across the fixed tracks. Return passes are counted separately. Region selection scans headers from the permanent left marker; offsets are reached by consuming unary rulers. No binary integer serves as a random-access address.

2. For a long addition, comparison, copy, shift, or mesh rounding, select operand regions, align their marked ends using zero padding, stream through them, and return the heads. Eight traversal-equivalents cover these steps. A carry or borrow occupies finite control; rounding away from zero also keeps a flag recording whether any discarded bit was nonzero. For a product by a short coefficient, visit its at most U bits and add the corresponding shifted long word to a fresh accumulator. Initial clearing, sign handling, and returns give at most $16(U+1)$ traversal-equivalents per product.

These are serial addition and shift-and-add algorithms (Brent & Zimmermann, 2010, Sections 1.2–1.4); operand selection and marked returns are charged here as well.

Short arithmetic can be implemented even more conservatively: simulate each local bit action by one traversal-equivalent, including location of its operand. Table 11 counts local actions before this simulation.

Table 11: Local bit actions for short arithmetic. Each pass costs at most sixteen actions per position, and inner-loop pass counts include returns. Add the two multiplication rows, or the two division rows, for the complete macro.

Step	Repetitions	Cost each	Subtotal
Copy, add, compare, or shift: forward and return passes	1	$32(U+2)$	$32(U+2)$
Multiplication: clear accumulator	1	$16(2U+2)$	$16(2U+2)$
Multiplication: inspect bit, shifted add, and returns	U	$64(2U+2)$	$64U(2U+2)$
Division: initialize remainder and quotient	1	$32(U+2)$	$32(U+2)$
Division: shift, compare/subtract, restore, and returns	U	$96(U+2)$	$96U(U+2)$

Multiplication uses at most four passes per multiplier bit over a $2U+2$-cell accumulator. Binary long division uses at most six passes per quotient bit over a $U+2$-cell remainder: shift, compare, subtract, conditional restore, and returns. Signs are handled at the input and output boundaries. For

$U \geq 2$, summing the rows gives

$$T_{\mathrm{mul}}^{\mathrm{local}} \leq 16(2U+2)(4U+1) \leq C_{\mathrm{sh}}(U),$$
$$T_{\mathrm{div}}^{\mathrm{local}} \leq 16(U+2)(6U+2) \leq C_{\mathrm{sh}}(U),$$
$$C_{\mathrm{sh}}(U) := 128U^2 + 256U.$$

Expanding the two products and comparing their coefficients gives

$$T_{\mathrm{mul}}^{\mathrm{local}} \leq 128U^2 + 160U + 32, \quad C_{\mathrm{sh}}(U) - (128U^2 + 160U + 32) = 96U - 32 \geq 0,$$
$$T_{\mathrm{div}}^{\mathrm{local}} \leq 96U^2 + 224U + 64, \qquad C_{\mathrm{sh}}(U) - (96U^2 + 224U + 64) = 32(U+2)(U-1) \geq 0.$$

The remaining row costs $32(U+2) \leq C_{\mathrm{sh}}(U)$. Thus the same bound applies to every short macro without treating a product or quotient as one bit action. Only dimension-dependent words use these quadratic routines.

The coefficient construction of Lemma 6 uses at most

$$S_d = 128 + 48N + 50d + 96d^2$$

short macros, as itemized in Table 7. The term $48N$ is four loops, N iterations per loop, and twelve macros per iteration. Since $N = 64(\ell+1)$, $\ell + 1 \leq d + 1$, and $U \leq 2^{11}(d+1)$,

$$S_d < 2^{13}(d+1)^2, \qquad C_{\mathrm{sh}}(U) < 2^{30}(d+1)^2,$$
$$S_d C_{\mathrm{sh}}(U) < 2^{43}(d+1)^4 < 2^{41}(d+1)^6 \qquad (d \geq 2).$$

For an explicit check, put $D = d + 1 \geq 3$. Then $48N \leq 3072D$, and

$$S_d \leq 128 + 3072D + 50D + 96D^2 \leq 3346D^2 < 2^{13}D^2,$$
$$C_{\mathrm{sh}}(U) \leq 2^{29}D^2 + 2^{19}D < 2^{30}D^2,$$
$$\frac{2^{43}D^4}{2^{41}D^6} = \frac{4}{D^2} \leq \frac{4}{9} < 1.$$

The last expression counts traversal-equivalents because each local short-word action was allowed one full traversal. Thus addressing, not just arithmetic, is included.

Table 12 expands one probe and receipt path; the entire first-order reply is scanned even though only its value is retained. For the update in Table 13, matrix multiplication uses three nested loops: for each output row and column, clear an accumulator and visit all d summands. The vector product uses two nested loops. Products and sums are exact dyadics until the stated mesh rounding.

Table 12: Traversal-equivalents for one probe and complete receipt. Each streamed long-word macro costs at most eight traversal-equivalents.

Step	Repetitions	Cost each	Subtotal
Copy/shift selected factor entry and add center	d	16	$16d$
Count lengths and emit numerator/denominator frames	d	16	$16d$
Write query/precision framing; scan and decode full reply	1	16	16
Compare value and, if smaller, copy candidate and its value	1	$8(d+2)$	$8(d+2)$
Select/return heads, update rulers, erase query scratch	1	$8(d+1)$	$8(d+1)$
Total for one probe			$48d + 40 \leq 64(d+1)$

Indeed, direct addition gives

$$J_d = 16(U+1)(d^3+d^2) + 8d^3 + 72d^2 + 32d + 16 < 2^{17}(d+1)^4.$$

Table 13: Traversal-equivalents for (14). Inflation by $\alpha = 1 + \delta_d$ is a binary shift and addition. The old factor is retained while forming the scratch factor αB. The sum of subtotals is J_d.

Step	Repetitions	Cost each	Subtotal
Products for $B\bar{v}$	d^2	$16(U+1)$	$16(U+1)d^2$
Products for $(\alpha B)\bar{M}$	d^3	$16(U+1)$	$16(U+1)d^3$
Accumulate products and add c	d^3+d^2+d	8	$8(d^3+d^2+d)$
Inflate B and $\widehat{D}$	$2d^2$	16	$32d^2$
Round $c_+, \widehat{D}, B_+$	$d+2d^2$	8	$8(d+2d^2)$
Clear, check completed banks, and commit selectors	d^2+d+1	16	$16(d^2+d+1)$
Total			$J_d < 2^{17}(d+1)^4$

To verify the stated bound for J_d, let $D = d+1$. Since $U+1 \le 2^{12}D$ and $d^3+d^2 = d^2D \le D^3$,

$$J_d \le 2^{16}D^4 + (8+72+32+16)D^3 = 2^{16}D^4 + 128D^3 < 2^{17}D^4.$$

For direction formation, allow d subtraction macros, $2d$ macros for magnitude comparisons and replacement of the current maximum, d truncation macros, and $2d+4$ macros for the common-scale ruler, leading-one search, and marked returns. This is at most $6d+4 \le 8(d+1)$ long macros, each costing eight traversal-equivalents. Hence their total is at most

$$8 \cdot 8(d+1) = 64(d+1) \le 32(d+1)^2 \qquad (d \ge 2).$$

No long-word division is used: rescaling locates a binary position and copies t_d bits.

Table 14: Complete per-cut traversal ledger. Multiply execution counts by per-execution costs and sum the last column. Each traversal-equivalent costs at most $16\mathcal{L}_m$ actions. Aborted or zero-direction paths use a prefix plus cleanup.

Operation	Executions	Cost each	Subtotal
Probe, full receipt, candidate comparison	$d+1$	$64(d+1)$	$64(d+1)^2$
Differences, leading-one search, truncation	1	$32(d+1)^2$	$32(d+1)^2$
Certified short-word macros	S_d	$C_{\mathrm{sh}}(U)$	$S_dC_{\mathrm{sh}}(U)$
Stored center/factor update	1	J_d	J_d
Loop selection, zero branch, abort cleanup	1	$64(d+1)$	$64(d+1)$
Total, denoted by D_d			$< 2^{42}(d+1)^6$

Writing $D = d+1$, the exact sum of the noncoefficient rows is $96D^2 + J_d + 64D$. Consequently

$$96D^2 + J_d + 64D < (96 + 2^{17} + 64)D^4 < 2^{18}D^4.$$

Adding the coefficient bound $S_dC_{\mathrm{sh}}(U) < 2^{43}D^4$ and using $2^{43} + 2^{18} < 2^{44}$ gives

$$D_d < 2^{44}(d+1)^4 \le 2^{42}(d+1)^6 \qquad (d \ge 2).$$

4. From cuts to an epoch. Before allocating the first high-dimensional layout, establish its precision budget. Starting from $p = 0$, retain the input, converted anchor, and selected report, and build only the dimension and length rulers described below. Their lengths are at most

$$q_1 = A + 4d + \ell_d + 2 \le A + 5d + 2.$$

Keep a complete precision-stage ruler 1^t0, initially $t = 0$. Four nested dimension loops append q_1 marks each, producing $t = d^4q_1$; sixteen successive doublings then produce $t = 2^{16}d^4q_1 = P_1$. To append a mark, first write a new terminal zero one cell beyond the current zero, then change the old zero to one. Until that last write the old ruler is complete; afterwards the complete ruler has length

parameter $t+1$. Thus its declared precision increases only after the replacement is valid. A doubling copies the old marks to a temporary ruler, appends one mark for each copied mark, and erases the temporary ruler. No useful query is issued during this process.

At a prefix with declared precision t, two precision rulers have length at most $t+2$ each. The startup and converted-anchor rows of Table 10 use at most $13A + 68d$ cells before the common track factor. The padding buffer uses $R(0) = 8d + 12$; the thirty-two reusable control regions need only $q_1 + 16$ cells each at this step. Thus

$$13A + 68d + R(0) + 32(q_1 + 16) \le 64(q_1 + 16),$$

since $13A + 76d + 12 \le 19q_1$ and $q_1 \ge A + 4d + 2$. Allowing the same sixteen tracks, including delimiters and head marks, bounds the transition storage by

$$\begin{aligned} S_{\mathrm{boot}}(t) &\le 16\{2(t+2) + 64(q_1 + 16)\} \\ &\le 2^{14}(A + d + t + 1) \\ &\le 2^{15}\{A + d[t + \ell_d]\} < 2^{20}\{A + d[t + \ell_d]\}. \end{aligned}$$

For the second line, substitute $q_1 \le A + 5d + 2$ and compare coefficients; for the third, use $A \ge 1$, $d \ge 2$, and $\ell_d \ge 1$. The bound includes the extra terminal cell before t increases. These regions and two precision rulers reuse the startup and control regions of Table 10; their emission, copying, erasure, and head returns are included in the allocation passes below. Only after $t = P_1$ is complete do we allocate the remaining layout under the established precision budget.

For subsequent epochs, the existing precision already suffices during resize. With $Q_0 = A + 4d + \ell_d + 1$,

$$\frac{P_{2m}}{P_m} = \frac{Q_0 + 2m}{Q_0 + m} < 2, \qquad \mathcal{L}_{2m} < 2^{19} d P_{2m} < 2^{20} d P_m.$$

The peak layout includes old and replacement banks, so it fits the old budget until the new precision ruler is complete.

Generate the layout by nested unary loops. The decoded dimension is initially binary. To obtain a unary dimension ruler, read its bits from most to least significant: for the next bit β, replace a current ruler of length t by two copies followed by β marks, giving length $2t + \beta$. Keep the old ruler until its replacement is complete, then erase it. If t_j is the integer represented by the first j binary digits and the complete word has length $r = \ell_d$, then $t_j = \lfloor d/2^{r-j} \rfloor$. Therefore

$$\sum_{j=1}^{r} t_j \le d \sum_{h=0}^{r-1} 2^{-h} < 2d.$$

Every copied or erased mark is charged, and this total fits within the allocation charge below. The same input scan supplies a ruler of length ℓ_d, its binary word length. Scan the complete input to build its length ruler; concatenate rulers to obtain q_m. Four dimension loops and sixteen doublings generate P_m; the first epoch reuses the ruler just constructed. Generate b_m, K_m, row/column rulers, and record lengths in the same way. At most eight nested counters suffice for any record in the inventory. An advance and a reset each cost at most two marked passes per counter. Including location and initialization of the next cell and head returns, allow sixty-four traversal-equivalents per emitted layout cell. More explicitly, at most eight counter levels can each require an advance and reset, using at most $8(2+2) = 32$ traversals. A further four streaming macros, each with eight traversals, locate the destination, initialize its data and boundary marks, restore the source heads, and restore the emitter mark. The total is $32 + 4 \cdot 8 = 64$ traversal-equivalents per cell. The largest variable loop product is $d^5 q_m$; five dimension loops, one length loop, and two fixed bank/repetition loops use at most these eight levels. Ruler construction is included in this pass.

Table 15 adds all epoch costs. Publication retains the old report until the replacement is complete; ending scans erase obsolete banks, rather than merely dropping their selectors.

The publication/cleanup row allows sixty-four traversal-equivalents per group for copying, checking, erasing, and head returns. Useful replies are scanned in the cut row; the last row only adds their delivery/control rounds. More explicitly, let U_m be the actual number of useful calls. Summing the first three rows gives

$$T_{\mathrm{main}} \le 1024\mathcal{L}_m^2 + 16K_m D_d \mathcal{L}_m + 1024(d^2 + d + 1)\mathcal{L}_m.$$

Table 15: Additive epoch work and round accounting. The first three rows count main-program bit actions. Executing one such action per padding round and adding the last row bounds physical rounds. Padding processing is charged within each round as shown below.

Stage	Repetitions	Cost each	Subtotal
Allocate/resize and initialize layout cells and rulers	$\le \mathcal{L}_m$	$1024\mathcal{L}_m$	$1024\mathcal{L}_m^2$
Probe and update iteration	$\le K_m$	$16D_d\mathcal{L}_m$	$16K_mD_d\mathcal{L}_m$
Publish, erase, and return, per record group	$\le d^2+d+1$	$1024\mathcal{L}_m$	$1024(d^2+d+1)\mathcal{L}_m$
Useful-call and boundary-control rounds	$\le (d+1)K_m+2$	1 round	$(d+1)K_m+2$ rounds

The schedule in Step 5 assigns one padding round to each pending main action and separate rounds to useful delivery and boundary control. Consequently

$$H_m \le T_{\mathrm{main}} + U_m + 2, \qquad U_m \le (d+1)K_m.$$

Using $\mathcal{L}_m < 2^{32}(d+1)^5 q_m$ and $K_m \le 2^6(d+1)^2 q_m$, the rows give

$$\begin{aligned} 1024\mathcal{L}_m^2 &< 2^{74}(d+1)^{10}q_m^2, \\ 16K_mD_d\mathcal{L}_m &< 2^{84}(d+1)^{13}q_m^2, \\ 1024(d^2+d+1)\mathcal{L}_m &< 2^{42}(d+1)^7 q_m, \\ (d+1)K_m+2 &\le 2^7(d+1)^3 q_m. \end{aligned}$$

For example, the cut term combines exponents $4+6+42+32=84$ of two and $2+6+5=13$ of $d+1$. For the other rows, $d^2+d+1 \le (d+1)^2$, $q_m \ge 1$, and $K_m \le 2^6(d+1)^2 q_m$ give exactly the displayed powers. Let $D = d+1 \ge 3$. Relative to the cut upper bound $2^{84}D^{13}q_m^2$, the sum of all four rows is less than

$$1 + \frac{2^{-10}}{D^3} + \frac{2^{-42}}{D^6 q_m} + \frac{2^{-77}}{D^{10} q_m} < 1 + 2^{-10} + 2^{-42} + 2^{-77} < 2.$$

Thus

$$H_m < 2^{85}(d+1)^{13}q_m^2 \le 2^{80}(d+1)^{22}q_m^2,$$

since the second inequality is equivalent to $32 \le (d+1)^9$, and $(d+1)^9 \ge 3^9 > 32$. An abort or zero direction follows a prefix of the charged cut path plus its budgeted cleanup; it cannot increase this maximum. This includes early aborts, all reply fields, and head returns. The following schedule converts the main-work count into physical rounds.

5. Per-round work. Use the padding handler of Table 5. In a padding round, scan and erase the short reply, return its head, and execute one suspended main primitive. This costs at most

$$14R(0) + 2 = 112d + 170 \le 282d < 4096d.$$

A useful-delivery round receives its complete buffer externally and uses one charged selector write to restore the padding query for the following round. Its useful-reply scan then advances as suspended main work during later padding rounds. Thus useful-delivery rounds also obey the cap. Publication and query selection are charged main writes, permitted only after the destination record is complete. This schedule applies unchanged to receipt, abort, allocation, and cleanup paths. All loops have fixed nesting depth and read their bounds from stored data or rulers. Scans, arithmetic, and selector changes use the same fixed finite controller for every d, m; dimension-dependent words are charged data, not additional control states. This proves the claimed uniformity as well as the resource bounds. □

To distinguish useful calls, delivered bits, and computation, retain U_m for the useful-call count and H_m for the total round count in epoch m ($d \ge 2$). Then

$$\begin{aligned} U_m &\le (d+1)K_m \le 64d(d+1)^2(m+1), \\ R(P_m) &= 4 + 2(d+1)(16P_m+4) \le 48(d+1)P_m, \\ R(0) &= 4 + 8(d+1). \end{aligned}$$

The reply-length estimate follows directly from

$$R(P_m) = 32(d+1)P_m + 8(d+1) + 4,$$

because $P_m \geq 1$ and $8(d+1)+4 \leq 16(d+1)P_m$. Consequently the complete epoch delivers

$$\underbrace{U_m R(P_m)}_{\text{useful replies}} + \underbrace{(H_m - U_m)R(0)}_{\text{padding replies}} \quad \text{bits.}$$

Through round n, the call count is n and charged work is at most $2^{12}dn$. Report copies are counted in epoch work; keeping a completed report visible requires no new copy.

B.9 Serial Scheduling and Publication

Lemma 13 (A complete report in every round)**.** *For $d \geq 2$ use the implementation and parameters of Lemma 12; for $d = 1$ use Lemma 11. With valid input records as in Appendix A.2, these implementations can be scheduled so that every round has one oracle call, obeys the Wd bit-action cap, and ends with a complete stored public report. No useful reply arrives before the preceding useful reply has been fully consumed. Startup, aborts, allocation, and publication obey the same rules.*

Proof. Initially select the immutable anchor as both query and report. Round one calls $(a, 0)$ before any input-sized copy is attempted. The transducer of Lemma 3 then advances one charged transition after each complete padding reply. Hence decoding the dimension and converting the anchor take $N_{\text{startup}} \leq 128(A+d)+1$ counted rounds, and $z_n = a$ throughout this startup.

The controller uses the five modes in Table 16.

Table 16: Serial controller modes. Each operation may pause between bit actions; a selector changes only when its destination is complete.

Mode	Suspended work	Completion or branch rule
build	Write rational coordinates and unary precision.	Issue the complete query at a round start, then restore $(a, 0)$.
receive	Scan status, all framed fields, and terminal marker.	Keep the useful buffer; permit no new useful call during this scan.
compute	Build new banks while preserving old ones.	Overflow goes to cleanup with the old report; zero direction goes to publication.
publish	Copy the candidate to the inactive report bank.	Switch the report selector after its terminal marker is complete.
erase	Clear obsolete storage by charged scans.	Reuse only the cleared regions.

A useful-delivery round ends by writing the padding-query selector; its next useful-buffer operation is postponed to a later padding round. Every padding round during suspended main work consumes its separate short reply and then executes exactly one charged main primitive. At entry and exit, at most two additional rounds execute the boundary-control transitions. A new useful call is permitted only after the previous useful scan is complete and the next query is fully stored.

To count rounds explicitly, let T be the number of suspended main primitives in an epoch and U its number of useful calls. There are at most T padding rounds that advance those primitives, at most U separate useful-delivery rounds, and at most two boundary-control rounds:

$$H \leq T + U + 2.$$

All writes selecting a newly completed query or report are included in T. The return-to-padding selector write is charged separately in its useful-delivery round. External oracle evaluation and delivery are not charged main actions. The mode rules preserve two invariants:

- A selected query or report is a complete stored vector, including while another bank is being copied or an epoch is aborted.
- A useful reply is never overwritten during receipt; completion of its full scan precedes the next useful delivery round.

The invariants hold initially because the selected anchor is a complete input record. Each possible transition preserves them:

1. Building, computing, and erasing operate on inactive regions, so they do not invalidate the selected query or report.
2. Receipt writes to its dedicated useful buffer. Every other call during its scan is padding and writes only to the disjoint padding buffer. Even after an overflow status is read, receipt continues through the terminal marker before cleanup begins.
3. Publication first completes the replacement vector and its terminal marker, then changes its selector by a charged write. The former selected bank becomes erasable only after this write.
4. A useful query is selected only after its record is complete and the preceding useful terminal marker has been consumed. The call takes place at the following round start, so useful deliveries occur in strictly later rounds than preceding scan completion.

Induction over individual charged primitives and round boundaries therefore proves both invariants at every time, including a suspension in the middle of a multi-action group move.

The padding-round bound from Table 5 is

$$\mathsf{work}_{\mathrm{round}} \le 14R(0) + 2 = 112d + 170 \le 282d < 4096d.$$

A useful-delivery round uses only its selector write and has lower charged work. These bounds hold for every $d \ge 1$. The epoch clocks in Lemmas 12 and 11 include the at most $U + 2$ additional rounds described above. All banks used by these modes were included in the storage inventories. Consequently the schedule respects both budgets at every point, not only when averaged over an epoch. □

B.10 Every-Round Convergence and Required Precision

The previous lemmas give an error bound only after an epoch is completed. We now prove the full theorem by summing the costs of geometrically increasing epochs and using the previous report while the next one is being computed. This is the doubling-trick principle applied to precision (Besson & Kaufmann, 2018, Sections 2–3).

Theorem 14 (Every-round upper bound, restated)**.** *There is one deterministic uniform program satisfying the model of Appendix A, and for each dimension $d \ge 1$ there are constants $\gamma_d, C_d > 0$ depending only on d and the fixed budgets. For every fixed dyadic anchor $a \in \mathbb{Q}^d$ and every fixed convex $F \in C^1(\mathbb{R}^d)$ with globally Lipschitz gradient and an attained minimum, there are finite integers $n_0 \ge 0$, $n_* \ge n_0$, and a constant $C_0 > 0$ such that every legal rounding transcript satisfies*

$$F(z_n) - F^* \le C_0 \exp(-\gamma_d \sqrt{n - n_0}) \qquad (n \ge n_*).$$

The program has made exactly n calls by round n and at most Wdn charged bit actions. For $0 < \varepsilon \le C_0$, accuracy ε is attained and maintained by

$$n_0 + C_d \lceil \log_+(C_0/\varepsilon) \rceil^2 \quad \textit{rounds}, \qquad \log_+(t) = \max\{1, \log t\}.$$

For sufficiently small ε, the maximum working precision through the completed epoch attaining this accuracy is at most $C_d[A + \lceil \log_+(C_0/\varepsilon) \rceil]$. This is Theorem 1.

Proof. The $d \ge 2$ program is defined in Appendix B.3; the $d = 1$ branch is Lemma 11. Lemma 3, the resource lemmas, and Lemma 13 give a uniform finite-control implementation, legal storage, one

call per round, and $\mathcal{B}_n \le Wdn$, where $\mathcal{B}_n$ is cumulative charged bit work and $W = 2^{12}$. No step uses L, x^*, a target accuracy, or a terminal horizon.

Write $\mathcal{E} = \{1, 2, 4, \ldots\}$ for the epoch indices. On each legal transcript, let H_m count all rounds of epoch m and let $n_{\mathrm{complete}}(m)$ be its actual completion round, after publication or abort and final cleanup. For $d = 1$, H_m denotes the quantity $H_m^{(1)}$ of Lemma 11. Both resource lemmas bound every epoch, including early aborts; together with finite startup, this ensures that the run reaches and finishes every index in $\mathcal{E}$.

Recall $\ell_d = \lceil \log_2(d+1) \rceil$ and let A be the encoded input length. Define the clock constants and length scale by

$$Q_0 = A + 4d + \ell_d + 1, \qquad K_d = 2^{80}(d+1)^{22}, \qquad q_m = Q_0 + m.$$

Here K_d is a dimension-dependent clock constant, distinct from the cut cap K_m. For $d = 1$, $q_m = A + m + 6$ and $H_m < 2^{32} q_m^2 \le K_d q_m^2$ by Lemma 11. For $d \ge 2$, Lemma 12 gives $H_m \le K_d q_m^2$.

1. A transcript-independent onset. Lemmas 8 and 10, or Lemma 11, provide a finite threshold after which every useful reply is normal and every completed epoch satisfies its error bound, uniformly over all legal rounding choices for this fixed instance. Choose a power of two m_0 above that threshold and above Q_0. Choose $C_0 > 0$ at least as large as the constant in the completed-epoch bound, so that

$$F(z^{(m)}) - F^* \le C_0 4^{-m} \qquad (m \in \mathcal{E},\ m \ge m_0).$$

These are analysis constants; the program neither computes nor receives them. An explicit deterministic upper bound for startup and all epochs through m_0 is

$$n_0 = \left\lceil 128(A+d) + 1 + K_d \sum_{\substack{s \in \mathcal{E} \\ s \le m_0}} (Q_0 + s)^2 \right\rceil.$$

The sum is finite. Hence epoch m_0 has completed by round n_0 on every legal transcript. The same m_0, C_0, n_0 work for all such transcripts, although their actual completion rounds can differ. All subsequent completed epochs are successful as well. Set $n_* = n_0 + 1$.

2. From a running epoch to an every-round rate. For $m \ge m_0$, the choice $m_0 \ge Q_0$ gives

$$q_m = Q_0 + m \le 2m, \qquad H_m \le K_d q_m^2 \le 4K_d m^2.$$

Fix $n \ge n_*$ and let m be the epoch containing that round. If a round is at a boundary, assign it to the epoch whose cleanup or initialization it executes. Since epoch m_0 has completed by n_0, this index satisfies $m \ge 2m_0$. Let $t_0 = n_{\mathrm{complete}}(m_0) \le n_0$ on the chosen transcript. The rounds after t_0 and through n belong to epochs $2m_0, 4m_0, \ldots, m$. Bound the elapsed part of the current epoch by its full duration, and apply the clock bound only to these indices:

$$\begin{aligned} n - n_0 \le n - t_0 &\le \sum_{\substack{s \in \mathcal{E} \\ 2m_0 \le s \le m}} H_s \\ &\le 4K_d \sum_{\substack{s \in \mathcal{E} \\ 2m_0 \le s \le m}} s^2 \le 4K_d m^2 \sum_{j \ge 0} 4^{-j} \\ &= \frac{16}{3} K_d m^2 \le 10 K_d m^2. \end{aligned}$$

The last comparison adds nonnegative numerical terms to a geometric series; it makes no assumption on the costs of epochs before m_0. By Lemma 13, before publication the complete selected report comes from epoch $m/2 \ge m_0$; after publication it comes from epoch m. Both have error at most $C_0 4^{-m/2}$, so

$$F(z_n) - F^* \le C_0 4^{-m/2} = C_0 e^{-(\log 2)m}.$$

Because $n - n_0 \le 10K_d m^2$ and $m > 0$, taking square roots gives $m \ge \sqrt{n - n_0}/\sqrt{10K_d}$. The exponential is decreasing in m, and therefore

$$F(z_n) - F^* \le C_0 \exp\left[-\frac{\log 2}{\sqrt{10K_d}} \sqrt{n - n_0} \right].$$

This proves the claimed rate with $\gamma_d = (\log 2)/\sqrt{10K_d}$.

3. Accuracy and its maintenance. For $0 < \varepsilon \le C_0$, define

$$r = \log_4(C_0/\varepsilon) \ge 0, \qquad h = \lceil \log_+(C_0/\varepsilon) \rceil \ge 1.$$

Then $r = \log(C_0/\varepsilon)/\log 4 \le h/\log 4$. Choose the smallest power of two m satisfying $m \ge \max\{m_0, r\}$. Its completed report obeys

$$C_0 4^{-m} \le C_0 4^{-r} = \varepsilon.$$

Every later completed epoch has a larger index and hence the same error guarantee. Between publications the selected report remains one of these completed records, which proves maintenance as well as attainment.

If $r \le m_0$, the chosen epoch is m_0 and the guarantee already holds by n_0. If $r > m_0$, minimality of the power of two gives $m/2 < r \le m$, and hence $m < 2r$. Using the same finite clock sum,

$$\begin{aligned} n_{\mathrm{complete}}(m) - n_0 \le n_{\mathrm{complete}}(m) - t_0 &= \sum_{\substack{s \in \mathcal{E} \\ 2m_0 \le s \le m}} H_s \\ &\le 10K_d m^2 < 40K_d r^2 \\ &\le \frac{40K_d}{(\log 4)^2} h^2. \end{aligned}$$

This also holds if $n_{\mathrm{complete}}(m) < n_0$; the derivation does not require that completion to occur after n_*. Thus any dimension constant $C_d \ge 40K_d/(\log 4)^2$ gives the asserted round budget for every $0 < \varepsilon \le C_0$.

4. Required working precision. Make "sufficiently small" explicit by taking $0 < \varepsilon < C_0 4^{-m_0}$, so that $r > m_0$. For $d \ge 2$, useful precision increases with the epoch index; startup has precision zero and old retained banks have no larger precision than the current bank. For the epoch m chosen in Step 3, the maximum working precision through its completion is therefore P_m, and

$$\begin{aligned} P_m &= 2^{16} d^4 (A + 4d + \ell_d + 1 + m) \\ &< 2^{16} d^4 \left(A + 4d + \ell_d + 1 + \frac{2h}{\log 4} \right) \\ &\le 2^{16} d^4 \left(4d + \ell_d + 2 + \frac{2}{\log 4} \right) (A + h). \end{aligned}$$

For the last step, use $A + h \ge 1$, $A \le A + h$, and $h \le A + h$. For $d = 1$, the largest precision is $p = m < 2h/\log 4$. Thus one common choice valid for both conclusions is

$$C_d = \max\left\{ 1, \frac{40K_d}{(\log 4)^2}, 2^{16} d^4 \left(4d + \ell_d + 2 + \frac{2}{\log 4} \right), \frac{2}{\log 4} \right\}.$$

It depends only on dimension and the fixed budgets.

Finally, for successful epochs $m \ge m_0$ and $d \ge 2$, $q_m \le 2m$ implies $P_m \le 2^{17} d^4 m$, or $m \ge P_m/(2^{17} d^4)$. Substituting this inequality in the decreasing error envelope yields

$$F(z^{(m)}) - F^* \le C_0 4^{-m} \le C_0 \exp\left[-\frac{\log 4}{2^{17} d^4} P_m \right].$$

In one dimension the corresponding formula is $C_0 4^{-p}$. These precision bounds describe the run; the program is not given ε and may continue increasing its precision.

□

C Proof of the Lower Bound

This chapter proves Theorem 2, including competitors with unrestricted storage. After the overview, the argument has three parts:

1. *Build the family.* Construct four convex patches, prove report separation, and realize every infinite nested sequence as one smooth function.
2. *Preserve the transcript.* Place coarse replies in common rounding cells, then prove that every adaptive reply agrees with all remaining completions.
3. *Force slow convergence.* Count serial scans when refinements continue; choose a persistent bad completion when they stop. Combine both cases into the fixed-instance theorem and the corresponding bit-work bound.

C.1 Overview and Proof Dependencies

Fix the competing program before constructing its objective. At depth j, the unresolved interval I_j has four possible children. The construction responds according to the information a query can reveal:

- A coarse query has the same fixed nearest-grid reply for all remaining completions, so the interval stays unresolved.
- A sufficiently precise query triggers refinement. Choose a child away from that query and the preceding public report; the report then has a definite gap for every deeper completion.

Two scales are deliberately different:

$$|I_j| = \tfrac{1}{2}\,64^{-j}, \qquad s_j = \tfrac{1}{200}\,128^{-j}.$$

Here s_j is the magnitude of the derivative at the active endpoints. Their product decreases as 2^{-13j} and determines the gap. The derivative amplitude determines the precision needed to distinguish a child: event j requires $p_j = \Omega(j)$. Serial consumption of complete replies then gives

$$n_j \geq \sum_{i<j} \Omega(p_i) = \Omega(j^2), \qquad F(z_{n_j}) - F^* \geq c\,2^{-13j} \geq c\exp(-C\sqrt{n_j}).$$

The nested-family idea is used in fixed-function lower bounds (Sikorski & Trojan, 1990); the lemmas below establish the convexity, smoothness, objective gaps, and exact nearest-rounding compatibility needed here. If refinement stops, four indistinguishable fixed completions instead give a positive gap infinitely often.

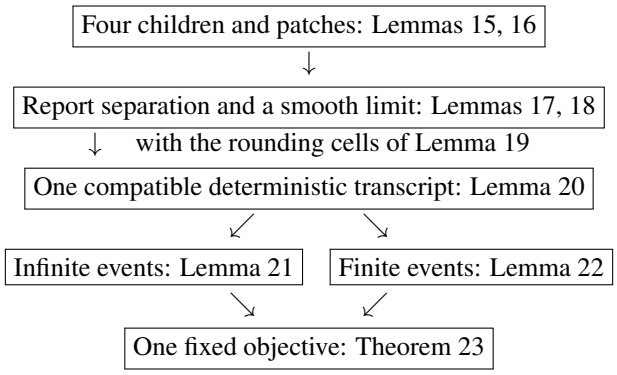

Figure 4: Dependencies in the lower-bound proof. Both event cases use the same fixed nearest-grid rule. The final case selection fixes a single objective independently of any terminal horizon.

C.2 Scales, Endpoint Data, and Four Children

We start in one dimension with anchor zero. A completion means a globally defined convex C^1 function whose derivative is 1-Lipschitz and which agrees with all the values and derivatives already fixed outside the active interval. At depth j the interval and its endpoint data are

$$I_j = [a_j, b_j], \quad \ell_j = b_j - a_j = \tfrac{1}{2}\,64^{-j}, \quad s_j = \tfrac{1}{200}\,128^{-j},$$
$$f(a_j) = f(b_j) = H_j, \quad f'(a_j) = -s_j, \quad f'(b_j) = s_j.$$

Initially $I_0 = [1/4, 3/4]$ and $H_0 = 0$. Outside I_0, fix the derivative to $-s_0$ on the left and s_0 on the right, with endpoint value zero. Thus the exterior is explicitly

$$f(t) = s_0(a_0 - t) \quad (t \le a_0), \qquad f(t) = s_0(t - b_0) \quad (t \ge b_0).$$

In particular the anchor jet is $f(0) = 1/800$, $f'(0) = -1/200$. The following geometric choice is made at every depth.

Lemma 15 (Four separated children)**.** *Let $a \in \mathbb{R}$ and $\ell > 0$. For the parent $I = [a, a + \ell]$, let the four children be*

$$C_1 = [a + 47\ell/128,\ a + 49\ell/128], \qquad C_2 = [a + 173\ell/384,\ a + 179\ell/384],$$
$$C_3 = [a + 205\ell/384,\ a + 211\ell/384], \quad C_4 = [a + 79\ell/128,\ a + 81\ell/128].$$

Every child has length $\ell/64$. The isolation neighborhoods $N(C) = [u - \ell/100, v + \ell/100]$ for $C = [u, v]$ are pairwise disjoint and contained in I. The margin $\ell/100$ on each side uses the parent length ℓ. For any two real numbers x, z, at least two children satisfy $x, z \notin N(C)$.

Proof. For $r \in \{1, 2, 3, 4\}$, write $C_r = a + \ell[c_r - 1/128, c_r + 1/128]$, where

$$(c_1, c_2, c_3, c_4) = (3/8, 11/24, 13/24, 5/8).$$

For example, $11/24 \pm 1/128 = (176 \pm 3)/384$, which gives C_2; the same conversion gives the other displayed endpoints. Every width is therefore $2\ell/128 = \ell/64$.

The actual child center is $a + \ell c_r$. The normalized neighborhood half-width is $h_N = 1/128 + 1/100 = 57/3200$, so

$$N(C_r) = [a + \ell(c_r - h_N),\ a + \ell(c_r + h_N)].$$

All successive center differences equal $1/12$, so the gap between two consecutive closed neighborhoods is

$$\frac{1}{12} - 2h_N = \frac{1}{12} - \frac{57}{1600} = \frac{229}{4800} > 0.$$

Nonconsecutive neighborhoods are even farther apart. The distance from the leftmost neighborhood to the left parent endpoint equals the corresponding rightmost distance:

$$c_1 - h_N = 1 - c_4 - h_N = \frac{3}{8} - \frac{57}{3200} = \frac{1143}{3200} > 0.$$

Multiplying these positive gaps by $\ell > 0$ proves disjointness and containment in I, including neighborhood endpoints. A real number can therefore belong to at most one $N(C_r)$, including when it is a neighborhood endpoint. The points x and z together exclude at most two indices, leaving at least $4 - 2 = 2$ eligible children. □

C.3 Constructing One Convex Patch

To refine a parent, we must define the function on the whole parent, not just specify a new interval. The construction below fixes the exterior of the selected child and leaves only its interior unresolved. It also explains why asymmetrically placed children can have the same endpoint levels: the lengths of the constant derivative pieces are chosen to make the two exterior integrals prescribed quantities.

Lemma 16 (A patch with matching endpoint data)**.** *Let $a < b$, $\ell = b - a > 0$, $H \in \mathbb{R}$, and $s > 0$. Prescribe endpoint values and derivatives $H, H, -s, s$ on $I = [a, b]$. For any child $C = [u, v]$ from Lemma 15, there is a continuous nondecreasing function $g_C : I \to \mathbb{R}$ such that, with $s' = s/128$,*

$$g_C(a) = -s, \quad g_C(u) = -s', \quad g_C(v) = s', \quad g_C(b) = s,$$
$$\int_a^u g_C = -s\ell/4, \qquad \int_u^v g_C = 0, \qquad \int_v^b g_C = s\ell/4.$$

The derivative profile g_C satisfies

$$|g_C(x) - g_C(y)| \le \frac{100s}{\ell}|x - y| \qquad (x, y \in I).$$

The function $f_C(t) = H + \int_a^t g_C(r)\, dr$ is convex and continuously differentiable on I, with $f_C' = g_C$ and

$$f_C(a) = f_C(b) = H, \qquad f_C(u) = f_C(v) = H - s\ell/4.$$

Any C^1 function $\widetilde{f}$ on $[u, v]$ with these same child endpoint values and derivatives can replace f_C there while preserving C^1 regularity and leaving f_C, f_C' unchanged on $I \setminus (u, v)$. If $\widetilde{f}$ is also convex and $\widetilde{f}'$ has Lipschitz constant $\widetilde{L} \geq 0$, the joined function is convex on I and its derivative has Lipschitz constant at most $\max\{100s/\ell, \widetilde{L}\}$ on I.

Proof. We specify the lengths, write the derivative pieces, and then check values and regularity.

1. Lengths of the constant pieces. Let $h = \ell/100$ be each outer ramp width. Choose the lengths A_- and A_+ of the left $-s$ and right s constant pieces as

$$A_- = \frac{s\ell/4 - s'(u-a)}{s - s'} - \frac{h}{2}, \qquad A_+ = \frac{s\ell/4 - s'(b-v)}{s - s'} - \frac{h}{2}.$$

The denominator is positive because $s - s' = 127s/128 > 0$. For the normalized exterior lengths $t_- = (u-a)/\ell$ and $t_+ = (b-v)/\ell$, the child endpoints give

$$t_-, t_+ \in \{47/128, 173/384, 205/384, 79/128\} \subset [47/128, 79/128] \subset (0, 1).$$

Substituting $s' = s/128$ and $h = \ell/100$ gives

$$\frac{A_\pm}{\ell} = \frac{1/4 - t_\pm/128}{127/128} - \frac{1}{200} = \frac{32 - t_\pm}{127} - \frac{1}{200}.$$

Since $0 < t_\pm < 1$, the required uniform bounds follow from

$$\left(\frac{31}{127} - \frac{1}{200}\right) - \frac{1}{5} = \frac{993}{25400} > 0,$$
$$\frac{1}{4} - \left(\frac{32}{127} - \frac{1}{200}\right) = \frac{77}{25400} > 0.$$

Thus $\ell/5 < A_\pm < \ell/4$. The remaining constant segment on either exterior has positive length because

$$t_\pm \ell - A_\pm - h > \ell\left(\frac{47}{128} - \frac{1}{4} - \frac{1}{100}\right) = \frac{343\ell}{3200} > 0.$$

All three pieces on each side therefore fit without overlap or reversed endpoints.

2. The derivative on all of the parent. On the two exteriors define

$$g_C(t) = \begin{cases} -s, & a \leq t \leq a + A_-, \\ -s + (s - s')(t - a - A_-)/h, & a + A_- \leq t \leq a + A_- + h, \\ -s', & a + A_- + h \leq t \leq u, \end{cases}$$

$$g_C(t) = \begin{cases} s', & v \leq t \leq b - A_+ - h, \\ s' + (s - s')(t - b + A_+ + h)/h, & b - A_+ - h \leq t \leq b - A_+, \\ s, & b - A_+ \leq t \leq b. \end{cases}$$

Use the following temporary linear derivative inside the child:

$$g_C(t) = -s' + \frac{2s'}{v - u}(t - u) \qquad (u \leq t \leq v).$$

Every ramp takes the stated adjacent constant values at its two endpoints. The child ramp takes $-s'$ at u and s' at v. The pieces therefore meet continuously, in the nondecreasing order $-s \leq -s' \leq s' \leq s$. Only the exteriors are frozen when the child is selected; the child formula supplies a local interpolation on I and can later be replaced.

3. Integral and endpoint identities. A linear ramp has integral equal to its length times the average of its two endpoint values. Applying this to each side gives

$$\begin{aligned}\int_a^u g_C(t)\,dt &= -sA_- - \frac{s+s'}{2}h - s'(u-a-A_- - h)\\ &= -(s-s')(A_- + h/2) - s'(u-a)\\ &= -[s\ell/4 - s'(u-a)] - s'(u-a) = -s\ell/4,\\ \int_v^b g_C(t)\,dt &= s'(b-v-A_+ - h) + \frac{s+s'}{2}h + sA_+\\ &= (s-s')(A_+ + h/2) + s'(b-v)\\ &= [s\ell/4 - s'(b-v)] + s'(b-v) = s\ell/4.\end{aligned}$$

Inside the child,

$$\int_u^v \left[-s' + \frac{2s'}{v-u}(t-u)\right] dt = -s'(v-u) + \frac{s'}{v-u}(v-u)^2 = 0.$$

Consequently, for $f_C(t) = H + \int_a^t g_C(r)\,dr$,

$$\begin{aligned}&f_C(a) = H, && f_C(u) = H - s\ell/4,\\ &f_C(v) = f_C(u) + 0, && f_C(b) = H - s\ell/4 + 0 + s\ell/4 = H.\end{aligned}$$

4. Lipschitz continuity and convexity. On an outer ramp the slope of g_C is $(s-s')/h = 100(s-s')/\ell \le 100s/\ell$. On the child ramp it is

$$\frac{2s'}{v-u} = \frac{2(s/128)}{\ell/64} = \frac{s}{\ell},$$

and it is zero on each constant piece. For $a \le x < y \le b$, list the breakpoints in $[x,y]$ as $x = t_0 < t_1 < \cdots < t_M = y$. Continuity at these breakpoints gives

$$\begin{aligned}|g_C(y) - g_C(x)| &\le \sum_{r=1}^{M} |g_C(t_r) - g_C(t_{r-1})|\\ &\le \frac{100s}{\ell}\sum_{r=1}^{M}(t_r - t_{r-1}) = \frac{100s}{\ell}(y-x).\end{aligned}$$

The fundamental theorem of calculus gives $f_C' = g_C$ and $f_C \in C^1$. For $x, y \in I$ with $x < y$, monotonicity gives $\int_x^y g_C(t)dt \ge g_C(x)(y-x)$; for $y < x$, it gives $-\int_y^x g_C(t)dt \ge -g_C(x)(x-y)$. Thus $f_C(y) \ge f_C(x) + g_C(x)(y-x)$ for either ordering. To obtain convexity from this inequality, let $z = (1-t)x + ty$ with $0 \le t \le 1$ and apply it with base point z:

$$f_C(x) \ge f_C(z) + g_C(z)(x-z), \qquad f_C(y) \ge f_C(z) + g_C(z)(y-z).$$

Multiplying by $1-t$ and t, respectively, and adding gives

$$\begin{aligned}(1-t)f_C(x) + tf_C(y) &\ge f_C(z) + g_C(z)[(1-t)(x-z) + t(y-z)]\\ &= f_C(z),\end{aligned}$$

because $(1-t)x + ty = z$. This is the definition of convexity.

5. Replacing the unresolved interior. Let $\widetilde{f}$ be a C^1 function on $[u,v]$ with

$$\widetilde{f}(u) = \widetilde{f}(v) = H - s\ell/4, \qquad \widetilde{f}'(u) = -s', \quad \widetilde{f}'(v) = s'.$$

Define the joined function on I by

$$\widehat{f}(t) = \begin{cases} f_C(t), & t \in [a,u] \cup [v,b],\\ \widetilde{f}(t), & t \in [u,v].\end{cases}$$

The two formulas agree at u, v, and the matching derivatives make $\widehat{f} \in C^1(I)$. Moreover,

$$\int_u^v \widetilde{f}'(t)dt = \widetilde{f}(v) - \widetilde{f}(u) = 0 = \int_u^v g_C(t)dt.$$

Thus even if the right values are defined by integration from a, the replacement makes no change to them. The left values and both exterior derivatives are unchanged by definition.

Now assume additionally that $\widetilde{f}$ is convex and $\widetilde{f}'$ is $\widetilde{L}$-Lipschitz. For $u \le x < y \le v$, adding the two first-order convexity inequalities (3) with the base points exchanged gives

$$[\widetilde{f}'(y) - \widetilde{f}'(x)](y - x) \ge 0.$$

Hence $\widetilde{f}'$ is nondecreasing and lies in $[-s', s']$. The exterior derivatives increase from $-s$ to $-s'$ and from s' to s, so $\widehat{f}'$ is nondecreasing on all of I. The argument in Step 4 then proves convexity of $\widehat{f}$. For any $a \le x < y \le b$, partition $[x, y]$ at any of u, v lying in its interior, writing $x = t_0 < \cdots < t_M = y$. Each subinterval lies in one region, where the derivative has Lipschitz constant at most $100s/\ell$ or $\widetilde{L}$. Continuity at the joins gives

$$\begin{aligned} |\widehat{f}'(y) - \widehat{f}'(x)| &\le \sum_{i=1}^{M} |\widehat{f}'(t_i) - \widehat{f}'(t_{i-1})| \\ &\le \max\{100s/\ell, \widetilde{L}\} \sum_{i=1}^{M} (t_i - t_{i-1}) \\ &= \max\{100s/\ell, \widetilde{L}\}(y - x). \end{aligned}$$

This proves the asserted bound on I after replacement.

□

Applying Lemma 16 repeatedly gives the recurrences

$$\ell_{j+1} = \ell_j/64, \quad s_{j+1} = s_j/128, \quad H_{j+1} = H_j - s_j\ell_j/4. \tag{18}$$

Write $q_j = s_j\ell_j = 2^{-13j}/400$ and $r = 2^{-13}$. These symbols refer to the lower construction only. The local bound in Lemma 16 decreases under this recursion:

$$\frac{100s_{j+1}}{\ell_{j+1}} = \frac{100(s_j/128)}{\ell_j/64} = \frac{1}{2}\frac{100s_j}{\ell_j}, \qquad \frac{100s_0}{\ell_0} = 1.$$

Thus $100s_j/\ell_j = 2^{-j}$, and every outer ramp created at depth j has Lipschitz constant at most $2^{-j} \le 1$. The temporary completion at any active depth has derivative slope $2s_j/\ell_j \le 1/50$. The join estimate above takes the maximum of the regional constants. Hence every finite completed construction has a globally 1-Lipschitz derivative, including across the joins and the constant outer tails. These recursive scales therefore produce completions in the global sense defined in Appendix C.2.

C.4 A Gap Away from the Selected Child

Lemma 17 (Report separation)**.** *Let $j \in \mathbb{N}_0$ and select child $C = [u, v]$ in the depth-j parent of* (18) *using Lemma 16. Let $f : \mathbb{R} \to \mathbb{R}$ be any convex C^1 completion that preserves the resulting frozen exterior and the prescribed child endpoint values and derivatives. Write*

$$N(C) = [u - \ell_j/100,\ v + \ell_j/100].$$

The function attains a finite minimum $f^* = \min_{x\in\mathbb{R}} f(x)$, every minimizer belongs to C, and, for every $t \in \mathbb{R}$,

$$t \notin N(C) \quad \Longrightarrow \quad f(t) - f^* \ge \frac{s_j\ell_j}{12800} = \frac{2^{-13j}}{5{,}120{,}000}.$$

Proof. Write $s' = s_j/128 > 0$ and $h = \ell_j/100 > 0$. The endpoint data from Lemma 16 give $f'(u) = -s'$ and $f'(v) = s'$.

1. Global monotonicity and existence of a minimizer. For $x, y \in \mathbb{R}$ and $0 < \lambda \le 1$, convexity gives

$$f(x + \lambda(y - x)) \le (1 - \lambda)f(x) + \lambda f(y), \qquad \frac{f(x + \lambda(y - x)) - f(x)}{\lambda} \le f(y) - f(x).$$

Letting $\lambda \downarrow 0$ and then exchanging x, y yields

$$f(y) \ge f(x) + f'(x)(y - x), \qquad f(x) \ge f(y) + f'(y)(x - y).$$

Adding and rearranging for $x < y$ yields

$$[f'(y) - f'(x)](y - x) \ge 0, \qquad f'(y) \ge f'(x).$$

Thus f' is nondecreasing, and the preserved endpoint slopes imply

$$t \le u \implies f'(t) \le f'(u) = -s' < 0, \qquad t \ge v \implies f'(t) \ge f'(v) = s' > 0.$$

Continuity and $f'(u) < 0 < f'(v)$ give an $\alpha \in (u, v)$ with $f'(\alpha) = 0$. For $t \ge \alpha$, the derivative is nonnegative on $[\alpha, t]$, and for $t \le \alpha$ it is nonpositive on $[t, \alpha]$. Hence

$$f(t) - f(\alpha) = \begin{cases} \displaystyle\int_\alpha^t f'(r)\, dr \ge 0, & t \ge \alpha, \\ \displaystyle -\int_t^\alpha f'(r)\, dr \ge 0, & t \le \alpha. \end{cases}$$

Thus $f^* = f(\alpha)$ is attained. A differentiable function has zero derivative at any minimizer on $\mathbb{R}$, so the strict exterior signs above place every minimizer in $(u, v) \subset C$.

2. The two isolation bands. For $t \ge v + h$, the points satisfy $\alpha < v < v + h \le t$. Monotonicity gives $f' \ge 0$ on $[\alpha, v]$ and $f' \ge s'$ on $[v, t]$. Thus all three integrals below are nonnegative, and the middle one spans the full right isolation band:

$$\begin{aligned} f(t) - f^* &= \int_\alpha^v f'(r)dr + \int_v^{v+h} f'(r)dr + \int_{v+h}^t f'(r)dr \\ &\ge \int_v^{v+h} s'\, dr = s'h. \end{aligned}$$

For $t \le u - h$, we have $t \le u - h < u < \alpha$, $-f' \ge s'$ on $[t, u]$, and $-f' \ge 0$ on $[u, \alpha]$. The corresponding left calculation is

$$\begin{aligned} f(t) - f^* &= \int_t^{u-h} [-f'(r)]dr + \int_{u-h}^u [-f'(r)]dr + \int_u^\alpha [-f'(r)]dr \\ &\ge \int_{u-h}^u s'\, dr = s'h. \end{aligned}$$

The complement of the closed interval $N(C) = [u - h, v + h]$ is $(-\infty, u - h) \cup (v + h, \infty)$, so these two estimates cover every $t \notin N(C)$; they also hold at the two neighborhood endpoints. The derivative inequalities were proved on all of $\mathbb{R}$, so the argument includes points outside the parent interval.

3. The depth-dependent gap. The scales give

$$s_j \ell_j = \frac{1}{200} \frac{1}{2}\, 128^{-j} 64^{-j} = \frac{2^{-13j}}{400}.$$

Therefore

$$s'h = \frac{s_j}{128} \frac{\ell_j}{100} = \frac{s_j \ell_j}{12800} = \frac{2^{-13j}}{400 \cdot 12800} = \frac{2^{-13j}}{5{,}120{,}000}.$$

□

C.5 Existence of One Fixed Smooth Function

Lemma 18 (One fixed smooth realization)**.** *Start from $I_0 = [a_0, b_0] = [1/4, 3/4]$, $H_0 = 0$, and $s_0 = 1/200$, use the scales in* (18)*, and install the exterior patch of Lemma 16 at each selected child. Every infinite sequence of such nested choices has a globally* 1*-Lipschitz, continuous, nondecreasing limiting derivative $g : \mathbb{R} \to \mathbb{R}$. Its integral with $f(a_0) = 0$ is a convex C^1 function on $\mathbb{R}$, agrees with every frozen exterior, and has a unique minimizer α, where $\{\alpha\} = \bigcap_{j\geq 0} I_j$. The common endpoint levels and minimum are*

$$H_j = -\frac{1 - r^j}{1600(1 - r)}, \qquad f^* = H_\infty = -\frac{1}{1600(1 - r)} = -\frac{128}{204775}, \qquad r = 2^{-13}.$$

A finite sequence can also be completed, using a linear derivative on the final unresolved interval, with the same convexity and Lipschitz properties.

Proof. Fix any infinite sequence of nested choices. We first construct the limiting derivative, then its integral and minimizer. For a function $h : \mathbb{R} \to \mathbb{R}$, write

$$\|h\|_\infty = \sup_{x\in\mathbb{R}} |h(x)|.$$

The bounds below show that this supremum is finite for the functions and differences to which the norm is applied.

1. Finite completions and a uniform derivative limit. For each $j \in \mathbb{N}_0$, let $g^{(j)}$ equal the frozen derivative on $\mathbb{R} \setminus (a_j, b_j)$ and set

$$g^{(j)}(t) = -s_j + \frac{2s_j}{\ell_j}(t - a_j) \qquad (t \in I_j).$$

At every join its values agree with the adjacent frozen pieces. Each piece is nondecreasing, and the endpoint values occur in nondecreasing order, so $g^{(j)}$ is continuous and nondecreasing. For every earlier depth $i < j$, the outer ramp slopes satisfy

$$\frac{100(s_i - s_i/128)}{\ell_i} \leq \frac{100s_i}{\ell_i} = 2^{-i} \leq 1.$$

The active ramp has slope $2s_j/\ell_j = 2^{-j}/50 \leq 1$, and the tails have slope zero. Subdividing any finite segment $[x, y]$ at its finitely many joins and adding the piecewise bounds, as in Lemma 16, yields

$$|g^{(j)}(y) - g^{(j)}(x)| \leq |y - x| \qquad (x, y \in \mathbb{R}).$$

This explains the global Lipschitz bound, including across joins and between a tail and an interior point.

For $k \geq j$, all later changes lie inside (a_j, b_j), so $g^{(k)} = g^{(j)}$ on $\mathbb{R} \setminus (a_j, b_j)$. In particular, endpoint preservation in Lemma 16 gives

$$g^{(k)}(a_j) = g^{(j)}(a_j) = -s_j, \qquad g^{(k)}(b_j) = g^{(j)}(b_j) = s_j.$$

Monotonicity bounds both derivatives between $-s_j$ and s_j throughout I_j. Therefore

$$\|g^{(k)} - g^{(j)}\|_\infty \leq 2s_j = \frac{1}{100}128^{-j} \longrightarrow 0.$$

For each x, the last estimate makes $(g^{(j)}(x))_j$ a Cauchy sequence, so completeness of $\mathbb{R}$ gives a limit $g(x) = \lim_j g^{(j)}(x)$. First take $k \to \infty$ for fixed x, j:

$$|g(x) - g^{(j)}(x)| = \lim_{k\to\infty} |g^{(k)}(x) - g^{(j)}(x)| \leq 2s_j.$$

Taking the supremum over x now gives $\|g - g^{(j)}\|_\infty \leq 2s_j \to 0$, proving uniform convergence on the whole line. The frozen endpoint data also pass to the limit:

$$g(a_j) = -s_j, \qquad g(b_j) = s_j, \qquad |g(x)| \leq s_j \quad (x \in I_j).$$

For any $x < y$,

$$g(x) = \lim_j g^{(j)}(x) \le \lim_j g^{(j)}(y) = g(y),$$

and for any x, y,

$$|g(x) - g(y)| = \lim_j |g^{(j)}(x) - g^{(j)}(y)| \le |x - y|.$$

Thus g is nondecreasing and 1-Lipschitz, hence continuous.

2. The limiting function and the frozen data. Define

$$f(x) = \int_{a_0}^{x} g(t)\,dt, \qquad f^{(j)}(x) = \int_{a_0}^{x} g^{(j)}(t)\,dt.$$

These integrals exist on every finite segment. The fundamental theorem of calculus gives $f' = g$, $f \in C^1$, and $f(a_0) = 0$. Monotonicity of g gives, for either ordering of x, y,

$$f(y) - f(x) \ge g(x)(y - x).$$

For completeness, if $w = (1-t)x + ty$ and $0 \le t \le 1$, apply this inequality with base point w to x and y and add with weights $1 - t, t$. Since $(1-t)(x-w) + t(y-w) = 0$, the result is

$$(1-t)f(x) + tf(y) \ge f(w),$$

which proves convexity.

To verify the frozen function values, consider one refinement from depth i to $i+1$. The old linear fill and the new three-part patch have the same total integral on the parent:

$$\int_{a_i}^{b_i} g^{(i)}(t)\,dt = -s_i\ell_i + \frac{2s_i}{\ell_i}\frac{\ell_i^2}{2} = 0,$$

$$\int_{a_i}^{b_i} g^{(i+1)}(t)\,dt = -\frac{s_i\ell_i}{4} + 0 + \frac{s_i\ell_i}{4} = 0.$$

The second equality uses the left exterior, child, and right exterior integrals from Lemma 16. The difference $g^{(i+1)} - g^{(i)}$ vanishes outside (a_i, b_i) and integrates to zero over that interval. Since $a_0 \le a_i$, it follows that

$$f^{(i+1)}(x) - f^{(i)}(x) = \int_{a_0}^{x} [g^{(i+1)}(t) - g^{(i)}(t)]\,dt = 0 \quad (x \le a_i \text{ or } x \ge b_i).$$

This proves preservation on both exterior components, including their endpoints. At depth zero, $f^{(0)}(a_0) = f^{(0)}(b_0) = 0 = H_0$. If the depth-$i$ endpoint values are H_i, the patch identities give

$$f^{(i+1)}(a_{i+1}) = H_i + \int_{a_i}^{a_{i+1}} g^{(i+1)}(t)\,dt = H_i - \frac{s_i\ell_i}{4} = H_{i+1},$$

$$f^{(i+1)}(b_{i+1}) = f^{(i+1)}(a_{i+1}) + \int_{a_{i+1}}^{b_{i+1}} g^{(i+1)}(t)\,dt = H_{i+1}.$$

Induction therefore gives, for every $k \ge j$,

$$f^{(k)}(a_j) = f^{(j)}(a_j) = H_j, \qquad f^{(k)}(b_j) = f^{(j)}(b_j) = H_j,$$

and $f^{(k)} = f^{(j)}$ on $\mathbb{R} \setminus (a_j, b_j)$. For $x \in I_j$,

$$\begin{aligned} |f^{(k)}(x) - f^{(j)}(x)| &= \left| \int_{a_j}^{x} [g^{(k)}(t) - g^{(j)}(t)]dt \right| \\ &\le (x - a_j)\|g^{(k)} - g^{(j)}\|_\infty \le 2s_j\ell_j. \end{aligned}$$

Also, for every fixed x, uniform convergence of the derivatives gives

$$|f^{(k)}(x) - f(x)| \le |x - a_0|\,\|g^{(k)} - g\|_\infty \longrightarrow 0.$$

Letting $k \to \infty$ gives $f = f^{(j)}$ on $\mathbb{R} \setminus (a_j, b_j)$ and $f(a_j) = f(b_j) = H_j$. Together with the derivative preservation in Step 1, this proves that both parts of every frozen jet are retained. The interior bound and the zero difference outside yield

$$\|f - f^{(j)}\|_\infty = \sup_{x \in I_j} |f(x) - f^{(j)}(x)| \le 2 s_j \ell_j.$$

3. The unique minimizer. The endpoints satisfy $a_j \le a_{j+1} \le b_{j+1} \le b_j$ and remain in the bounded interval I_0. Thus they have limits, and

$$\lim_j b_j - \lim_j a_j = \lim_j \ell_j = 0.$$

Write their common limit as α; then $\bigcap_j I_j = \{\alpha\}$. If $x < \alpha$, convergence $a_j \uparrow \alpha$ gives some j with $x < a_j$, whence $g(x) \le g(a_j) = -s_j < 0$. If $x > \alpha$, choose j with $x > b_j$ to obtain $g(x) \ge g(b_j) = s_j > 0$. Since $\alpha \in I_j$ and $|g| \le s_j$ there, $|g(\alpha)| \le s_j$ for every j, so $g(\alpha) = 0$. Consequently

$$f(x) - f(\alpha) = \begin{cases} -\displaystyle\int_x^\alpha g(t)dt > 0, & x < \alpha, \\ \displaystyle\int_\alpha^x g(t)dt > 0, & x > \alpha. \end{cases}$$

The integrals are strict because g has the indicated strict sign at every interior point of the corresponding nonempty segment. Thus α is the unique minimizer.

4. Endpoint levels and the minimum. The scale product is $q_j = s_j \ell_j = (1/400) r^j$ with $r = 2^{-13}$. Starting from $H_0 = 0$ and summing $H_{i+1} - H_i = -q_i/4$ gives

$$H_j = -\frac{1}{1600} \sum_{i=0}^{j-1} r^i = -\frac{1 - r^j}{1600(1 - r)}.$$

The formula also holds for $j = 0$, using an empty sum. For $x \in I_j$, put $\theta = (x - a_j)/\ell_j \in [0, 1]$. The endpoint values established in Step 2 and convexity give

$$f(x) \le (1 - \theta) f(a_j) + \theta f(b_j) = H_j.$$

The derivative bound in Step 1 gives the other side:

$$f(x) = H_j + \int_{a_j}^x g(t)dt \ge H_j - s_j(x - a_j) \ge H_j - q_j.$$

In particular $H_j - q_j \le f(\alpha) \le H_j$. Because $q_j \to 0$ and $0 < r < 1$, both bounds converge to the finite limit $H_\infty := \lim_{j \to \infty} H_j$, namely

$$H_\infty = -\frac{1}{1600(1 - 2^{-13})} = -\frac{8192}{1600 \cdot 8191} = -\frac{128}{204775}.$$

This proves $f^* = f(\alpha) = H_\infty$.

5. A finite nested sequence. Given any finite sequence ending at depth $J \in \mathbb{N}_0$, the definitions of $g^{(J)}$ and $f^{(J)}$ in Steps 1–2 use only those J choices. They therefore provide a complete convex C^1 function with 1-Lipschitz derivative and all frozen data. Its active linear derivative vanishes at $x_J = (a_J + b_J)/2$ and has opposite strict signs on either side within I_J. Outside I_J, monotonicity and the endpoint values give $g^{(J)} \le -s_J < 0$ on the left and $g^{(J)} \ge s_J > 0$ on the right. Thus x_J is its unique minimizer, and its value is

$$\begin{aligned} \min_{x \in \mathbb{R}} f^{(J)}(x) &= H_J + \int_{a_J}^{a_J + \ell_J/2} \left[-s_J + \frac{2 s_J}{\ell_J}(t - a_J) \right] dt \\ &= H_J - \frac{s_J \ell_J}{2} + \frac{s_J \ell_J}{4} = H_J - \frac{s_J \ell_J}{4}. \end{aligned}$$

□

C.6 Why Coarse Replies Are Exactly Identical

The geometric construction alone does not justify reusing an old finite-precision answer. With a fixed rounding rule, a later change could move a true value across a rounding midpoint even when that change is small. We therefore prove that all remaining values lie strictly inside one and the same rounding cell. For spacing η_p, a nearest-rounding cell has endpoints $(r-1/2)\eta_p$ and $(r+1/2)\eta_p$ for an integer r (Brent & Zimmermann, 2010, Section 3.1.9). The strict inequalities below also make the tie convention irrelevant for coarse active queries.

Lemma 19 (A common nearest-rounding cell)**.** *Fix $k, p \in \mathbb{N}_0$ and let $I_k = [a_k, b_k]$ be the active interval at depth k. Its length $\ell_k = b_k - a_k$ and the endpoint parameters s_k, H_k follow* (18)*, starting from $\ell_0 = 1/2$, $s_0 = 1/200$, and $H_0 = 0$. Let $\eta_p = 2^{-8p}$ satisfy $\eta_p \geq 2s_k$. Use the fixed nearest rounding of Appendix A.4 and put $\widehat{H}_k = \mathrm{RN}_p(H_k)$. Then*

$$[H_k - s_k\ell_k, H_k] \subset (\widehat{H}_k - \eta_p/2, \widehat{H}_k + \eta_p/2), \qquad s_k < \eta_p/2. \tag{19}$$

For every convex differentiable completion $h : \mathbb{R} \to \mathbb{R}$ with $h(a_k) = h(b_k) = H_k$, $h'(a_k) = -s_k$, and $h'(b_k) = s_k$, and every $x \in I_k$, this implies

$$\begin{aligned} |\widehat{H}_k - h(x)| &< \eta_p/2, \qquad |h'(x)| < \eta_p/2, \\ \mathrm{RN}_p(h(x)) &= \widehat{H}_k, \qquad \mathrm{RN}_p(h'(x)) = 0. \end{aligned} \tag{20}$$

Proof. Write $q_k = s_k\ell_k = 2^{-13k}/400$ and $r_0 = 2^{-13}$. We first enclose all possible values, and then check that this enclosure crosses no rounding midpoint.

1. One interval for every completion. For $x \in I_k$, let $t = (x - a_k)/\ell_k \in [0, 1]$. Convexity and the prescribed endpoint values give

$$h(x) \leq (1-t)h(a_k) + th(b_k) = H_k.$$

The first-order convexity inequality at a_k gives the lower bound

$$h(x) \geq h(a_k) + h'(a_k)(x - a_k) = H_k - s_k(x - a_k) \geq H_k - q_k.$$

Monotonicity of the derivative similarly gives $-s_k = h'(a_k) \leq h'(x) \leq h'(b_k) = s_k$. These statements include $x = a_k, b_k$. The first-order inequality and derivative monotonicity follow from convexity as derived in Step 1 of the proof of Lemma 17.

By the geometric sum in Lemma 18,

$$H_\infty = -\frac{1}{1600(1-2^{-13})} = -\frac{128}{204775}, \qquad H_k - H_\infty = \frac{q_k}{4(1-r_0)} < q_k. \tag{21}$$

Indeed, $H_k - H_\infty > 0$ and $4(1-r_0) > 1$. Here H_∞ serves as a reference value determined by the prescribed endpoint levels; the argument also applies to completions obtained by stopping at a finite depth. Thus both the possible value $h(x)$ and the fixed reference H_∞ belong to $J_k = [H_k - q_k, H_k]$. Since this interval has length q_k,

$$\sup_{y \in J_k} |y - H_\infty| \leq q_k.$$

The bounds use only depth-k endpoint data, so they continue to hold after every later refinement.

2. Distance from the reference to a midpoint. For a real number t and a nonempty set $S \subseteq \mathbb{R}$, write $\mathrm{dist}(t, S) = \inf_{u \in S} |t - u|$. Let $D = 204775$ and let $r \in \mathbb{Z}$ index a rounding midpoint. Because $H_\infty/\eta_p = -128\,2^{8p}/D$,

$$\begin{aligned} \left| \frac{H_\infty}{\eta_p} - (r + \tfrac{1}{2}) \right| &= \frac{|-256\,2^{8p} - D(2r+1)|}{2D} \\ &\geq \frac{1}{2D}. \end{aligned}$$

Here $256\,2^{8p}$ is an even integer, whereas $D(2r+1)$ is odd. Their difference is therefore a nonzero integer of magnitude at least one. Multiplying by η_p and taking the infimum over r gives

$$\mathrm{dist}\big(H_\infty, \eta_p(\mathbb{Z} + \tfrac{1}{2})\big) \geq \frac{\eta_p}{409550}. \tag{22}$$

For $k \geq 3$, the assumption $\eta_p \geq 2s_k$ gives

$$\frac{q_k}{\eta_p} = \frac{s_k \ell_k}{\eta_p} \leq \frac{\ell_k}{2} = 2^{-6k-2} \leq 2^{-20} < \frac{1}{409550},$$

where the last strict inequality uses $409550 < 2^{20} = 1048576$. Hence every $y \in J_k$ satisfies

$$|y - H_\infty| \leq q_k < \operatorname{dist}\big(H_\infty, \eta_p(\mathbb{Z} + \tfrac{1}{2})\big).$$

Moving from H_∞ to any such y cannot reach a midpoint. Thus the entire closed interval J_k is strictly inside one open rounding cell. Its endpoint H_k lies in that cell, so its center is $\mathrm{RN}_p(H_k) = \widehat{H}_k$.

3. The initial depths. The coarse condition is equivalent to

$$2^{-8p} \geq 2^{-7k}/100 \quad \Longleftrightarrow \quad 8p \leq 7k + \log_2 100.$$

Since $6 < \log_2 100 < 7$ and p is a nonnegative integer, the allowed precisions at $k = 0, 1, 2$ are respectively $\{0\}$, $\{0, 1\}$, and $\{0, 1, 2\}$. These are exactly the six rows of Table 17. For its first five rows the lower endpoint has magnitude at most one of

$$\frac{1}{400} < \frac{1}{4}, \qquad \frac{2049}{12800} < \frac{1}{4}, \qquad \frac{3355853}{20971520} < \frac{1}{4};$$

the last two comparisons follow from $4 \cdot 2049 = 8196 < 12800$ and $4 \cdot 3355853 = 13423412 < 20971520$. Both endpoints are nonpositive, so each of these five intervals lies in $(-1/4, 1/4) \subset (-1/2, 1/2)$ and rounds to zero. For the sixth row,

$$-\frac{3355853}{81920} - \left(-\frac{83}{2}\right) = \frac{43827}{81920} > 0,$$
$$-\frac{8193}{200} - \left(-\frac{3355853}{81920}\right) = \frac{1}{409600} > 0,$$
$$-\frac{81}{2} - \left(-\frac{8193}{200}\right) = \frac{93}{200} > 0.$$

Thus it lies strictly in $(-41 - 1/2, -41 + 1/2)$ and rounds to -41. Rescaling by η_p proves the value-cell inclusion for $k = 0, 1, 2$ as well.

Table 17: Initial-depth rounding cells. The middle columns are the endpoints of $[H_k - q_k, H_k]/\eta_p$; each interval lies strictly within $(r - 1/2, r + 1/2)$ for the integer r in the last column.

k	p	$(H_k - q_k)/\eta_p$	H_k/η_p	r
0	0	$-1/400$	0	0
1	0	$-2049/3276800$	$-1/1600$	0
1	1	$-2049/12800$	$-4/25$	0
2	0	$-3355853/5368709120$	$-8193/13107200$	0
2	1	$-3355853/20971520$	$-8193/51200$	0
2	2	$-3355853/81920$	$-8193/200$	-41

4. Derivative rounding and strict error. Equality $\eta_p = 2s_k$ would imply

$$2^{-8p} = 2^{-7k}/100 \quad \Longrightarrow \quad 100 = 2^{8p-7k}.$$

The exponent is an integer, but $100 = 2^2 \cdot 5^2$ is not an integer power of two. Equality is impossible, so $|h'(x)| \leq s_k < \eta_p/2$. The unique nearest grid point is then zero. The established strict containment of J_k in the cell centered at $\widehat{H}_k$ similarly gives $|h(x) - \widehat{H}_k| < \eta_p/2$ and $\mathrm{RN}_p(h(x)) = \widehat{H}_k$. No tie rule is needed for either field in these cases.

□

C.7 The Adaptive Reply Rule and Transcript Consistency

We now fix a deterministic program and answer its successive physical calls. To use the scalar construction in dimension d, define the projection onto the first coordinate by

$$\pi_1 : \mathbb{R}^d \to \mathbb{R}, \qquad \pi_1(x_1, \dots, x_d) = x_1.$$

For each scalar function f, use the objective $F(x) = f(\pi_1(x))$. It depends only on the first coordinate, so the remaining gradient coordinates are zero. The active scalar interval starts at $I_0 = [1/4, 3/4]$. At depth j, write $I_j = [a_j, b_j]$, where a_j and b_j are its left and right endpoints, and $\ell_j = b_j - a_j$ is its length. Let $\mathcal{F}_j$ denote the set of globally defined convex C^1 functions with 1-Lipschitz derivatives that agree with the frozen function and derivative on $\mathbb{R} \setminus (a_j, b_j)$. In particular, every member has the endpoint data $(H_j, H_j, -s_j, s_j)$; the interior of I_j is its only freedom. The following rule applies whenever a query (x, p) is delivered.

1. If $\pi_1(x) \notin I_j$, use the already fixed exterior value and derivative, followed by the fixed nearest-grid and overflow rule.
2. If $\pi_1(x) \in I_j$ and $\eta_p \geq 2s_j$, return the normal fields $(\mathrm{RN}_p(H_j), 0, \dots, 0)$ and do not refine.
3. If $\pi_1(x) \in I_j$ and $\eta_p < 2s_j$, this is informative event j. Let t_j be its delivery round, $n_j = t_j - 1$, and let z_{n_j} be the preceding complete report. Choose the lowest-indexed child whose neighborhood contains neither $\pi_1(x)$ nor $\pi_1(z_{n_j})$; this is possible by Lemma 15. Install the patch of Lemma 16, freeze its exterior to the child, and return the rounded true jet at $\pi_1(x)$ there. The child becomes I_{j+1}.

At every call, all $d + 1$ fields, the status, and the terminal marker are encoded as in Appendix A.4. This rule depends on the observed query and preceding report, not on a future stopping horizon.

Lemma 20 (Compatibility with a fixed nearest-grid oracle)**.** *Fix an integer $d \geq 1$, the deterministic program, the zero anchor $a = 0 \in \mathbb{R}^d$, and the adaptive rule above. After every finite number of physical calls, all remaining completions in the current family $\mathcal{F}_j$ produce exactly the same replies as the construction on every past query, including framing, signs, overflow, and padding. The first coordinate of every earlier informative query lies outside the current active interval. If refinements continue infinitely, the function of Lemma 18 generates the entire constructed run under its fixed nearest-grid oracle. If refinements stop, every remaining completion generates that same entire run.*

Proof. For $t \in \mathbb{N}_0$, let $j(t)$ count informative events in the first t physical calls and put $\mathcal{C}_t = \mathcal{F}_{j(t)}$. Index calls by $i = 1, 2, \dots$, writing their queries as $(x_i, p_i) \in \mathbb{Q}^d \times \mathbb{N}_0$. For a scalar completion h, define the objective and its gradient by

$$F_h(y) = h(\pi_1(y)), \qquad \nabla F_h(y) = (h'(\pi_1(y)), 0, \dots, 0) \qquad (y \in \mathbb{R}^d).$$

Let reply_i be the complete bit string delivered by the construction, and let $\mathcal{O}_h(x_i, p_i)$ be the fixed nearest-grid reply for F_h, including the status and framing. We prove the following invariant, together with $\mathcal{C}_{t+1} \subseteq \mathcal{C}_t$ at every call:

$$\begin{gathered} \mathcal{C}_t \neq \varnothing, \\ \forall h \in \mathcal{C}_t \; \forall i \in \{1, \dots, t\} : \mathcal{O}_h(x_i, p_i) = \mathsf{reply}_i, \\ \pi_1(x_i) \notin I_{j(t)} \qquad \text{for every informative call } i \leq t. \end{gathered}$$

At $t = 0$ there are no reply equalities to check. The linear derivative on I_0, joined to the two constant tails, gives a convex C^1 completion with derivative Lipschitz constant $2s_0/\ell_0 = 1/50$. Thus $\mathcal{C}_0 \neq \varnothing$. Assume the invariant, set $j = j(t)$, and write the next query as $(x, p) = (x_{t+1}, p_{t+1})$.

1. Exterior query. If $\pi_1(x) \notin I_j$, every member of $\mathcal{C}_t$ has the same frozen value and derivative there. The other $d - 1$ gradient coordinates are all zero. Applying the same rounding, tie, and overflow rules to this identical jet produces the same numerical fields and status. Canonical sign and framing then produce identical bits. The family is unchanged, so the invariant extends to this call.

2. Coarse active query. If $\pi_1(x) \in I_j$ and $\eta_p \geq 2s_j$, the endpoint data of every remaining h meet Lemma 19. Therefore

$$\mathrm{RN}_p(h(\pi_1(x))) = \mathrm{RN}_p(H_j), \qquad \mathrm{RN}_p(h'(\pi_1(x))) = 0.$$

Recall $q_j = s_j \ell_j = 2^{-13j}/400$, $\eta_p = 2^{-8p}$, and $B_p = 2^{8p}$. To check that the fixed oracle selects normal status, use

$$|h(\pi_1(x))| \le |H_j| + q_j \le \frac{1}{1600(1-2^{-13})} + \frac{1}{400} < \frac{1}{100},$$
$$|h'(\pi_1(x))| \le s_j \le \frac{1}{200} < \frac{1}{100}.$$

The first strict bound follows, for example, from $1/(1-2^{-13}) < 2$, which bounds its left side by $1/800 + 1/400 = 3/800 < 1/100$. Since $p \ge 0$, we have $\eta_p \le 1$ and $B_p \ge 1$. For each such field y,

$$|\,\mathrm{RN}_p(y)| \le |y| + \eta_p/2 < 1/100 + 1/2 = 51/100 < 1 \le B_p.$$

Thus all fields are in range and the normal reply is exactly the one prescribed. The family again remains unchanged. This argument also covers queries at a_j or b_j.

3. Informative query. If $\pi_1(x) \in I_j$ and $\eta_p < 2s_j$, the selected child C satisfies $\pi_1(x) \notin N(C)$ and hence $\pi_1(x) \notin C$. Its newly frozen exterior therefore fixes the jet at the current query. All members of the new family $\mathcal{F}_{j+1}$ give the same fixed nearest-grid reply there, including the range test, status, signs, and framing.

The installed patch matches the parent endpoint values and derivatives, and the child interior can be filled by its linear derivative. The slope bounds in Lemma 18's finite-completion argument are at most one at every depth. Hence this filling proves $\mathcal{F}_{j+1} \neq \varnothing$. Every member of $\mathcal{F}_{j+1}$ also belongs to $\mathcal{F}_j$: it is convex and C^1 with 1-Lipschitz derivative, and agrees with the old frozen values and derivatives on $\mathbb{R} \setminus (a_j, b_j)$, including the parent endpoints. In this case,

$$j(t+1) = j+1, \qquad \varnothing \neq \mathcal{C}_{t+1} = \mathcal{F}_{j+1} \subseteq \mathcal{F}_j = \mathcal{C}_t.$$

Thus all earlier reply identities follow from the induction invariant; the newly fixed jet supplies the identity for call $t+1$. Explicitly, the two ways past data remain valid are:

- An old exterior or informative query lies in a previously frozen region; the nested changes inside C cannot alter its true jet.
- An old coarse query at depth $k \le j$ may lie in C. Nevertheless the old endpoint data on I_k remain intact. Applying Lemma 19 with its original precision gives exactly the old rounded value and zero derivative, including at an old endpoint.

The current informative query is outside the new active interval, and all earlier informative queries were outside its parent. They therefore remain outside every subsequent active interval. This proves both the invariant and the stated geometric property, including for repeated queries.

4. Padding replies. Padding causes no additional case: the first coordinate of the zero anchor lies outside $I_0 = [1/4, 3/4]$ and hence outside every active child. Its fixed jet is

$$(1/800, -1/200, 0, \dots, 0) \xrightarrow{p=0} (0, 0, \dots, 0),$$

since both nonzero magnitudes are strictly below $\eta_0/2 = 1/2$. All rounded fields lie in $[-B_0, B_0] = [-1, 1]$. Thus the status is normal, zero has positive sign, and framing is the same for every completion.

5. A single function for the entire transcript. If there are infinitely many refinements, Lemma 18 gives one convex C^1 limit f with 1-Lipschitz derivative, a minimizer, and all frozen exterior data. In particular,

$$f \in \bigcap_{t \ge 0} \mathcal{C}_t, \qquad \mathcal{O}_f(x_i, p_i) = \mathsf{reply}_i \quad \text{for every } i \ge 1.$$

Each equality follows by applying the invariant at any finite $t \ge i$.

If exactly $J \in \mathbb{N}_0$ refinements occur, then $j(t) \le J$ for every t, and nesting gives

$$\mathcal{F}_J \subseteq \mathcal{F}_{j(t)} = \mathcal{C}_t \qquad (t \in \mathbb{N}_0).$$

After the final refinement, $\mathcal{C}_t = \mathcal{F}_J$; subsequent calls use only the exterior, coarse, or padding cases above. Hence every $h \in \mathcal{F}_J$ satisfies $\mathcal{O}_h(x_i, p_i) = \mathsf{reply}_i$ for every $i \geq 1$. This also covers $J = 0$. Such an h attains a global minimum: its continuous derivative has $h'(a_J) = -s_J < 0 < h'(b_J) = s_J$, so some $\alpha \in (a_J, b_J)$ has $h'(\alpha) = 0$, and first-order convexity gives

$$h(y) \geq h(\alpha) + h'(\alpha)(y - \alpha) = h(\alpha) \qquad (y \in \mathbb{R}).$$

6. Equality of the actual program runs. Equality on the constructed query sequence implies equality of actual runs by determinism. Initially the program state and input records agree. If states agree before a call, the next query and its stored precision agree; the preceding equality gives the same delivered bits. Every following charged action reads the same bit from the same state and therefore performs the same transition, write, move, and report selection. Induction over these actions and then over rounds gives identical queries and reports forever. This argument places no bound on the amount of stored state. □

C.8 Informative Events and the Time Lower Bound

The event condition now supplies the precision cost. This is the role played by numerical-prefix charges in communication lower bounds (Tsitsiklis & Luo, 1987, Proposition 3.2). Here we apply it to the single fixed-function construction, with the exact reply length and serial delivery rule.

Lemma 21 (Informative events and report gaps)**.** *In the run generated by the preceding adaptive rule, index informative events by $j = 0, 1, \ldots$ as they occur. Immediately before event j, exactly j refinements have occurred, so the active interval is I_j. Let $p_j \in \mathbb{N}_0$ be its query precision, t_j its delivery round, and $n_j = t_j - 1$ the round of the report available before that delivery. For an integer $d \geq 1$, assume the serial reply protocol and at most $Wd = 4096d$ charged bit actions per round. Then every event satisfies*

$$p_j > \frac{7j + \log_2 100}{8} > \frac{j+1}{2}.$$

If its reply is completely scanned, that scan spans at least $p_j/128$ rounds; if it is never completed, no later useful reply can arrive. If there are infinitely many informative events, let f be the single limit function of Lemma 18, whose fixed oracle realizes the entire run by Lemma 20, and put $F(x) = f(\pi_1(x))$, $f^* = \min_{t \in \mathbb{R}} f(t)$, and $F^* = \min_{x \in \mathbb{R}^d} F(x) = f^*$. Then the preceding report rounds n_j are unbounded and satisfy, for every $j \geq 0$,

$$n_j \geq \frac{j^2}{1024}, \qquad F(z_{n_j}) - F^* \geq \frac{2^{-13j}}{5{,}120{,}000} \geq \frac{1}{5{,}120{,}000} e^{-289\sqrt{n_j}}.$$

Proof. 1. Precision forced by one event. The threshold defining informative event j is

$$2^{-8p_j} < 2s_j = \frac{2^{-7j}}{100}.$$

Taking binary logarithms, which preserve order, gives $-8p_j < -7j - \log_2 100$. Multiplication by -1 reverses the inequality, so

$$p_j > \frac{7j + \log_2 100}{8} > \frac{4j+4}{8} = \frac{j+1}{2},$$

where $j \geq 0$ and $\log_2 100 > 4$ justify the second strict inequality.

2. Mandatory work and disjoint receiver intervals. An informative query cannot be a padding call, since its first coordinate belongs to $I_j \subset [1/4, 3/4]$, whereas the padding anchor has first coordinate zero. Its reply is therefore a useful reply subject to the complete-scan rule. If it finishes, let c_j be the round consuming its terminal marker. All $R(p_j)$ bits must have been read in rounds $t_j, \ldots, c_j$. If it never finishes, no later useful reply can arrive; in particular there can be no next informative event.

Table 18 lists the mandatory read operations, regardless of whether the program uses their numerical information. Here i indexes informative events, and $R(p_i)$ is the full framed reply length from (2).

There are $c_j - t_j + 1$ receiver rounds in that scan, each with at most Wd charged actions. Therefore every completed scan satisfies

$$(c_j - t_j + 1)Wd \geq R(p_j), \qquad c_j - t_j + 1 \geq \left\lceil \frac{R(p_j)}{Wd} \right\rceil.$$

Table 18: Mandatory receiver work for informative reply i. Multiply the middle columns and sum the last column to obtain $R(p_i)$. Only required reads are counted; head moves, writes, query construction, padding processing, and other computation can add further work. The oracle's patch construction and delivery are external and uncharged.

Read step	Repetitions	Reads per repetition	Subtotal
Framed status	1	2	2
Framed scalar field	$d+1$	$32p_i+8$	$(d+1)(32p_i+8)$
Terminal marker	1	2	2
Complete reply			$32(d+1)p_i+8d+12$

Using $d \geq 1$ and the exact reply length,

$$\begin{aligned}\frac{R(p_j)}{Wd} &= \frac{32(d+1)p_j+8d+12}{4096d} \\ &\geq \frac{32(d+1)p_j}{4096d} = \frac{d+1}{d}\frac{p_j}{128} \geq \frac{p_j}{128}.\end{aligned}$$

This is a lower bound on any completed event scan; other processing can only add rounds. Distinct useful reply scans occupy disjoint round intervals because the next useful delivery follows the previous completion round.

For the rest of the proof, assume infinitely many events occur. Then every c_j is finite: otherwise no next event could occur. The serial rule gives

$$t_{j+1} \geq c_j + 1, \qquad t_{j+1} - t_j \geq c_j - t_j + 1 \geq \frac{R(p_j)}{Wd} \geq \frac{p_j}{128}.$$

For $i < j$, completion satisfies $c_i \leq t_{i+1} - 1 \leq t_j - 1 = n_j$. Thus all mandatory reads of replies $0, \ldots, j-1$ have already occurred by the end of round n_j. Let $\mathcal{B}_n$ denote the cumulative charged bit actions through round n, as in Appendix A.5. Summing those mandatory reads gives

$$\mathcal{B}_{n_j} \geq \sum_{i=0}^{j-1} R(p_i) = 32(d+1)\sum_{i=0}^{j-1} p_i + j(8d+12). \tag{23}$$

The sums are empty when $j = 0$. The current reply j is not included: it is delivered in round $t_j = n_j + 1$.

3. Quadratic event clock. Since $t_0 \geq 1$, for $j \geq 1$ telescoping and the previous estimates give

$$\begin{aligned}n_j = t_j - 1 &= (t_0 - 1) + \sum_{i=0}^{j-1}(t_{i+1} - t_i) \\ &\geq \sum_{i=0}^{j-1}\frac{p_i}{128} > \frac{1}{256}\sum_{i=0}^{j-1}(i+1) \\ &= \frac{j(j+1)}{512} \geq \frac{j^2}{1024}.\end{aligned}$$

Here $\sum_{i=0}^{j-1}(i+1) = j(j+1)/2$. For $j = 0$, $n_0 = t_0 - 1 \geq 0$ gives the same final inequality. If events are infinite, $n_j \geq j^2/1024 \to \infty$.

4. The report gap and the exponent direction. At event j, the selected neighborhood avoids $\pi_1(z_{n_j})$. The fixed limit f preserves that selected exterior patch and the child endpoint data by Lemma 18; hence Lemma 17 gives

$$F(z_{n_j}) - F^* = f(\pi_1(z_{n_j})) - f^* \geq \frac{s_j \ell_j}{12800} = \frac{2^{-13j}}{5{,}120{,}000}.$$

The clock estimate bounds the depth from above:

$$n_j \ge j^2/1024 \quad \Longrightarrow \quad j \le 32\sqrt{n_j}.$$

Multiplying by the negative number $-13\log 2$ reverses this inequality. Since the exponential is increasing,

$$2^{-13j} = \exp[-13(\log 2)j] \ge \exp[-416(\log 2)\sqrt{n_j}] \ge \exp[-289\sqrt{n_j}].$$

For the last numerical comparison, the logarithmic series gives

$$\log 2 = 2\sum_{r=0}^{\infty} \frac{1}{(2r+1)3^{2r+1}} < \frac{2}{3} + \frac{2}{3}\sum_{r=1}^{\infty} 3^{-(2r+1)} = \frac{2}{3} + \frac{1}{36} = \frac{25}{36}.$$

The strict inequality uses $2r+1 \ge 3$ for $r \ge 1$, with strictness for $r \ge 2$. Thus $416\log 2 < 416(25/36) = 2600/9 < 289$. Combining this exponential comparison with the gap estimate proves the report bound.

Only the work cap and serial complete scans entered the clock; no storage restriction was used. □

C.9 When Only Finitely Many Events Occur

Lemma 22 (A fixed completion with a persistent gap)**.** *In the infinite run of the preceding deterministic program and adaptive rule, suppose exactly $J \in \mathbb{N}_0$ informative events occur. Then there exist a fixed scalar completion $f \in \mathcal{F}_J$ and a strictly increasing sequence of rounds $m_1 < m_2 < \cdots$, with $m_k \to \infty$, such that, for $F(x) = f(\pi_1(x))$ and $F^* = \min_{x\in\mathbb{R}^d} F(x) = \min_{t\in\mathbb{R}} f(t)$, the corresponding report subsequence $(z_{m_k})_{k\ge1}$ satisfies*

$$F(z_{m_k}) - F^* \ge \kappa_J \qquad (k \ge 1), \qquad \kappa_J = \frac{s_J \ell_J}{12800} = \frac{2^{-13J}}{5{,}120{,}000} > 0.$$

Its fixed nearest-grid oracle generates the same entire transcript.

Proof. 1. Four admissible fixed completions. Set $T = t_{J-1}$ if $J \ge 1$, the delivery round of the last informative event, and set $T = 0$ if $J = 0$. After that event the active interval remains $I_J = [a_J, b_J]$; when $J = 0$, it remains I_0 from the start. Let $C_r = [u_r, v_r]$, $r \in \{1,2,3,4\}$, be the four children of I_J from Lemma 15, and write $s' = s_J/128$ and $\ell' = \ell_J/64$. Construct four alternative completions of the same frozen exterior. For each r, install the patch of Lemma 16 and fill its child interior by

$$f_r(u_r) = H_J - s_J\ell_J/4, \qquad f_r'(t) = -s' + \frac{2s'}{\ell'}(t - u_r) \quad (u_r \le t \le v_r).$$

This specifies the scalar function f_r on the whole line. These choices complete the unresolved interior of I_J; the constructed run still has exactly J informative events. The patch preserves all previously frozen data, is convex and C^1, and has derivative Lipschitz constant at most

$$\frac{100 s_J}{\ell_J} = 2^{-J} \le 1, \qquad \frac{2s'}{\ell'} = \frac{s_J}{\ell_J} = \frac{2^{-J}}{100} \le 1.$$

The first bound controls the new patch and the second its linear child filling. Together with the already frozen pieces, Lemma 18's finite-completion argument gives global 1-Lipschitz derivatives. Thus

$$f_1, f_2, f_3, f_4 \in \mathcal{F}_J.$$

Each f_r has its minimizer at $(u_r + v_r)/2$, by the linear-filling argument in Step 5 of Lemma 18's proof; write $f_r^* = \min_{t\in\mathbb{R}} f_r(t)$. All four are compatible with every reply through round T by Lemma 20.

2. Why their entire future runs agree. Consider round $n > T$ in the constructed run. Write (x_n, p_n) for its query and reply_n for the complete reply bit string. As in Lemma 20, $\mathcal{O}_{f_r}(x_n, p_n)$ denotes the fixed nearest-grid reply for the objective $f_r \circ \pi_1$. If $\pi_1(x_n) \notin I_J$, all four functions have the same frozen value and derivative at $\pi_1(x_n)$, and the same range and encoding rules give identical replies, including overflow when applicable. If $\pi_1(x_n) \in I_J$, the absence of later events means $\eta_{p_n} \ge 2s_J$. Lemma 19 gives identical rounded values and derivatives, and the range check in Step 2

of Lemma 20's proof establishes normal status. These two cases are exhaustive, including queries at the endpoints. Padding belongs to the exterior case because its first coordinate is zero. Hence, on every query of the constructed sequence after T,

$$\mathcal{O}_{f_1}(x_n, p_n) = \cdots = \mathcal{O}_{f_4}(x_n, p_n) = \mathsf{reply}_n \qquad (n > T).$$

This statement about replies also gives the same actual query sequence: Lemma 20 gives the common state after round T, including any unread part of its reply, or the common initial state when $T = 0$. Apply determinism inductively to the next call and every intervening action. Identical replies preserve the same next state, query, and complete report. Thus there is one common report sequence $(z_n)_{n>T}$ for all four fixed functions; no different future behavior has been assumed without justification.

3. One function is inaccurate infinitely often. The closed isolation neighborhoods are

$$N(C_r) = [u_r - \ell_J/100,\ v_r + \ell_J/100] \qquad (r = 1, \ldots, 4).$$

They are pairwise disjoint by Lemma 15. For $r = 1, \ldots, 4$, define

$$\mathcal{N}_r = \{n \in \mathbb{N} : n > T,\ f_r(\pi_1(z_n)) - f_r^* \geq \kappa_J\}, \qquad \kappa_J = s_J \ell_J / 12800 > 0.$$

At any such round, the scalar $\pi_1(z_n)$ belongs to at most one of the disjoint neighborhoods $N(C_1), \ldots, N(C_4)$. For each of the other at least three indices, Lemma 17 applied to f_r gives the defining gap. Consequently

$$\sum_{r=1}^{4} \mathbf{1}_{\{n \in \mathcal{N}_r\}} \geq 3 \qquad (n > T).$$

Here the indicator is one when $n \in \mathcal{N}_r$ and zero otherwise. If all four sets were finite, their union would be finite; choosing $n > T$ beyond every element of that union would make the left side zero, a contradiction. Choose one fixed r_* for which $\mathcal{N}_{r_*}$ is infinite. Enumerate its elements increasingly as $m_1 < m_2 < \cdots$; these are integer rounds, so $m_k \to \infty$. Set $f = f_{r_*}$ and $F(x) = f(\pi_1(x))$. Since every real number occurs as a first coordinate, $F^* = \min_{x \in \mathbb{R}^d} F(x) = f_{r_*}^*$, and therefore

$$F(z_{m_k}) - F^* \geq \kappa_J \qquad (k \geq 1).$$

The index r_* is selected once, from four entire compatible runs. Its function reproduces all past and future replies and is not changed when a round or horizon is selected.

□

C.10 Proof of the Fixed-Instance Lower Theorem

Theorem 23 (Fixed-instance lower bound, restated)**.** *Fix an integer $d \geq 1$, the zero anchor $a = 0 \in \mathbb{R}^d$, and any deterministic uniform finite-control program $\mathcal{A}$ satisfying the model of Section 3 with only the live-storage bound removed. In particular, retain the serial reply protocol and the $Wd = 2^{12}d$ bit-action cap. There is one fixed convex $F \in C^1(\mathbb{R}^d)$ with a minimizer and a globally 1-Lipschitz gradient. Write $F^* = \min_{x \in \mathbb{R}^d} F(x)$. There are positive constants c, C and strictly increasing rounds $n_1 < n_2 < \cdots$, with $n_j \to \infty$, such that the report subsequence in its fixed nearest-grid oracle run satisfies*

$$F(z_{n_j}) - F^* \geq c e^{-C\sqrt{n_j}} \qquad (j \geq 1).$$

The oracle uses the tie and overflow rules of Appendix A.4. The objective may depend on the program, but neither it nor the rounding rule depends on a terminal horizon. Consequently no program in this class has an eventual every-round objective-gap guarantee with exponent $\omega(\sqrt{n})$ on every fixed instance. This is Theorem 2.

Proof. 1. Fixing the program and constructing one infinite run. Fix the program $\mathcal{A}$, d, and the input records $\mathsf{Pos}(d)$ and $\mathsf{DyadVec}(0)$ before answering calls. Use the adaptive rule of Appendix C.7. Lemma 20 supplies a nonempty family of completions after each call and proves legality of every reply. This defines the sequence of program states, queries, replies, and reports without specifying any terminal horizon. The event count is either infinite or a finite integer J.

- If events are infinite, choose the scalar limit f from Lemma 18. Lemma 20 shows that the fixed nearest-grid oracle for $x \mapsto f(\pi_1(x))$ reproduces the entire run. Use the preceding report rounds $n_j = t_j - 1$ from Lemma 21, retaining $j \geq 1$; omitting the event indexed by zero leaves an unbounded, strictly increasing sequence. That lemma gives
$$f(\pi_1(z_{n_j})) - \min f \geq \frac{1}{5{,}120{,}000} e^{-289\sqrt{n_j}}, \qquad n_j \longrightarrow \infty.$$

- If there are exactly $J \in \mathbb{N}_0$ events, let f be the one fixed scalar completion in Lemma 22. Write its increasing bad-round sequence $(m_j)_{j \geq 1}$ as $n_j = m_j$ for this branch. Then $n_j \to \infty$ and
$$f(\pi_1(z_{n_j})) - \min f \geq \kappa_J \geq \kappa_J e^{-289\sqrt{n_j}},$$
since $0 < e^{-289\sqrt{n_j}} \leq 1$.

Thus the same exponent constant works in both cases:
$$C = 289, \qquad c = \begin{cases} 1/5{,}120{,}000, & \text{if events are infinite,} \\ \kappa_J, & \text{if there are exactly } J \text{ events.} \end{cases}$$
Both choices of c are positive. The possibly instance-dependent constant c is fixed once f is fixed.

2. Embedding the scalar instance in dimension d. Define $F(x) = f(x_1)$ for all $x \in \mathbb{R}^d$. For $d = 1$ this is just f; for general d,
$$F^* = \min f, \qquad \arg\min F = (\arg\min f) \times \mathbb{R}^{d-1}, \qquad \nabla F(x) = (f'(x_1), 0, \ldots, 0).$$
The first two identities follow because every scalar first coordinate is realized by some $x \in \mathbb{R}^d$. The set of minimizers is nonempty by the construction of f. Continuity of f' gives $F \in C^1$, and
$$\begin{aligned} \|\nabla F(x) - \nabla F(y)\|_2 &= |f'(x_1) - f'(y_1)| \\ &\leq |x_1 - y_1| \leq \left(\sum_{i=1}^{d} |x_i - y_i|^2 \right)^{1/2} = \|x - y\|_2. \end{aligned}$$
For $0 \leq t \leq 1$, scalar convexity also gives
$$\begin{aligned} F((1-t)x + ty) &= f((1-t)x_1 + ty_1) \\ &\leq (1-t)f(x_1) + tf(y_1) = (1-t)F(x) + tF(y). \end{aligned}$$
Hence F has every smoothness and convexity property claimed. Its extra gradient fields are the zeros already included in the constructed transcript, so the d-dimensional oracle run is the same run used in the preceding gap estimates.

3. The fixed-instance quantifiers. The adaptive construction is used to establish existence, while the final oracle always applies the prescribed fixed nearest-grid rule. After selecting either the infinite nested limit or the finite-event completion, one function is defined on the whole domain. For every finite prefix, Lemma 20 proves that this same function gives precisely the replies used to construct that prefix. Thus the quantifier order is
$$\begin{aligned} \forall \mathcal{A}\ \forall d \in \mathbb{N} \quad \exists F,\ c > 0,\ C > 0 \quad \exists\, n_1 < n_2 < \cdots,\ n_j \to \infty : \\ F(z_{n_j}) - F^* \geq c e^{-C\sqrt{n_j}} \qquad (j \geq 1). \end{aligned}$$
Here $\mathcal{A}$ ranges over the stated program class, and z_n is its report under the fixed oracle for the chosen F. There is no choice of a new objective after a terminal horizon is fixed. The argument used no bound on live storage.

4. Excluding a faster eventual every-round exponent. Suppose, to the contrary, that the fixed program guarantees on every fixed admissible objective F a bound with a finite integer onset $N_F \in \mathbb{N}$, a finite prefactor $M_F > 0$, and an exponent function $\Phi_F : \mathbb{N} \to \mathbb{R}$ such that
$$F(z_n) - F^* \leq M_F e^{-\Phi_F(n)} \qquad (n \geq N_F), \qquad \frac{\Phi_F(n)}{\sqrt{n}} \longrightarrow \infty.$$

The exponent function may depend on F; no monotonicity is assumed. Allowing $M_F > 0$ loses no generality: a zero-error bound also implies any positive choice of M_F. Apply the guarantee to the one constructed F. Since $n_j \to \infty$, all sufficiently late subsequence elements satisfy $n_j \geq N_F$, and

$$ce^{-C\sqrt{n_j}} \leq M_F e^{-\Phi_F(n_j)}.$$

Both sides are positive, so taking logarithms and rearranging gives

$$\log c - C\sqrt{n_j} \leq \log M_F - \Phi_F(n_j) \quad \Longrightarrow \quad \Phi_F(n_j) - C\sqrt{n_j} \leq \log(M_F/c).$$

But

$$\Phi_F(n_j) - C\sqrt{n_j} = \sqrt{n_j}\left[\frac{\Phi_F(n_j)}{\sqrt{n_j}} - C\right] \longrightarrow \infty,$$

contradicting the fixed finite right side. The same contradiction allows a positive instance-dependent exponent factor $a_F > 0$ and a fixed integer index shift $s_F \in \mathbb{Z}$. For integer $n > \max\{0, s_F\}$, the argument $n - s_F$ belongs to $\mathbb{N}$, and

$$\frac{a_F \Phi_F(n - s_F)}{\sqrt{n}} = a_F \frac{\Phi_F(n - s_F)}{\sqrt{n - s_F}} \sqrt{1 - s_F/n} \longrightarrow \infty \qquad (n \to \infty).$$

Here $n - s_F \to \infty$ and $\sqrt{1 - s_F/n} \to 1$. This proves the eventual-rate obstruction and hence Theorem 2.

□

C.11 The Same Order in Total Charged Bit Work

Proposition 24 (Charged bit-work guarantees)**.** *Fix an integer $d \geq 1$. Let $\mathcal{B}_n$ be the cumulative number of charged bit actions through round n, and write $F^* = \min F$ for the objective in each run. For every fixed admissible instance, the program of Theorem 14 satisfies, on every legal rounding transcript and at every sufficiently late round,*

$$F(z_n) - F^* \leq C_0 \exp(-c'_d \sqrt{\mathcal{B}_n}).$$

Here C_0 is the instance constant in that theorem. For every deterministic program in the class of Theorem 23, with zero anchor and no live-storage bound, there are one fixed admissible instance, a constant $c > 0$, and strictly increasing rounds $n_k \to \infty$ such that its fixed nearest-grid oracle run satisfies

$$F(z_{n_k}) - F^* \geq c \exp(-C'_d \sqrt{\mathcal{B}_{n_k}}) \qquad (k \geq 1).$$

For $W = 2^{12}$, the positive constants may be chosen as

$$c'_d = \frac{\gamma_d}{\sqrt{2Wd}}, \qquad C'_d = \frac{13 \log 2}{\sqrt{8(d+1)}},$$

where γ_d is from Theorem 14. They depend only on dimension and the fixed budgets; C_0 and c may depend on the instance. If $\sup_n \mathcal{B}_n < \infty$ on the constructed lower-bound instance, a fixed positive gap still occurs along an unbounded subsequence of rounds, so its report errors do not converge to zero.

Proof. We use opposite inequalities for upper and lower guarantees, so we state each change of clock explicitly.

1. The upper guarantee in actual charged work. The count starts at $\mathcal{B}_0 = 0$, and the per-round work cap gives

$$0 \leq \mathcal{B}_n - \mathcal{B}_{n-1} \leq Wd.$$

Summing over rounds $1, \ldots, n$ yields $\mathcal{B}_n \leq Wdn$. Fix an admissible instance and take C_0, γ_d, n_0, n_* from Theorem 14, uniformly over its legal transcripts. For every $n \geq \max\{n_*, 2n_0\}$, that theorem applies and

$$n - n_0 \geq n/2 \geq \frac{\mathcal{B}_n}{2Wd}.$$

The square root is increasing and multiplication by $-\gamma_d$ reverses order. Therefore

$$F(z_n) - F^* \le C_0 \exp[-\gamma_d \sqrt{n - n_0}] \le C_0 \exp\left[-\frac{\gamma_d}{\sqrt{2Wd}} \sqrt{\mathcal{B}_n}\right].$$

Take $c'_d = \gamma_d / \sqrt{2Wd} > 0$.

2. Infinitely many lower-bound events. Fix a program in the class of Theorem 23 and use its adaptive construction. When informative events are infinite, take the fixed scalar limit f from Lemma 18 and set $F(x) = f(\pi_1(x))$. By Lemma 20, the fixed oracle reproduces all charged actions, so $\mathcal{B}_n$ is the same in the constructed and fixed oracle runs. For informative event i, let p_i be its precision and t_i its delivery round; set $n_j = t_j - 1$, as in Lemma 21. The mandatory read counts in Table 18 and (23) give, for $j \ge 1$,

$$\begin{aligned} \mathcal{B}_{n_j} &\ge \sum_{i=0}^{j-1} R(p_i) \ge 32(d+1) \sum_{i=0}^{j-1} p_i \\ &> 16(d+1) \sum_{i=0}^{j-1} (i+1) \\ &= 8(d+1) j(j+1) \ge 8(d+1) j^2, \end{aligned}$$

using $p_i > (i+1)/2$ and the sum of the first j positive integers. Thus

$$j \le \frac{\sqrt{\mathcal{B}_{n_j}}}{\sqrt{8(d+1)}}.$$

Inserting this upper bound on depth into the negative exponent in Lemma 21 gives

$$\begin{aligned} F(z_{n_j}) - F^* &\ge \frac{1}{5{,}120{,}000} \exp[-13(\log 2) j] \\ &\ge \frac{1}{5{,}120{,}000} \exp\left[-\frac{13 \log 2}{\sqrt{8(d+1)}} \sqrt{\mathcal{B}_{n_j}}\right]. \end{aligned}$$

Both the round indices and the charged work are unbounded in this case. Taking $c = 1/5{,}120{,}000$ and $C'_d = 13 \log 2 / \sqrt{8(d+1)} > 0$ proves the stated work guarantee.

3. Finitely many events, including bounded work. If exactly $J \in \mathbb{N}_0$ events occur, Lemma 22 gives one fixed instance and strictly increasing rounds $m_k \to \infty$ with $F(z_{m_k}) - F^* \ge \kappa_J > 0$. Use the same positive C'_d as in Step 2. Since $\mathcal{B}_n \ge 0$,

$$0 < \exp[-C'_d \sqrt{\mathcal{B}_{m_k}}] \le 1, \qquad F(z_{m_k}) - F^* \ge \kappa_J \exp[-C'_d \sqrt{\mathcal{B}_{m_k}}].$$

Thus $c = \kappa_J$ and $n_k = m_k$ give the stated lower bound. This includes $J = 0$ and permits either bounded or unbounded charged work. If $\sup_n \mathcal{B}_n < \infty$ in the constructed run, infinitely many events would contradict Step 2's quadratic work bound. Events must then be finite, and the positive gap κ_J on the subsequence above rules out convergence of the report errors to zero.

4. Why the work exponent is also sharp. Let (n_k) be the bad-round sequence supplied by the applicable event case. Suppose that $\mathcal{B}_n$ is unbounded on this fixed instance. The cumulative count is nondecreasing. For every integer threshold $b > 0$, choose N_b with $\mathcal{B}_{N_b} \ge b$. Since $n_k \to \infty$, all sufficiently large k satisfy $n_k \ge N_b$ and hence $\mathcal{B}_{n_k} \ge b$. Thus $\mathcal{B}_{n_k} \to \infty$, including in the finite-event case.

If a faster eventual every-round guarantee on this instance had the form

$$F(z_n) - F^* \le M \exp[-\Psi(\mathcal{B}_n)], \qquad \frac{\Psi(b)}{\sqrt{b}} \longrightarrow \infty \quad (b \to \infty),$$

for a finite $M > 0$ and a function $\Psi : \mathbb{N}_0 \to \mathbb{R}$, combining it with the lower bound on sufficiently late bad rounds gives

$$\Psi(\mathcal{B}_{n_k}) - C'_d \sqrt{\mathcal{B}_{n_k}} \le \log(M/c).$$

The left side equals

$$\sqrt{\mathcal{B}_{n_k}}\left[\frac{\Psi(\mathcal{B}_{n_k})}{\sqrt{\mathcal{B}_{n_k}}} - C'_d\right] \longrightarrow \infty,$$

contradicting the fixed right side. When the work count is bounded, Step 3 instead supplies the positive error obstruction.

All comparisons use the actual charged actions of the program; external oracle construction work is absent from these counts. □

D Numerical Illustrations

This chapter illustrates finite runs, organized around three questions:

1. What fixed objective, output rule, and numerical schedule is used?
2. Which operations are counted, and how are the finite rates fitted?
3. How do the convergence curves and construction snapshots reflect the mechanisms proved above?

The plotted gap is $F(z) - \min F$ for the actual output z. Exact rational calculations check replies and gaps against one completed objective per run. The work T is a partial operation count, distinct from the full round clock n. These finite illustrations use the reduced schedules and privileged drivers specified below; the universal rate is proved in Appendices B and C.

D.1 Objectives and Optimization Schedules

A fixed smooth objective for each run. The scalar component f uses the nested intervals of Appendix C. One run is constructed and checked in the following order:

$$\text{finite prefix} \longrightarrow \text{complete the derivative linearly} \longrightarrow \text{exact replay}.$$

The middle step fills the final unresolved interval. The completion is convex, C^1, and piecewise quadratic with a globally Lipschitz derivative. Hard behavior is asserted only for the observed prefix. For the main experiments, the objective is

$$F(x) = f(v^\top x) + \frac{1}{20}\big(\|x\|_2^2 - (v^\top x)^2\big), \qquad v = \begin{cases} (1), & d = 1, \\ (3/5, 4/5), & d = 2, \\ (2/3, 1/3, 2/3), & d = 3. \end{cases}$$

Each v is a rational unit vector. In dimensions two and three the hard direction is rotated and a positive transverse quadratic makes the objective depend on every direction. Its gradient is globally 1-Lipschitz. Each run has its own fixed objective; the optimizer uses only numerical replies, and does not access the hidden intervals, minimizer, or exact objective values when choosing its queries or reports.

Online selection of the output. Let $x_0 = r$ be the preceding completed-epoch report and $x_1 = b$ the current epoch's least-observed-value query. In the numerical runs we select

$$i_\star = \min \operatorname*{argmin}_{i \in \{0,1\}} \widehat{f}_i, \qquad z = x_{i_\star},$$

where $\widehat{f}_i$ is the finite-precision observation stored in slot i, not the true function value; before b is available, $z = r$. The output rule has three implementation requirements:

1. *Initialize and compare.* A precision-1 query supplies the anchor's observed value. Keep observations at their original precisions and compare after exactly aligning their dyadic scales.
2. *Resolve ties by storage order.* The outer minimum selects slot zero on a tie, hence r. Likewise, an incumbent precedes a new query when updating b.

3. *Publish a complete point.* Consume the reply and finish copying before changing z; count comparisons and copies. Hold the previous output during this work. At epoch completion, replace r by that epoch's best point.

The resisting oracle observes this selected z, so event certificates refer to the same output. True objective values are used afterward to evaluate gaps and verify the replies and certificates.

One-dimensional schedule. The noisy ternary schedule of Appendix B uses

$$m = 1, 2, 4, \ldots, 1024, \qquad p = m, \qquad \text{decisions per epoch} = 8m.$$

Query coordinates and function evaluation use exact rational arithmetic; replies are rounded and fully decoded according to the finite numerical interface, with the online output selection above. Every completed epoch satisfies the one-dimensional error bound in Appendix B for the completed test function.

Multidimensional schedule. The ellipsoid runs use $m = 1, 2, 4, \ldots, 256$ and the online output above. They keep the expanding initial ellipsoid, dense dyadic center and factor, value probes, short directions, two inflations, and outward rounding. Table 19 lists the reduced numerical constants.

Table 19: Proof and simulation schedules for $d \geq 2$. The reduced schedule makes finite numerical runs tractable; the universal guarantee for the proof schedule is not assumed for these constants.

Parameter	Proof schedule	Numerical schedule
Cut limit	$K_m = 64d(d+1)(m+1)$	$K = 2d(d+1)(m+1)$
Probe exponent	$b_m = 2^{12}d^3q_m$	$b = 4K + 4m + 64$
Oracle precision	$P_m = 2^{16}d^4q_m$	$p = \lceil (b + 2m + 64)/8 \rceil + 4$

The coefficient builder uses 48 binary fractional bits, integer restoring division and square root, and directed fixed-point arithmetic (Brent & Zimmermann, 2010, Sections 1.4–1.5 and 3.1.9). For every executed update, a separate rational interval calculation encloses the ideal coefficients and verifies the entrywise error bound ϵ_d of Lemma 6. The square-root enclosures are checked by integer squaring; no floating-point reference is used to decide this bound.

Early termination and query counts. The multidimensional runs retain the zero-difference branch of Algorithm 1. After a *probe batch* consisting of one center query and d directional queries, the epoch ends if all observed differences $\Delta_i = \widetilde{F}(c + 2^{-b}Be_i) - \widetilde{F}(c)$ vanish. It publishes the epoch's least-observed-value candidate and proceeds to the next epoch. This test uses the finite-precision replies. The final zero-difference batch contributes $d + 1$ useful replies but performs no ellipsoid update.

Every two-dimensional epoch reached its prescribed limit K. The three-dimensional run reached the limit for $m = 1, 2, 4, 8, 16$, but terminated early in the four epochs listed in Table 20.

Table 20: Early termination in the three-dimensional run. Executed probe batches include the final zero-difference batch; the number of ellipsoid updates in each row is therefore one smaller.

Epoch m	Prescribed limit K	Executed probe batches
32	792	705
64	1,560	1,119
128	3,096	1,961
256	6,168	3,636

The first five epochs used $48 + 72 + 120 + 216 + 408 = 864$ batches. Including the single anchor-initialization query, the total number of useful replies is consequently

$$1 + 4(864 + 705 + 1119 + 1961 + 3636) = 33{,}141,$$

compared with $1 + 4 \cdot 12{,}480 = 49{,}921$ if every epoch had reached its prescribed limit. Thus the reported query count includes all executed probes, including those that triggered early termination.

D.2 Cost Accounting and Observed Rates

Table 21 separates the three recorded costs. Arithmetic charges use actual operand lengths, and

$$B = \sum_{\text{replies}} R(p), \qquad T_{\text{scan}} = \sum_{\text{replies}} \left\lceil \frac{R(p)}{4096d} \right\rceil .$$

These quantities are measured from the run, not assigned from m^2.

Table 21: Numerical cost measures. None is the full physical-round clock n of the theorems.

Measure	Included	Not included
T	Addition, copy, shift, comparison, short shift-add multiplication, restoring division/square root, query construction, full replies, geometric updates, publication.	Head addressing, unary rulers, allocation, finite control, padding housekeeping, host-language rational normalization.
B	All bits in completely decoded replies.	Arithmetic and machine overhead.
T_{scan}	The reply-reading lower clock under the $4096d$ work cap.	Arithmetic and other machine overhead.

The output trace follows two rules:

- Record each completed output change and hold that point between changes; do not interpolate between epoch endpoints.
- Compute the true gap before logarithmic display, without clipping or an artificial error floor.

For each cost measure C, take the last five completed epochs. Let E_i be their gaps and $w_i = C_i/C_{\max}$ their normalized costs. The fit in Table 22 is

$$\mathcal{G} = \{0.15, 0.155, \ldots, 1.20\},$$
$$S(\beta) = \min_{a,b} \sum_{i=1}^{5} [-\ln E_i - a - b w_i^{\beta}]^2,$$
$$\widehat{\beta} \in \operatorname*{argmin}_{\beta \in \mathcal{G}} S(\beta).$$

The intercept absorbs the multiplicative gap constant. We also report R^2 with $\beta = 1/2$ fixed. Both diagnostics describe five finite points per run, not an infinite-time exponent estimate.

Table 22: Finite-prefix convergence diagnostics. T denotes accounted bit operations and B denotes complete-reply bits. Each fit uses the last five completed epochs of the corresponding run. Useful replies include anchor initialization and account for the early termination in Table 20.

d	Final m	Useful replies	Final gap	β_T	$R^2_{\sqrt{T}}$	β_B
1	1024	32,753	1.217×10^{-2463}	0.505	0.999991	0.500
2	256	18,721	1.426×10^{-150}	0.525	0.999769	0.505
3	256	33,141	4.947×10^{-186}	0.495	0.999589	0.470

We retain the complete query/reply records and check them with a separate program that imports neither the optimizer nor its objective evaluator. It rebuilds each scalar objective as explicit quadratic pieces, verifies continuity of values and derivatives and slopes in $[0, 1]$, and computes the minimum

by exact rational arithmetic. All 84,615 main-run replies are checked against the fixed nearest-grid rule of Appendix A.4, including ties toward zero and overflow without clipping. The checker also reconstructs the query schedule, finite-difference directions, rounded updates, and output selections from the saved replies. It verifies all 714 event lower bounds at the selected outputs immediately before the informative replies, together with complete-reply bit totals and scan costs. These are the events shown in Figures 6 and 5.

These checks certify the finite records and their mathematical consistency. They do not supply the omitted tape-machine operations, or a uniform convergence guarantee for the reduced schedule. Boundary tests and deliberately corrupted records test rejection of incorrect rounding, coefficients, replies, and output timing.

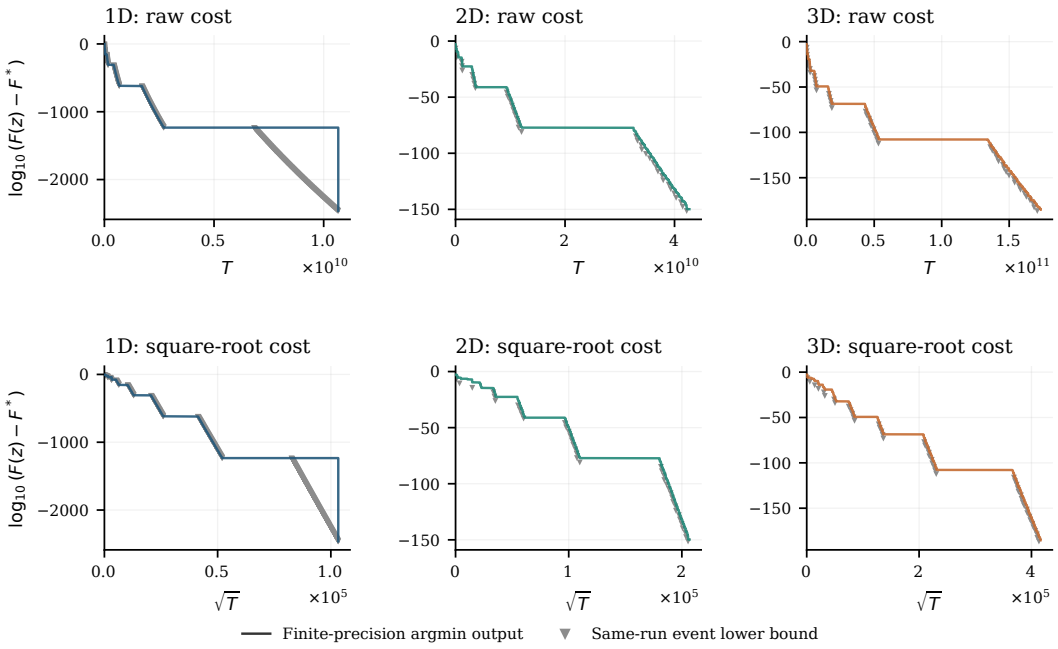

Figure 5: **Raw cost and square-root cost for the same optimization trajectories.** Each column shows one fixed smooth objective, with the accounted bit-operation cost T above and $\sqrt{T}$ below. Solid lines show the true errors of the online finite-precision argmin outputs throughout computation. Triangular markers are event lower certificates for these same outputs on the same objective, at the cost recorded before the informative reply arrives. Cost definitions and parameter choices are given above.

D.3 Convergence Profiles

Figure 6 illustrates the signature $-\log(\text{gap}) \approx a + b\sqrt{T}$ in dimensions one through three. Read the profiles in two ways:

- The online finite-precision selection in Appendix D.1 reveals improvements within epochs. Every plotted gap uses that output and its run's fixed objective.
- The last-five-epoch fits give $(\widehat{\beta}_1, \widehat{\beta}_2, \widehat{\beta}_3) = (0.505, 0.525, 0.495)$. They describe the finite trajectories under the partial cost T defined in Appendix D.2.

Figure 5 shows the same runs against both raw cost and square-root cost.

D.4 Visualization of the Lower-Bound Construction

Figure 7 follows 2,048 nested depths:

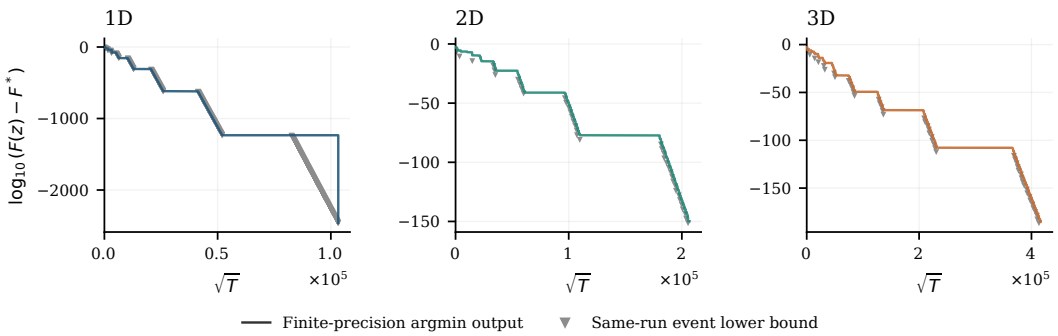

Figure 6: **Visualizing root-exponential convergence in one to three dimensions.** Solid lines show the true error of the finite-precision argmin output throughout computation. Triangles certify lower bounds for that same output immediately before informative replies. The one-dimensional run uses ternary search; the two- and three-dimensional runs use reduced-constant ellipsoid schedules. Cost T is a partial bit-operation count, excluding tape management and finite-control overhead; it is distinct from the full physical-round clock n in the theorems.

- *Driver and objective.* Queries use the hidden interval and slope scale, so this is a privileged construction driver. Dimensions two and three use $F(x) = f(x_1)$; each completed objective differs from the main optimization runs.
- *Displayed quantities.* Exact pre-event gaps and their certified lower bounds are plotted against $\sqrt{T_{\mathrm{scan}}}$.

The near-linear trends illustrate the mechanism

$$\text{depth } j: \quad \text{gap certificate} \propto 2^{-13j}, \quad p_j = \Omega(j), \quad T_{\mathrm{scan}} = \Omega(j^2)$$

for this driver. The saved transcripts additionally permit independent exact checking of all 12,288 replies and 6,144 event bounds across the three dimensions, including agreement with the fixed nearest-grid oracle.

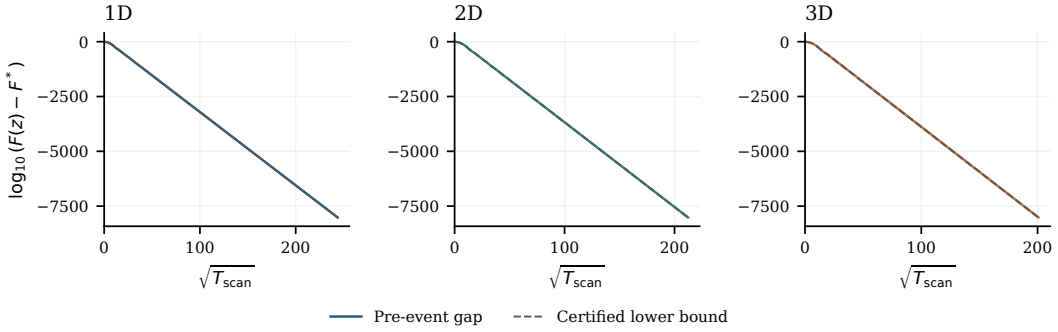

Figure 7: **Finite illustration of the lower-bound mechanism.** Exact pre-event gaps and certified lower bounds along a privileged construction driver, plotted against $\sqrt{T_{\mathrm{scan}}}$. Dimensions two and three are scalar embeddings. All replies and certificates are checked against one completed objective per run; the diagnostic illustrates a finite hard prefix.

D.5 Construction Snapshots

The two-dimensional visualization uses one epoch with $m = 2$, $K = 36$, $b = 216$, and $p = 40$ from the reduced schedule above, starting from $E(0, 4I)$. The objective is the quadratic

$$F(x) = \tfrac{1}{2}(x - x^*)^T \begin{pmatrix} 3/4 & 1/4 \\ 1/4 & 1/2 \end{pmatrix} (x - x^*), \qquad x^* = (1, 1/2).$$

All 108 replies are finite lattice values. The direction estimates, short coefficients, directed dyadic rounding, and both inflations follow the same numerical ellipsoid update as the main runs. Exact checks verify minimizer containment and decreasing area at every update. All 36 coefficient updates also pass the rational interval check; the area-contraction check uses a rational bound rather than a floating-point exponential comparison.

The one-dimensional snapshots in Figure 8 use coarse/precise pairs $(0, 3)$, $(1, 4)$, and $(2, 5)$. For each pair:

1. Check all four completions against the previous replies and the new coarse reply, then select a child.
2. Display the full patch above and magnify the valley below, using
$$H_{j+1} = H_j - s_j \ell_j / 4, \qquad s_{j+1} = s_j / 128.$$
3. Check the final function against all six replies; the three precise-query jets must remain exactly unchanged.

The driver accesses the active interval to place queries and reports, so these snapshots illustrate compatibility rather than optimizer performance.

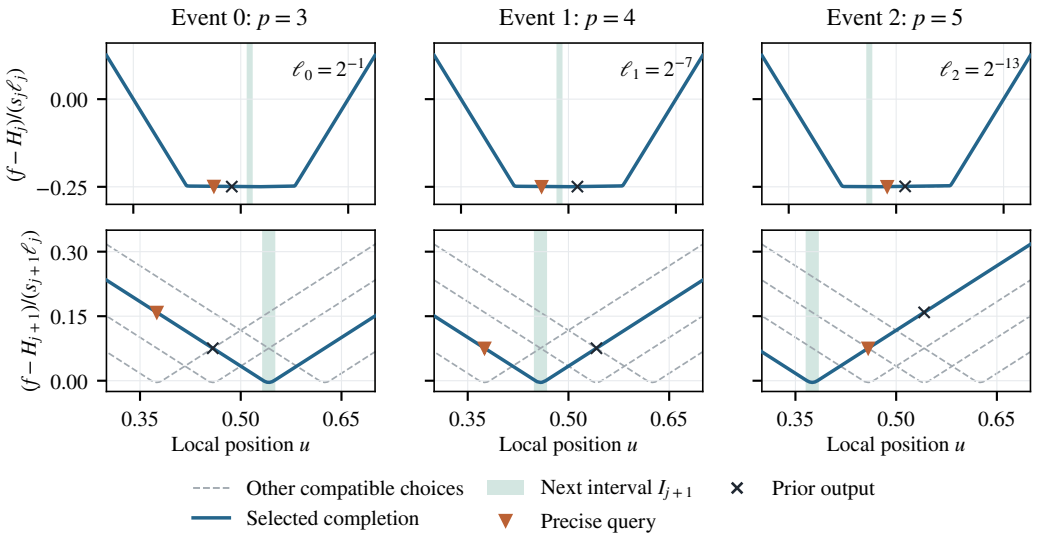

Figure 8: Compatible one-dimensional functions along three queries. Each column shows four smooth convex completions on the current parent interval, with the full patch above and its magnified valley below. All four fit the previous replies and the current coarse reply. The precise query (triangle) and preceding report (cross) select a child (green) avoiding both; the selected completion is blue, and later events refine only that child. The local coordinate is $u = (x - a_j)/\ell_j$; vertical scales are shown on the axes. All replies are checked against the final fixed completion.

E SUPPLEMENTARY MATERIAL

The supplementary archive contains the code, recorded data, and figures for Appendix D. It is organized into two sibling directories:

- `experiments/` contains the optimization and construction programs, numerical datasets, plotting scripts, and exact finite-run checkers. Its `certificates/` subdirectory supplies five complete compressed transcripts and recorded validation results for all six runs. The full one-dimensional optimizer transcript is omitted to meet the upload limit; its data, digest, and regeneration command are retained. The accompanying README explains the parameters, cost conventions, source provenance, and commands for both short checks and full reruns.
- `figures/` contains PDF and PNG versions of the manuscript figures and additional comparison plots. Its README maps every figure to its data and generating script.

The archive-root README gives the setup and execution instructions, and `MANIFEST.sha256` lists the file checksums. Python 3.12 or later is required; plotting dependencies are pinned in `experiments/requirements.txt`, while the exact certificate checkers use only the Python standard library. The package uses relative paths and requires no external dataset or manuscript source tree. Figures can be regenerated from the recorded data without rerunning the long optimization experiments.

The certificates verify finite trajectories on their completed objectives, including oracle replies, numerical updates, output selection, and event bounds, as applicable to each experiment. Their scope is the numerical methodology in Appendix D.1: the plotted cost is a partial operation count, the multidimensional schedules use reduced constants, and the lower-construction driver accesses hidden interval information. The universal convergence results are established by the proofs in Appendices B and C.