# Optimizing a 2D Function Satisfying Unimodality Properties

Erik D. Demaine[1] and Stefan Langerman[2⋆]

[1] MIT Computer Science and Artificial Intelligence Laboratory, 32 Vassar Street, Cambridge, MA 02139, USA, edemaine@mit.edu

[2] Département d'informatique, Université Libre de Bruxelles, ULB CP212, Bruxelles, Belgium, Stefan.Langerman@ulb.ac.be

**Abstract.** The number of probes needed by the best possible algorithm for locally or globally optimizing a bivariate function varies substantially depending on the assumptions made about the function. We consider a wide variety of assumptions—in particular, global unimodality, unimodality of rows and/or columns, and total unimodality—and prove tight or nearly tight upper and lower bounds in all cases. Our results include both nontrivial optimization algorithms and nontrivial adversary arguments depending on the scenario.

## 1 Introduction

Many problems in geometry, in particular problems about the set of distances among geometric objects (diameter, closest pairs, farthest pairs, etc.) can be seen as finding a maximum in a two-dimensional array. This abstraction is used by many algorithms, but one of the most remarkable results is probably the $O(n)$-time algorithm for optimizing "totally monotone" $n \times n$ matrices of Aggarwal et al. [1] and the application of this algorithm to many geometric proximity problems. This work later found many applications as a general technique for speeding up dynamic-programming algorithms. A survey of these applications can be found in [4]. The motivation for our work came from the desire to understand what matrix properties enable speeding a search from linear time down to a polylogarithmic number of probes. For example, we want to know the weakest properties that would have to be expressed in order to find the closest pair of points between two given convex polygons in logarithmic time [3]. Such an understanding could lead to many generalizations, for example to other metric spaces or variants of convexity.

The most general formulation of this discrete optimization problem is to maximize a given function $f : D \to \mathbb{R}$ over a discrete (finite) domain $D$. In general, of course, this problem may require $|D|$ probes to $f$. One approach to making optimization more tractable is to be satisfied with finding a *local* maximum, i.e., a point at which $f$ attains a value larger than all "neighboring" points, for some definition of neighborhoods. In particular, for the standard 1D

---

⋆ Chercheur qualifié du FNRS

domain $D = \{1, 2, \ldots, n\}$, Fibonacci search [6] finds a local maximum using $\log_\phi n + O(1)$ probes, where $\phi = (1 + \sqrt{5})/2$ is the golden ratio. Surprisingly, the problem complexity grows exponentially in 2D, even for a square domain $D = \{1, 2, \ldots, n\} \times \{1, 2, \ldots, n\}$: independently, Llewellyn et al. [7,8], Althöfer and Koschnick [2], and Mityagin [9] proved that $\Theta(n)$ probes to a function $f$ are sufficient and sometimes necessary to find any local optimum in an $n \times n$ array $D$. (Unless otherwise specified, we use the 4-neighborhood $\{(i-1, j), (i+1, j), (i, j-1), (i, j+1)\}$ of a point $(i, j)$ in the square grid.) Thus weakening the optimization problem to finding local maxima does not provide an exponential speedup in higher dimensions like it did in 1D. See also [10] for a survey on local optimization methods.

Another approach to making optimization more tractable is to add assumptions about the function $f$. Other than the monotonicity assumptions mentioned above, the main example in the literature of which we are aware is a kind of Lipschitz condition: if $f$ is *integral Lipschitz* in the sense that, between two neighboring points $x$ and $y$, $f(x)$ and $f(y)$ are integral and differ by at most $L$, then it is possible to find a local maximum in $O(L \log n)$ probes [2]. Another simple example is that, if we assume that $f$ is *unimodal* (denoted "$\odot$ unimodal"), i.e., it has exactly one local maximum, then finding local maxima and finding global maxima are equivalent. One could hope that having this structural information about the function would also help in finding that maximum. Unfortunately, a careful reading of the construction in [9] of 2D functions $f$ requiring $\Theta(n)$ probes reveals they are in fact $\odot$ unimodal.

We study the related condition that the 2D function $f$ is unimodal in every column ($\updownarrow$ *unimodal*) and/or in every row ($\leftrightarrow$ *unimodal*). These properties are satisfied by e.g. convex functions, but are more general: for example, the distance function between a point on a convex chain and a point on a monotone chain satisfies one of these properties (in fact, it is $\updownarrow$ convex) but not the other. While seemingly weaker than $\odot$ unimodality, these properties are incomparable to unimodality, and in fact result in exponential speedup for finding local maxima. We also study the stronger condition that a function is *totally unimodal* in the sense that every submatrix is $\odot$ unimodal. This property is the first that allows us to find the (unique) local maximum of the array in $O(\log n)$ probes. Table 1 summarizes all of our results.

A notion related to a totally unimodal matrix is a *unique-sink orientation* of the $m \times n$ grid graph, as considered for arbitrary-dimensional grids in [5]. However, the latter notion is less restrictive: essentially, unique-sink orientations capture only relative comparisons between adjacent vertices, whereas total unimodality captures comparisons between arbitrary vertices in a total order. The relative comparisons of unique-sink orientations may not even be realizable by a total order because of directed cycles in the orientation. When restricted to the two-dimensional case, the algorithms of [5] have running time $O(m+n)$; our totally unimodal algorithms are exponentially faster (but less general).

In the next section, we show how total properties of matrices can be expressed as a set of forbidden partial orders in submatrices. This characterization allows

| Assumption | Local optimization | | Global optimization | |
|---|---|---|---|---|
| None | $\leq min \cdot (\lg \frac{max}{min} + 4) + O(\lg max)$ | [9] | $\leq m \cdot n$ | [obvious] |
| | $\geq \min\{min, max/2\}$ | [9] | $\geq m \cdot n$ | [obvious] |
| Totally $+j$ monotone | $\leq min \cdot (\lg \frac{max}{min} + 4) + O(\lg max)$ | [9] | $O(min(1 + \lg \frac{max}{min}))$ | [1] |
| | $\geq \log_\phi max$ | [Lem. 7] | $\Omega(min(1 + \lg \frac{max}{min}))$ | [1] |
| $\odot$ unimodal | $\leq min \cdot (\lg \frac{max}{min} + 4) + O(\lg max)$ | [9] | same as local | |
| | $\geq \min\{min, max/2\}$ | [9] | | |
| $\updownarrow$ unimodal | $\leq \log_\phi m \log_\phi n + O(\lg n)$ | [Lem. 8] | $\leq n \log_\phi m + O(n)$ | [Lem. 12] |
| | $\geq \frac{1}{4} \lg m \lg n - O(\lg m \lg \lg n)$ if $m \leq n$ | [Thm. 1] | $\geq n \log_\phi m - O(n)$ | [Lem. 13] |
| | $\geq \frac{1}{4} \lg^2 n - O(\lg n \lg \lg n)$ if $m \geq n$ | [Thm. 1] | | |
| $\odot$, $\updownarrow$ unimodal | same as $\updownarrow$ unimodal | | same as local | |
| $\updownarrow$, $\leftrightarrow$ unimodal | $\leq \frac{3}{\lg \phi} \lg^2 min + O(\lg max)$ | [Thm. 2] | $\leq min(\log_\phi max + O(1))$ | [Lem. 12] |
| | $\geq \frac{1}{4} \lg^2 min - O(\lg max \lg \lg max)$ | [Thm. 3] | $\Omega(min)$ | [Lem. 14] |
| | $\geq \log_\phi max$ | [Lem. 7] | | |
| $\odot$, $\updownarrow$, $\leftrightarrow$ unimodal | same as $\odot$, $\updownarrow$ unimodal | | same as local | |
| $\odot$, $\updownarrow$, $\leftrightarrow$ unimodal & totally $+i$, $+j$ monotone | same as $\odot$, $\updownarrow$, $\leftrightarrow$ unimodal | | same as local | |
| Totally unimodal | $O(\lg max)$ | [Thm. 4] | same as local | |
| | $\geq \log_\phi max$ | [Lem. 7] | | |

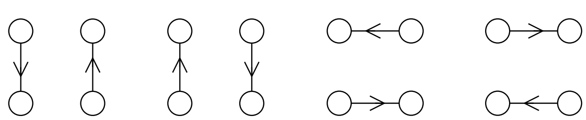
**Table 1.** Worst-case bounds on the number of probes required to maximize a function $f : \{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\} \to \mathbb{R}$. In the bounds, $max = \max\{m, n\}$ and $min = \min\{m, n\}$.

us to determine easily which combinations of properties imply which others. We then proceed to present nearly tight bounds for finding a local or global maximum for most combinations of properties: in $\updownarrow$ unimodal functions (Section 4), in $\updownarrow, \leftrightarrow$ unimodal functions (Section 5), and in totally unimodal functions (Section 6). Finally, in Section 7, we analyze a natural random probing strategy and show that it falls in between the last two strategies.

## 2 Forbidden Submatrix Partial Orders

In this section we show how several properties of real-valued functions on the $m \times n$ grid, or equivalently a real $m \times n$ matrix, can be expressed by finite forbidden substructures.

For any matrix property $\mathcal{P}$, we say that a matrix is *totally* $\mathcal{P}$ if every of its submatrices has property $\mathcal{P}$. In this section, we show how many total properties for matrices can be expressed as a finite set of constant-size partial orders that are forbidden to occur in any submatrix. This characterization of total properties gives an easy way to determine which combination of properties imply which other.



(a) (b) (c) (d)

**Fig. 1.** Forbidden $2 \times 2$ submatrices for total monotonicity. Arrows point to larger elements.

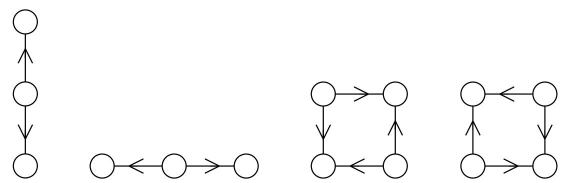
*Monotone.* Let $i(j)$ be the row index of the maximum in column $j$. A matrix is $+j$ *monotone* if $j \leq j'$ implies

$i(j) \leq i(j')$. A matrix is *totally $+j$ monotone* if every submatrix is $+j$-monotone. It can be shown [1] that it is sufficient to consider only $2 \times 2$ matrices. Thus to obtain the class of totally $+j$ monotone matrices, we just have to forbid the configuration shown in Figure 1(a).

Total monotonicity can be defined in all four directions: $+j$, $-j$, $-i$, and $+i$. The corresponding four forbidden configurations are shown in Figure 1(a–d).

A matrix is *totally monotone* if it forbids any one of these four configurations.

*$\updownarrow$ or $\leftrightarrow$ unimodal.* Note that $\updownarrow$ or $\leftrightarrow$ unimodality are total properties. Each property has a single forbidden configuration, as shown in Figure 2(a, b). Of course, $\updownarrow, \leftrightarrow$ unimodality is given by forbidding both of these configurations.



(a) (b) (c) (d)

**Fig. 2.** Forbidden submatrices for total unimodality. (a) $\updownarrow$ unimodality; (b) $\leftrightarrow$ unimodality.

*Totally unimodal.* A matrix is *totally unimodal* if every submatrix is $\odot$ unimodal, i.e., every submatrix has a unique local maximum. This property has four forbidden configurations, shown in Figure 2(a, b, c, d).

**Lemma 1.** *If a matrix is $\updownarrow, \leftrightarrow$ unimodal and totally $+j, -i$ monotone (or totally $-j, +i$ monotone), then it is totally unimodal.*

**Lemma 2.** *A matrix is totally unimodal if and only if it is $\updownarrow, \leftrightarrow$ unimodal, and every $2 \times 2$ submatrix is $\odot$ unimodal.*

**Corollary 1.** *If a matrix is totally unimodal, then it is $\odot, \updownarrow, \leftrightarrow$ unimodal.*

## 3 Elimination Lemmas

This section develops a battery of lemmas for guaranteeing that the solution we desire is not in a particular region, or more precisely, that at least one desired solution is in the remaining region. Different lemmas apply to different scenarios of assumptions made on the function, while one lemma is generic.

We consider the following more general (non-matrix) setting. A *discrete domain $D$* is a finite set along with a notion of adjacency (defined by a graph on the finite set). As mentioned, we mainly focus on the square grid domain $D = \{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\}$, primarily with 4-adjacency—two points are adjacent if their $\ell_1$ distance is 1—but several of our definitions and basic results apply more generally.

A *local maximum* of a function $f : D \to \mathbb{R}$ is a point $p$ of the domain $D$ such that all points adjacent to $p$ have strictly smaller $f$ values than $p$. In other words, a point is a local maximum if all incident edges (adjacencies) are *downhill* (in $f$). In this paper we assume that adjacent points have distinct $f$ values; otherwise, a constant function $f$ satisfies all (nonstrict) unimodality properties but is impossible to optimize in less than $|D|$ probes.

The following lemmas allow us to restrict the region in which we must search for a local maximum. In particular, given various configurations and/or unimodality assumptions on $f$, our goal is to identify elements that are effectively *eliminated* by a constant number of probes, in the sense that the remaining uneliminated region contains a local maximum.

The first lemma is useful in particular when the region in which we are searching disconnects into multiple components. In general, a *region* $R$ of a discrete domain $D$ is a subset of $D$. The *skin* of a region $R$ is the set of points in the domain $D$ that are not in $R$ but are adjacent to points in $R$.

**Lemma 3.** *For a function $f : D \to \mathbb{R}$, if the maximum $f$ value over a region $R$ of the domain $D$ is larger than the $f$ values of all points on the skin of $R$, then $R$ contains a local maximum of $f$.*

The next lemma shows that, in the $\updownarrow$-unimodal case, whenever an algorithm makes a probe it can probe a vertically adjacent point (losing at most a factor of 2 in probe count) and eliminate either the top or bottom "half" of the column, depending on which of the two points has a larger $f$ value. Define $(\leq i, j) = \{(i', j) \mid i' \leq i\}$ and similarly for $(?i, j)$ and $(i, ?j)$ for $? \in \{\leq, \geq, <, >\}$.

**Lemma 4.** *Suppose $f : \{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\} \to \mathbb{R}$ is $\updownarrow$ unimodal and suppose that region $R$ contains a local maximum. If $f(i, j) > f(i + 1, j)$, then $R \setminus (> i, j)$ contains a local maximum. Similarly, if $f(i, j) < f(i + 1, j)$, then $R \setminus (\leq i, j)$ contains a local maximum.*

The next lemma shows an analogous result for $\odot, \updownarrow, \leftrightarrow$ unimodality, except that the constant factor loss is now at most 3, and the eliminated elements are nearly an entire quadrant. (The entire quadrant can be eliminated at a cost of at most a factor of 5.) Define $(\leq i, \leq j) = \{(i', j') \mid i' \leq i, j' \leq j\}$ and similarly for $(?i, ¿j)$ for $?, ¿ \in \{\leq, \geq, <, >\}$.

**Lemma 5.** *Suppose $f : \{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\} \to \mathbb{R}$ is $\odot, \updownarrow, \leftrightarrow$ unimodal and suppose that region $R$ contains a local maximum. Consider a point $(i, j)$ in $R$. If $f(i, j) > f(i+1, j)$ and $f(i, j) > f(i, j+1)$, then $R \setminus [(\geq i, \geq j) - (i, j)]$ contains a local maximum. (Unless $(i, j)$ is also a local maximum, i.e., we also have $f(i, j) > f(i - 1, j)$ and $f(i, j) > f(i, j - 1)$, even $R \setminus (\geq i, \geq j)$ contains a local maximum.) Similarly, if $f(i, j) > f(i + 1, j)$ and $f(i, j) < f(i, j + 1)$, then $R \setminus (\geq i, \leq j)$ contains a local maximum; if $f(i, j) < f(i + 1, j)$ and $f(i, j) > f(i, j+1)$, then $R \setminus (\leq i, \geq j)$ contains a local maximum; and if $f(i, j) < f(i+1, j)$ and $f(i, j) < f(i, j + 1)$, then $R \setminus (\leq i, \leq j)$ contains a local maximum.*

Finally, we prove a more powerful quadrant elimination lemma for totally unimodal functions, where we can compare to nonadjacent points because total unimodality allows us to consider induced submatrices.

**Lemma 6.** *Suppose $f : \{1,2,\ldots,m\} \times \{1,2,\ldots,n\} \to \mathbb{R}$ is totally unimodal and suppose that region $R$ contains a local maximum. Consider two points $(i,j)$ and $(i',j')$ with $i < i'$ and $j < j'$, suppose that $R$ is already disjoint of the cornerless quadrants $(\leq i, \leq j) - (i,j)$ and $(\geq i', \geq j') - (i',j')$, and suppose that $f(i-1,j) < f(i,j)$ and $f(i,j-1) < f(i,j)$, If $f(i',j) > f(i,j)$ and $f(i',j) > f(i',j')$, then $R \setminus (\leq i, \geq j')$ contains a local maximum. If $f(i,j') > f(i,j)$ and $f(i,j') > f(i',j')$, then $R \setminus (\geq i', \leq j)$ contains a local maximum. If neither of these conditions hold, then $R \setminus ((\leq i, \geq j') \cup (\geq i', \leq j))$ contains a local maximum.*

Before proceeding to more difficult upper and lower bounds, we prove a simple logarithmic lower bound in the most specific case of totally unimodal functions:

**Lemma 7.** *Any comparison-based algorithm for finding a local maximum in a totally unimodal function must make at least $\log_\phi \max\{m,n\} - O(1)$ probes in the worst case.*

# 4 $\updownarrow$ Unimodal

## 4.1 Local Optimization

**Lemma 8.** *There is an algorithm that, given a $\updownarrow$ unimodal $m \times n$ matrix, finds a local optimum after $\leq \log_\phi n \log_\phi m + O(\log n)$ probes.*

**Theorem 1.** *For every algorithm that correctly finds a local optimum in an $m \times n$ $\updownarrow$ unimodal matrix, there is an adversary that (a) generates a $\updownarrow$ unimodal function with a unique local optimum, and (b) forces the algorithm to make $\geq \frac{1}{4} \lg m \lg n - O(\lg m \lg \lg n)$ probes if $m \leq n$, and $\geq \frac{1}{4} \lg^2 n - O(lgn \lg \lg n)$ if $m > n$.*

*Proof.* The adversary gives the algorithm extra information, which can only help. Whenever the algorithm probes the value at a particular point $(i,j)$, the adversary reveals not only that value, but also the slope of that value in that column, i.e., whether the mode in that column $j$ is above or below that point $(i,j)$. Furthermore, if the mode of column $j$ is above the probe point $(i,j)$, then the adversary reveals all values in the column $j$ below the point $(i,j)$; symmetrically, if the mode is below the probe point, the adversary reveals all values above the point in its column. If the algorithm discovers the mode of column $j$, the adversary reveals all values in the column $j$. Thus we maintain the invariant that every column that is not totally revealed has some revealed values in the topmost few rows, some revealed values in the bottommost few rows, and the algorithm knows that the mode of the column is somewhere in between.

If the unrevealed region ever becomes disconnected, the adversary reveals all values in all connected components except the largest connected component. Thus we maintain the invariant that the unrevealed region is connected. We also maintain the invariant that the algorithm cannot discover the unique local optimum until every value has been revealed. Together these two invariants make the goal of the algorithm to disconnect the unrevealed region; otherwise, the algorithm must make at least one probe per column, for a total of at least $n$ probes.

The main task of the adversary is to decide whether a probe point is above or below the mode of that column, and then to choose the revealed values below or above the probe point. The adversary bases its decision on matching the "nearest" previous decision, according to the $\ell_1$ distance function. Naturally, the distance between a point $(i, j)$ and the top horizontal wall is $i$, and the distance to the bottom wall is $m + 1 - i$.

Suppose that the algorithm probes the point $(i, j)$. If point $(i, j)$ is closer to a horizontal wall than every revealed point, then the adversary reveals all values in column $j$ between $(i, j)$ and the nearest wall, specifying that the mode is in the other direction. Otherwise, the adversary specifies $(i, j)$ to be above or below the mode in its column $j$ according to whether the revealed point $(i^*, j^*)$ nearest to $(i, j)$ is above or below the mode in its column $j^*$. Then the adversary reveals all unrevealed values starting from $(i, j)$ in the opposite direction to the mode in column $j$. (In the special case described below that the algorithm discovers the mode among these revealed values, the specification that the mode is above or below $(i, j)$ is false; in this case the adversary reveals all values in column $j$.)

The adversary chooses the revealed values as follows. Suppose that the algorithm probes $(i, j)$ and say that the adversary decides that probe point $(i, j)$ is below the mode in its column $j$. If the to-be-revealed points keep the unrevealed region connected, then the adversary repeatedly reveals that the bottommost unrevealed value in column $j$ is one more than the largest previously revealed value, until reaching point $(i, j)$. In this way the revealed values increase in an integer sequence from the bottommost unrevealed value to $(i, j)$. Equivalently, the adversary reveals every unrevealed point $(i', j)$ below $(i, j)$ in column $j$ to have value $m-d$ more than the largest previously revealed value, where $d = i-i'$ is the Manhattan distance between the unrevealed point $(i', j)$ and the probe point $(i, j)$.

On the other hand, if the to-be-revealed points disconnect the unrevealed region, then we either keep unrevealed the component left of column $j$ or the component right of column $j$, whichever has the largest number of unrevealed columns. Assume the component left of column $j$ is to be kept unrevealed, and let $j'$ be the rightmost unrevealed column in the matrix. We reveal the entries in the columns from column $j'$ to column $j$ as follows: when revealing the entries of column $j''$, $j' \geq j'' \geq j$, we identify an entry $(i'', j'')$ adjacent to an unrevealed entry in column $j''-1$. We set $(i'', j'')$ to be the mode of column $j''$, and reveal all entries in that column, by repeatedly revealing the topmost entry with one more than the previously revealed value until $(i'' - 1, j'')$ is revealed, then repeatedly revealing the bottommost entry with one more than the previously revealed value until $(i'', j'')$ is revealed, then proceed to reveal the entries of column $j'' - 1$ in the same manner, until column $j$ is completely revealed. This strategy ensures that whenever a point is revealed, it is connected to a yet unrevealed point, and so there is an increasing path from any entry in the table to the unique local optimum which is the last value to be revealed. Thus we obtain:

**Lemma 9.** *The only point to become a local maximum according to the adversary is the mode of the final column to become completely revealed.*

**Lemma 10.** *The algorithm must make* $\min\{n, \lg m\}$ *probes before the unrevealed region first disconnects into multiple connected components.*

**Lemma 11.** *The nearest point or horizontal wall to a point* $(i, j)$ *is in a column* $j'$ *such that* $|j - j'| \leq m$.

Finally we conclude the proof of Theorem 1. Consider an algorithm that makes fewer than $\lg n \lg m$ probes. As mentioned above, the algorithm must disconnect the unrevealed region or else it is doomed to make at least $n$ probes. Lemma 10 says that the algorithm must make at least $\min\{n, \lg m\}$ probes for the first disconnection. Consider the final probe that caused the disconnection. By the pigeon-hole principle, the $(\lg n \lg m)m$ consecutive columns including and to the right of this final probe must have a gap of at least $m$ consecutive empty columns, because there are at most $\lg n \lg m$ probes total. We remove columns starting from the final probe up to but not including this gap of $m$ consecutive empty columns. Similarly, we remove at most $(\lg n \lg m)m$ columns to the left of the final probe up to but not including a gap of $m$ consecutive empty columns. Thus we obtain two subproblems (one left and one right) that by Lemma 11 act completely independently from each other and from the probes causing the disconnection, as far as probes made so far. We recursively consider the subproblem corresponding to the larger connected component that remains. This recursive subproblem is a rectangle with $m$ rows and $n' \geq \lfloor n/2 \rfloor - (\lg n \lg m)m$ columns. The recursive subproblem may have already been probed, but we can consider such probes as happening after this subproblem. Thus the recursion applies until $n'/2 < (\lg n \lg m)m$.

Therefore we obtain the lower bound of $\min\{n', \lg m\}$ probes, where $n' \geq 2(\lg n \lg m)m$, at each of $\lg(n/(2(\lg n \lg m)m))$ levels of recursion. In total we obtain a lower bound of $(\lg m)(\lg(n/m) - 1 - \lg \lg n - \lg \lg m) \geq \lg m \lg(n/m) - O(\lg m \lg \lg n)$. If $m \leq \sqrt{n}$, then the lower bound is $\geq \frac{1}{2} \lg m \lg n - O(\lg m \lg \lg n)$. If $m > \sqrt{n}$, we perform the same argument on a submatrix with $m' = \sqrt{n}$ rows. The lower bound then becomes $\geq \lg m' \lg(n/m') - O(\lg m' \lg \lg n) \geq \frac{1}{4} \lg^2 n - O(lgn \lg \lg n)$. In particular, if $\sqrt{n} \leq m \leq n$, we obtain the lower bound $\geq \frac{1}{4} \lg m \lg n - O(\lg m \lg \lg n)$. $\square$

### 4.2 Global Optimization

**Lemma 12.** *There is an algorithm that, given a* $\updownarrow$ *unimodal* $m \times n$ *matrix, finds its global optimum after at most* $n \log_\phi m + O(n)$ *probes.*

**Lemma 13.** *Any algorithm that, given a* $\updownarrow$ *unimodal* $m \times n$ *matrix finds its global optimum must perform at least* $n \log_\phi m - O(n)$ *probes.*

## 5 $\updownarrow, \leftrightarrow$ Unimodal

**Theorem 2.** *There is an algorithm that finds a local optimum in a $\updownarrow, \leftrightarrow$ unimodal $m \times n$ matrix after at most $(3/\lg\phi)\lg^2 min + O(\lg max)$ probes.*

*Proof.* Assume without loss of generality that $m \leq n$. First find the maximum element on row $m/2$, among elements in columns $in/m$, $i = 1, \ldots, m$, in $\lg_\phi m$ time. (Ratios are implicitly rounded to integers, affecting only lower-order terms.) This finds two elements $jn/m$ and $(j+1)n/m$ on columns separated by $n/m$ elements. We can now eliminate from the search one of the two quadrants left of $(m/2, jn/m)$ and one of the two quadrants right of $(m/2, (j+1)n/m)$. Then find the maximum on columns $jn/m$ and $(j+1)n/m$ using Fibonacci search, and evaluate the right and left neighbors of those two maxima. We now know a local max is either (I) to the left of column $jn/m$, (II) between columns $jn/m$ and $(j+1)n/m$ or (III) to the right of column $(j+1)n/m$. Since one quadrant has been eliminated to the left of column $jn/m$ and one quadrant has been eliminated to the right of column $(j+1)n/m$, the size of the submatrix to recurse in is $(m/2) \times n$ in cases (I) and (III), or $m \times (n/m)$ in case (II). Thus, cases (I) and (III) can only happen $\lg m$ times, and case (II) can only happen $\lg n/\lg m$ times. Each step performs $3\lg_\phi m + O(1)$ probes, so the total number of probes is $(3/\lg\phi)(\lg^2 m + \lg n) + O(1)(\lg m + \lg n/\lg m)$. □

**Theorem 3.** *For every algorithm that correctly finds a local maximum in an $m \times n$ $\updownarrow, \leftrightarrow$ unimodal, totally $+j, +i$ monotone matrix, there is an adversary that (a) generates such a function with a unique local maximum, and (b) forces the algorithm to make $\frac{1}{4}\lg^2 min - O(\lg min \lg\lg min)$ probes.*

**Lemma 14.** *Any algorithm that, given a $\updownarrow, \leftrightarrow$ unimodal $m \times n$ matrix finds its global maximum must perform at least $\min\{n, m\}$ probes.*

## 6 Totally Unimodal

**Theorem 4.** *There is an algorithm that, given a totally unimodal $m \times n$ matrix, finds its global maximum after $O(\lg n + \lg m)$ probes.*

*Proof.* The algorithm performs successive probes and eliminates regions of the matrix known not to contain the local maximum. At every step of the algorithm, the unrevealed region will be a cross inside a submatrix, i.e., the algorithm maintains four indices $i_1$, $i_2$, $j_1$, $j_2$, with $i_1 + 1 < i_2$ and $j_1 + 1 < j_2$ such that the unique local maximum is known not to be in the quadrants $(\leq i_1, \leq j_1)$, $(\geq i_2, \leq j_1)$, $(\leq i_1, \geq j_2)$, $(\geq i_2, \geq j_2)$. Furthermore, we maintain the invariant that the apex of each of those four quadrants is the maximum value in the quadrant, e.g. for the first quadrant, that $f(i_1 - 1, j_1) < f(i_1, j_1)$ and $f(i_1, j_1 - 1) < f(i_1, j_1)$. We call this the *apex invariant.* The rectangular area with corners $(i_k + 1, j_l + 1)$ for $k, l \in \{1, 2\}$ is called the *center* of the cross, and is surrounded by four *legs*. We will sometimes refer to T shapes or L shapes instead of the cross, those are just crosses for which one or two of the legs is empty, respectively.

The algorithm then performs a constant number of probes which will reduce the unrevealed area of the matrix by a constant factor. This will be done in one of four ways: (a) by reducing the width of the cross ($i_2 - i_1$ and $j_2 - j_1$) by half, (b) by transforming the cross into an $L$ shaped region which is a constant fraction smaller than the original cross, but whose center might not be contained in the original center, or (c) by transforming the cross into an $L$ shaped region whose center (which might not be contained in the original center) has area at least one quarter of the total unrevealed area.

Let $i_m = \lfloor (i_1 + i_2)/2 \rfloor$ and $j_m = \lfloor (j_1 + j_2)/2 \rfloor$. We first probe $(i_m, j_m)$ and its four neighbors. Either $(i_m, j_m)$ is the local maximum, or by Lemma 5, one of its four quadrants can be eliminated. Assume that the eliminated quadrant is $(\geq i_m, \geq j_m)$, the other cases are handled symmetrically. Next, we apply the Lemma 6 on entries $(i_1, j_1)$ and $(i_m, j_m)$. For this, we probe the entries $(i_1, j_m)$ and $(i_m, j_1)$. Assume that $f(i_1, j_m) > f(i_m, j_1)$, the other case is handled symmetrically. Then the quadrant $(\geq i_m, \leq j_1)$ can be eliminated from the search. Furthermore, if the apex $(i_m, j_1)$ is the minimum of that quadrant, then the quadrant satisfies the apex invariant. If it does not, then we still know that $f(i_m, j_1)$ is smaller than one of $f(i_1, j_1)$ and $f(i_m, j_m)$, otherwise rows $i_1$ and $i_m$ and columns $j_1$ and $j_m$ form a $2 \times 2$ matrix with 2 local optima. If $f(i_m, j_1) < f(i_1, j_1)$, then $f(i_m + 1, j_1) < f(i_m, j_1)$ but $f(i_m, j_1 - 1) > f(i_m, j_1)$ since the apex invariant is not satisfied. This implies $f(i_m, j_1 + 1) < f(i_m, j_1)$ and applying Lemma 5, quadrant $(\geq i_m, \geq j_1)$ can be eliminated. Those two eliminated quadrants together remove all rows $\geq i_m$. Likewise, if $f(i_m, j_1) < f(i_m, j_m)$, then the quadrant $(\leq i_m, \leq j_1)$ can be eliminated and so all columns $\leq j_1$ can be removed.

At this point, we have eliminated at least two quadrants, and the left, right and bottom legs have had their width divided by 2, and the bottom or left leg might have been eliminated. We now probe the four neighbors of $(i_1, j_m)$ (whose value is know from the previous step), and apply Lemma 5 to eliminate one of its four quadrants. We now have four cases to consider.

If quadrant $(\leq i_1, \geq j_m)$ is eliminated, then we have achieved goal (a): we have a new cross of half the width (where one of the legs may have been eliminated), so the area of all four legs is multiplied by $\frac{1}{2}$, and the area of the center is multiplied by $\frac{3}{4}$.

If quadrant $(\leq i_1, \leq j_m)$ is eliminated, then we apply Lemma 6 to quadrants $(\leq i_1, \leq j_m - 1)$ and $(\geq i_m, \geq j_m)$. This eliminates either $(\leq i_1, \geq j_m - 1)$ in which case all rows $\leq i_1$ can be removed, or $(\geq i_m, \leq j_m)$, in which case all rows $\geq i_m$ can be removed. In both cases, the unrevealed region is a T shape, a cross with one leg cut, and all legs have had their width multiplied by a factor $\frac{1}{2}$, so we have again achieved goal (a).

If quadrant $(\geq i_1, \geq j_m)$ is eliminated, then all columns $\geq j_2$ are eliminated. We then apply Lemma 6 to quadrants $(\leq i_1, \leq j_1)$ and $(\geq i_1 + 1, \geq j_m)$. This further eliminates either $(\leq i_1, \geq j_m)$ or $(\geq i_1 + 1, \leq j_1)$. In the first case, all rows $\geq j_m$ can be removed and we obtain a T shaped unrevealed region, which is a cross with one leg cut off, and all legs have had their width multiplied by a factor

$\frac{1}{2}$, so we have again achieved goal (a). In the second case, the unrevealed region becomes L shaped: the left leg has been removed with quadrant $(\geq i_1+1, \leq j_1)$, the right leg was already removed, so what remains is the top leg, the center of the cross divided in two vertically and the bottom leg divided by two vertically. So unless the top leg contained more than half of the area unrevealed at the beginning of this step, we have eliminated at least one quarter of the total unrevealed area, and so we have achieved goal (b). Otherwise, if the top leg contained more than half of the area unrevealed at the beginning of this step, then the center of the new L shape contains more than half of the top leg, and so more than one quarter of the total unrevealed area, reaching goal (c).

Finally, if quadrant $(\geq i_1, \leq j_m)$ is eliminated, then all columns $\geq j_1$ are eliminated. We then apply Lemma 6 to quadrants $(\geq i_1+1, \leq j_m)$ and $(\leq i_1, \geq j_2)$. This further eliminates either $(\leq i_1, \leq j_m)$ or $(\geq i_1+1, \geq j_2)$. In the first case, all rows $\leq j_m$ can be removed and we obtain an L shaped unrevealed region, which is a cross with two legs cut off, and all legs have had their width multiplied by a factor $\frac{1}{2}$, so we have again achieved goal (a). In the second case, the unrevealed region becomes L shaped, containing just the top leg and a quarter of the original center. As in the previous case, if the top leg contained less than half of the area unrevealed at the beginning of this step, we have eliminated at least one quarter of the total unrevealed area, and so we have achieved goal (b). Otherwise, if the top leg contained more than half of the area unrevealed at the beginning of this step, then the center of the new L shape contains more than half of the top leg, and so more than one quarter of the total unrevealed area, and we have reached goal (c).

To conclude, note that every step performs a constant number of probes. After each elimination step, if goals (a) or (b) are reached, then one quarter of the unrevealed area has been eliminated. If goal (c) is attained, then the first set of probes of the next step eliminates one quarter of the area of the center of the cross, which is in this case at least one sixteenth of the total unrevealed area. So in all cases, two consecutive steps eliminate a constant fraction of the area, so the total number of steps is $O(\lg(mn))$. $\square$

# 7 Random Probing Algorithm

In this section we analyze a natural family of *uniform probing* strategies for finding a local optimum in an $\updownarrow, \leftrightarrow$ unimodal function. We specify and analyze the strategy only in the case of totally unimodal functions, where of course the algorithm finds the global maximum. Our lower bound on the strategy's performance also applies to any generalization of this algorithm to $\updownarrow, \leftrightarrow$ unimodal functions. Our upper bound is specific to totally unimodal functions.

The uniform probing algorithm for totally unimodal functions works as follows. Initially, we set the region $R$ to the entire domain $D = \{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\}$ of the function $f$. At each step, the algorithm chooses a point $(i, j)$ uniformly at random from the remaining region $R$. Then the algorithm makes three samples to eliminate a quadrant except for its corner $(i, j)$, according to

Lemma 5. If the remaining region $R'$ contains just one point, then it is the unique maximum; otherwise the algorithm continues.

**Theorem 5.** *Uniform probing makes $\Theta(\ln^2 max)$ expected probes in a totally unimodal function.*

## 8 Conclusion

We expect many of our results to generalize to several other scenarios. In particular, we expect similar bounds in $d$ dimensions, at least in the case of an $n \times n \times \cdots \times n$ matrix, where logarithmic bounds remain logarithmic and squared-logarithmic bounds grow to $\log^d$. We also believe that our results generalize to local maxima defined in terms of size-8 neighborhoods instead of the 4-neighborhoods we use. For example, as with 4-neighborhoods, total unimodality with 8-neighborhoods can be characterized as forbidding a finite set of partial orders on constant-size submatrices. More generally, it would be interesting to characterize the complexities achievable for all possible forbidden submatrices. In our work, we have seen how combining various unimodality conditions with the total monotonicity conditions of [1] yields surprising results. In particular, combining $\updownarrow, \leftrightarrow$ unimodality with total $(+i, -j)$ or $(-i, +j)$ monotonicity implies total unimodality (Lemma 1), and so an $O(\log n)$ optimization algorithm, while combining $\updownarrow, \leftrightarrow$ unimodality with total $(+i, +j)$ or $(-i, -j)$ monotonicity has an $\Omega(\log^2 min)$ lower bound (Theorem 3).

## References


1. A. Aggarwal, M. M. Klawe, S. Moran, P. W. Shor, and R. Wilber. Geometric applications of a matrix-searching algorithm. *Algorithmica*, 2:195–208, 1987.
2. I. Althöfer and K.-U. Koschnick. On the deterministic complexity of searching local maxima. *Discrete Appl. Math.*, 43(2):111–113, 1993.
3. H. Edelsbrunner. Computing the extreme distances between two convex polygons. *J. Algorithms*, 6:213–224, 1985.
4. Z. Galil and K. Park. Dynamic programming with convexity, concavity and sparsity. *Theoret. Computer Sci.*, 92(1):49–76, Jan. 1992.
5. B. Gärtner, W. D. Morris, and L. Rüst. Unique sink orientations of grids. In *Proc. 11th Internat. IPCO Conf. on Integer Prog. and Combinat. Opt.*, volume 3509 of *Lecture Notes in Computer Science*, pages 210–224, Berlin, Germany, June 2005.
6. J. Kiefer. Sequential minimax search for a maximum. *Proc. Amer. Math. Soc.*, 4:502–506, 1953.
7. D. C. Llewellyn, C. Tovey, and M. Trick. Local optimization on graphs. *Discrete Appl. Math.*, 23(2):157–178, 1989.
8. D. C. Llewellyn and C. A. Tovey. Dividing and conquering the square. *Discrete Appl. Math.*, 43(2):131–153, 1993.
9. A. Mityagin. On the complexity of finding a local maximum of functions on discrete planar subsets. *Theoret. Computer Sci.*, 310(1–3):355–363, Jan. 2004.
10. C. A. Tovey. Local improvement on discrete structures. In *Local Search in Combinatorial Optimization*, pages 57–89. John Wiley and Sons, 1997.