Curvature Batching Gives Single Exponential Integer Quadratic Programming

Cinar Ari **Robert Hildebrand**

Grado Department of Industrial and Systems Engineering, Virginia Tech

INTRODUCTION

Integer Quadratic Programming (IQP) is the problem:

$$\min\{x^T Qx + c^T x \ : \ Ax \le b, \ \ x \in \mathbb{Z}^n\},$$

where $Q \in \mathbb{Z}^{n\times n}$ is symmetric and possibly **indefinite**. IQP arises in:

- **Operations research:** network flow with quadratic arc costs, production planning, lattice decoding.
- **Parameterized complexity:** Optimal Linear Arrangement, Exact Crossing Number, Densest k-Subgraph.

The complexity depends on Q:

- **Convex** ($Q \succeq 0$): Lenstra-type algorithms apply.
- **Concave** ($Q \preceq 0$): minimizers at vertices of the integer hull.
- **Indefinite:** only recently shown to lie in NP (Del Pia, Dey, & Molinaro, 2017).

Prior Work

ILP is NP-hard but FPT for fixed n: Lenstra (1983) gave $2^{O(n^3)} \cdot \mathrm{poly}(\varphi)$; Reis & Rothvoss (2023) achieved $(\log n)^{O(n)} \cdot \mathrm{poly}(\varphi)$.

Nonlinear: finding an integer root of a degree-4 polynomial is undecidable, solving a system of quadratic equations is undecidable.

IQP: Lokshtanov (2017) and Zemmer (2017) showed IQP is FPT parameterized by n and L, but **neither gave an explicit running time**. We show Lokshtanov's algorithm is **doubly exponential**: $(nL)^{O(n^2 \cdot 3^n)} \cdot \mathrm{poly}(\varphi)$.

MAIN RESULT

Theorem (Curvature Batch Bound). IQP can be solved in time

$$\left(n\, L_A^n\, \Delta(A)\, L_Q\right)^{O(n)} \cdot \mathrm{poly}(\varphi) \text{ which is } (nL)^{O(n^2)} \cdot \mathrm{poly}(\varphi).$$

$L_A = \|A\|_{\max}$, $L_Q = \|Q\|_{\max}$, $L = \max(L_A, L_Q)$, $\Delta(A) =$ max absolute subdeterminant of A. This is the **first single-exponential** algorithm for IQP.

Corollary (TU Constraints). If A is totally unimodular: $(nL_Q)^{O(n)} \cdot \mathrm{poly}(\varphi)$.

Complexity Comparison

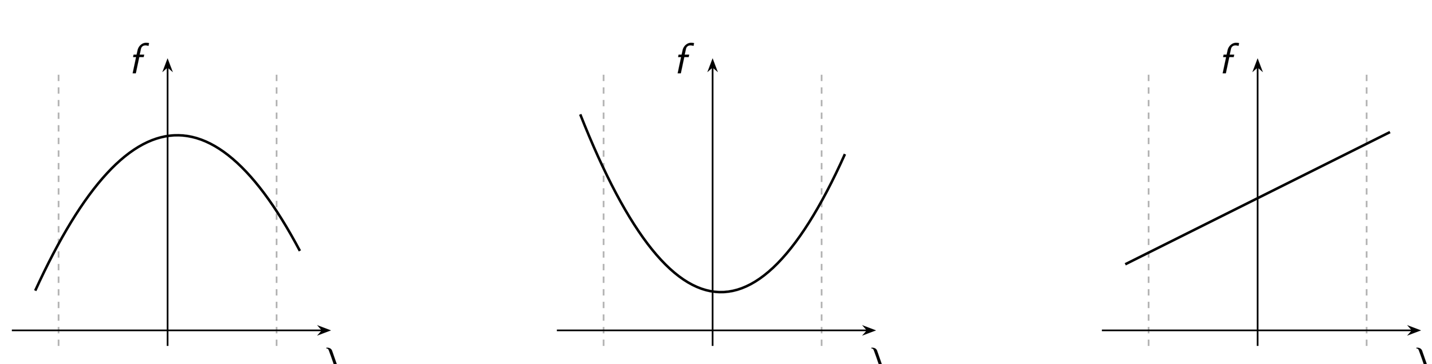
Algorithm	Complexity
Lokshtanov (2017)	$(nL)^{O(n^2 \cdot 3^n)} \cdot \mathrm{poly}(\varphi)$
Curvature Batch (ours)	$\boldsymbol{(nL)^{O(n^2)} \cdot \mathrm{poly}(\varphi)}$
ILP — Reis & Rothvoss (2023)	$(\log n)^{O(n)} \cdot \mathrm{poly}(\varphi)$

APPROACH

Curvature Along Fibers

Along $y \in \ker(C)$, the objective on fiber $x + \lambda y$:

$$f(x+\lambda y) = \underbrace{\lambda^2(y^T Qy)}_{\text{curvature}} + \lambda(2y^T Qx + c^T y) + f(x)$$



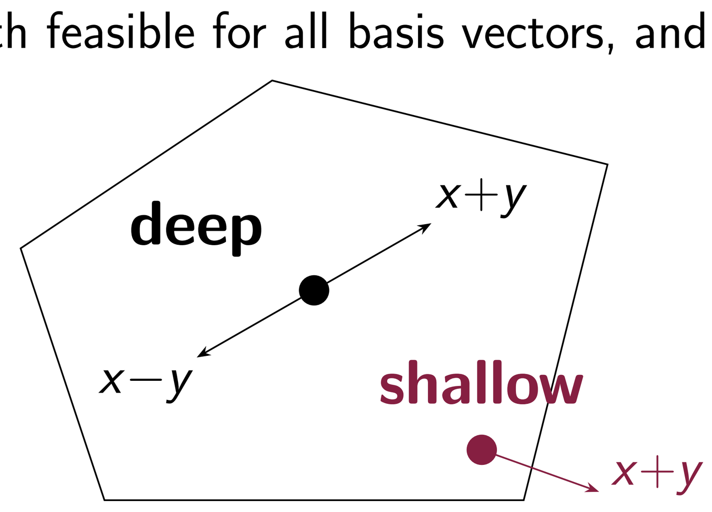
Concave $y^T Qy < 0$ opt. near boundary

Convex $y^T Qy > 0$ opt. in interior

Flat $y^T Qy = 0$ reduces to ILP

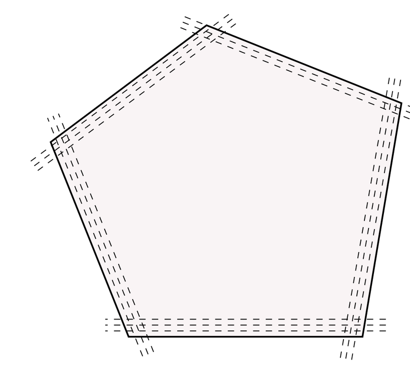
Deep and Shallow Solutions

Lokshtanov's algorithm separates equalities $Cx = d$, computes an integer basis $\{y_1, \ldots, y_r\}$ for $\ker(C)$ with $\|y_i\|_\infty \le \Delta(C)^2$, and branches. A solution x is **deep** if $x \pm y_i$ are both feasible for all basis vectors, and **shallow** otherwise.



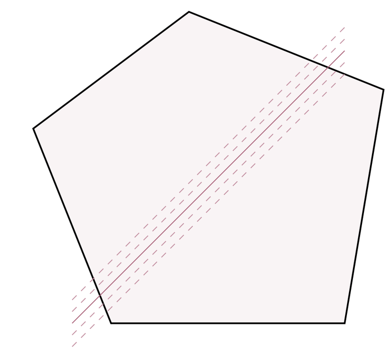
LOKSHTANOV'S ALGORITHM

Case 1: Constraint



If x^* is **shallow**: branch over $a_j^T x = b_j'$ for values b_j' in $\{b_j - nL_A\Delta, \ldots, b_j\}$.

Case 2: Gradient



If x^* is **deep**: optimality forces $|2y_i^T Qx^* + c^T y_i| \le y_i^T Qy_i$. Branch over integer z.

Case 3: Flat. If $y_i^T Q \in \mathrm{rowspace}(C)$ for all i, Q vanishes on $\ker(C)$ and the problem is an **ILP**. Cases 1 & 2 each reduce $\dim(\ker(C))$ by one; depth $\le n$.

Why Doubly Exponential?

Adding a row r to C:

$$\left|\, C \,\right| \rightarrow \left|\begin{array}{c} C \\ \hdashline r \end{array}\right| = C'$$

$|\det(M)| \le n \cdot \|r\|_\infty \cdot \Delta(C)$.

- **Constraint** ($r^T = a_j^T$): $\Delta(C') \le nL_A\Delta$ - cheap (linear).
- **Gradient** ($r^T = 2y^T Q$, $\|y\|_\infty \le \Delta^2$): $\Delta(C') \le 2n^2 L_Q \Delta^3$ - **cubing**.

With n gradient steps, subdeterminant grows as Δ^{3^n}: complexity $(nL)^{O(n^2 \cdot 3^n)}$ - **doubly exponential**.

OUR IMPROVEMENTS

1. Adjugate Basis

Lemma. Let $C \in \mathbb{Z}^{k\times n}$, $\mathrm{rank}(C) = k < n$. There exist linearly independent integer vectors $y_1, \ldots, y_{n-k} \in \ker(C)$ with $\|y_i\|_\infty \le \Delta(C)$.

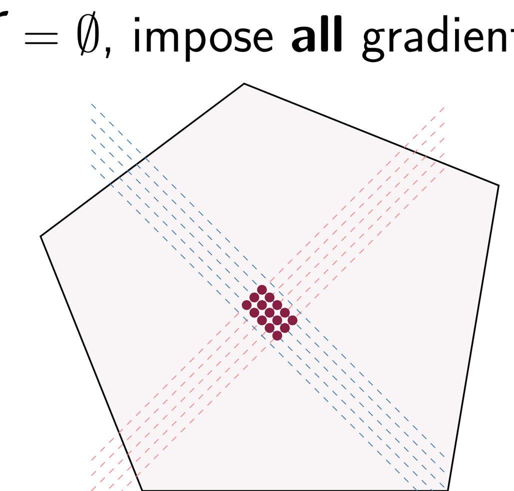
Improves basis bound from Δ^2 to Δ, reducing gradient growth from $\Delta \to \Delta^3$ to $\Delta \to \Delta^2$.

2. Curvature Batching

Classify basis directions by curvature:

- $\mathcal{N} = \{i : y_i^T Qy_i < 0\}$ — **negative:** forces shallowness.
- $\mathcal{G} = \{i : y_i^T Qy_i \ge 0, \ y_i^T Q \notin \mathrm{rowspace}(C)\}$ — **non-negative:** gradient candidates.
- $\mathcal{F} = \{i : y_i^T Q \in \mathrm{rowspace}(C)\}$ — **flat:** already resolved.

Batching idea: When $\mathcal{N} = \emptyset$, impose **all** gradient constraints at once.



Each dot is a subproblem

After the batch, Q **vanishes on the residual kernel**, so the remaining problem is an ILP. The batch is performed **at most once** per root-to-leaf path.

Phase 1	**Phase 2**	**Phase 3**
Constraint steps	Single batch	ILP leaf
$\Delta(C) \le \Delta(A)$	Δ may inflate	depends only on n

The Δ-inflation from the batch is **irrelevant**: the ILP solver depends only on n, not Δ.

Total: $\left(n\, L_A^n\, \Delta(A)\, L_Q\right)^{O(n)} \cdot \mathrm{poly}(\varphi) = (nL)^{O(n^2)} \cdot \mathrm{poly}(\varphi)$.

FUTURE DIRECTIONS

- **Mixed-integer generalization:** Extend the curvature batch framework to the mixed-integer setting.
- **Polynomial in n alone?** It remains a major open problem whether IQP is polynomial-time solvable for any fixed n, removing dependence on the coefficient size L entirely.