Fast Convolutions for Near-Convex Sequences

Cornelius Brand
Institute of Logic and Computation, Vienna University of Technology, Austria

Alexandra Lassota
Max Planck Institute for Informatics, SIC, Saarbrücken, Germany

Abstract

We develop algorithms for $(\min,+)$-Convolution and related convolution problems such as Super Additivity Testing, Convolution 3-Sum and Minimum Consecutive Subsums which use the degree of convexity of the instance as a parameter. Assuming the min-plus conjecture (Künnemann-Paturi-Schneider, ICALP'17 and Cygan et al., ICALP'17), our results interpolate in an optimal manner between fully convex instances, which can be solved in near-linear time using Legendre transformations, and general non-convex sequences, where the trivial quadratic-time algorithm is conjectured to be best possible, up to subpolynomial factors.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases $(\min,+)$-convolution, fine-grained complexity, convex sequences

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2023.16

Funding The research leading to this article was partially carried out at EPFL, Lausanne, Switzerland, funded by the Swiss National Science Foundation project Complexity of integer Programming (207365).
Cornelius Brand: Supported by FWF grant Y1329 in the START-Program (ParAI).
Alexandra Lassota: Funded by the Swiss National Science Foundation project Complexity of integer Programming (207365).

1 Introduction

The $(\min,+)$-convolution, also called *tropical convolution*, is an operation on sequences that forms the analogue of polynomial multiplication in the tropical semiring: additions are replaced by taking minima, whereas multiplications become additions.

Let $\bar{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$. Formally, the $(\min,+)$-convolution $a * b$ of two sequences $a = (a_0, \ldots, a_n) \in \bar{\mathbb{R}}^{n+1}$ and $b = (b_0, \ldots, b_n) \in \bar{\mathbb{R}}^{n+1}$ is defined as the sequence $c = (c_0, \ldots, c_{2n}) \in \bar{\mathbb{R}}^{2n+1}$ where

$$c_k = \min_{i+j=k} a_i + b_j \text{ for } k = 0, \ldots, 2n. \tag{1}$$

For computational purposes, the inputs are restricted to sequences over $\bar{\mathbb{Z}} = \mathbb{Z} \cup \{\infty\}$. Associated with this operation is the following problem.

$(\min,+)$-Convolution	
Input:	Two sequences $a, b \in \bar{\mathbb{Z}}^{n+1}$
Output:	All entries of $c = a * b$

The trivial algorithm computing $a * b$ given a and b takes a quadratic number of steps in n. Over the years, great efforts have been directed at improving substantially over this trivial bound to no avail. Consequently, it has been conjectured that this is essentially optimal up to subpolynomial factors:

▶ **Conjecture 1** ([19, 25]). *There is no algorithm solving* $(\min,+)$-Convolution *in time* $n^{2-\varepsilon} \cdot \mathrm{polylog}(d)$ *on integral inputs* a, b *of maximal absolute value* d*, for any constant* $\varepsilon > 0$.

While polynomial improvements seem out of reach, there are faster algorithms for the problem, shaving off subpolynomial factors [10, 31, 15].

However, some algorithms can solve the problem in truly subquadratic time, but do so only on restricted input classes. Namely, Chan and Lewenstein [14] give an $O(n^{1.87})$-time algorithm for inputs consisting of monotone sequences with numbers bounded by $O(n)$. This was recently improved to $\tilde{O}(n^{1.5})$ by a break-through result by Chi et al. [16]. Further, there is an algorithm that solves $(\min,+)$-Convolution on *convex* input sequences in time $\tilde{O}(n)$. It can be formulated abstractly as an application of the so-called Legendre transformation, which can even be implemented in linear instead of quasilinear time [27].

Tropical convolutions on general inputs are conjectured to need a least a quadratic running time, whereas there is a linear-time algorithm if both sequences are convex. So far, no results are known which interpolate between fully convex and non-convex sequences in a way that reproduces the quadratic running time in the hardest cases, and the linear running time in the easiest cases. This shows that the role convexity plays is not yet well understood and raises the following question: Can this difference in complexity be quantified? In particular, can the degree of convexity of a sequence be measured in a way that explains both the linear and the quadratic running times in the "most convex" and "most non-convex" cases? We answer these questions in the affirmative. We give two sensible measures to quantify the convexity of an instance. Further, we use them to obtain algorithms which interpolate between convex and non-convex instances.

This line of research can be understood as part of a recent branch of research in parameterized algorithms referred to as FPT-in-P algorithms [2, 22, 23, 24, 28].[1] These are algorithms that admit FPT-style running times of the form $f(k) \cdot n^{O(1)}$, but for problems that are polynomial-time solvable, with the goal of reducing the polynomial dependency on n for small values of k.

Related Problems

Given that $(\min,+)$-Convolution is a well-studied problem within fine-grained complexity theory, there are many other, closely related problems that can either be reduced to $(\min,+)$-Convolution or that $(\min,+)$-Convolution reduces to. The independent articles by Cygan et al. [19] and Künnemann, Paturi and Schneider [25] give an excellent overview over these problems and their relations to each other, and we make no attempt of recalling all of them here.

Yet, there are some specific problems that are relevant in the context of this article. Let us first address some obvious variations of $(\min,+)$-Convolution. Consider the problem obtained from $(\min,+)$-Convolution after changing the min in the definition of $a * b$ for a max, thereby obtaining the following problem.

$(\max,+)$-Convolution	
Input:	Two sequences $a, b \in \bar{\mathbb{Z}}^{n+1}$
Output:	All entries of the sequence $c \in \bar{\mathbb{Z}}^{2n+1}$, where $c_k = \max_{i+j=k} a_i + b_j$

For general inputs, these two cases are interreducible via a pointwise negation of the inputs. Since we are concerned with convexity of inputs, we note that whatever applies to convex sequences for $(\min,+)$-Convolution applies equally to *concave* sequences for $(\max,+)$-Convolution, which are precisely the pointwise negations of convex sequences. In particular,

[1] Of course, as with parameterized algorithms in general, there is a large number of works that fall into the regime of FPT-in-P algorithms before the term had been coined.

our algorithmic results and even our notion of measuring convexity translate in a direct manner to algorithmic results about $(\max, +)$-Convolution. For ease of exposition, we thus only speak about $(\min, +)$-Convolution in the remainder of this article.

Another very closely related problem studied in its maximization-guise in [19] is:

Super Additivity Testing	
Input:	A sequence $a \in \bar{\mathbb{Z}}^{n+1}$
Output:	Whether $a_k \leq \min_{i+j=k} a_i + a_j$ for all k holds

This can be immediately reduced to the special case of $(\min, +)$-Convolution where $a = b$ holds, and $O(n)$ additional comparisons – in particular, all our algorithms for $(\min, +)$-Convolution yield also algorithms for Super Additivity Testing.

One more problem relevant for this article that is close in spirit to $(\min, +)$-Convolution is the following:

Convolution 3-Sum	
Input:	A sequence $t \in \mathbb{Z}^{n+1}$
Output:	Whether there are $0 \leq i \leq j \leq n$ such that $t_i + t_j = t_{i+j}$

This problem is a variant of the well-known 3-Sum problem. Most relevant to our work is the reduction from 3-Sum to Convolution 3-Sum, first appearing as a randomized reduction [29]. Later, this reduction was made deterministic [13].

Finally, we deal with the following problem.

Minimum Consecutive Subsums	
Input:	A sequence $t \in \mathbb{Z}^n$
Output:	For each length $k = 1, \ldots, n$, $\min_{i=1,\ldots,n-k+1} \sum_{j=i}^{i+k-1} t_i$

In prose, we are looking for the minimum consecutive sum of every length in the sequence t. The fine-grained equivalence between $(\min, +)$-Convolution and Minimum Consecutive Subsums was first observed in [26]. Like $(\min, +)$-Convolution, the problem Minimum Consecutive Subsums has an obvious maximization variant Maximum Consecutive Subsums, which we also discuss.

Our Contribution

We present new measures to capture the degree of convexity of an instance of $(\min, +)$-Convolution, namely, the *convex sequence number* and the *convex rank*. We present a linear-time algorithm to compute the convex sequence number of a sequence (and its decomposition into convex sub-sequences). This measure can be large for a particular class of inputs whose convexity can nevertheless be exploited efficiently. To handle such cases, we introduce a generalization to the convex sequence number, which we call the convex rank of a sequence. We prove that we can compute a sufficiently accurate approximation to this parameter and output the corresponding convex sequences in polynomial time.

Secondly, we present efficient algorithms for $(\min, +)$-Convolution, Super Additivity Testing, Convolution 3-Sum, and Minimum Consecutive Subsums under both parameters. This yields $\tilde{O}(sn)$ algorithms for each problem if the input sequence(s) have length n and a convex sequence number of s. As for the convex rank, $(\min, +)$-Convolution, Super Additivity Testing, and Minimum Consecutive Subsums can be solved in time $\tilde{O}(r^2 n)$ for r being the convex rank of the input sequence(s). These algorithms contribute to the understanding of the usefulness of convexity and the structural properties

of instances with bounded convexity. Further, our observations also lead to the result for $(\min,+)$-Convolution on fully convex sequences but avoids the heavy machinery of Legendre transformations.

Related Work

We have already pointed out related algorithmic results on $(\min,+)$-Convolution. More closely related to our bounded-convexity regime, there are other works that consider sequences that are "near-convex" in other ways. For instance, Axiotis-Tzamos [4] consider k-step concave sequences (and the extension to convexity is trivial); Bateni et al. [6] as well as a recent preprint of Bringmann-Cassis [11] use so-called Δ-convexity; furthermore, Arkin et al. define a *convex partition number* [3] in a different context. While all of these measures are well-motivated in the respective papers, they do not meet the criterion that motivates the research reported on in the present paper: they do not allow in a natural manner to interpolate between maximally non-convex sequences (corresponding to parameter value roughly n) and maximally convex (that is, plainly, convex) sequences (corresponding to parameter value 1). It is this gap that this article addresses.

2 Preliminaries

A sequence $a \in \bar{\mathbb{R}}^{n+1}$ is called *convex* if, for all $i = 1, \ldots, n-1$, it holds that

$$2a_i \leq a_{i-1} + a_{i+1}.$$

This can be seen as a discretized version of the characterization that a differentiable function is convex if and only if its derivative is monotonically increasing; in particular, the preceding condition is equivalent to the *slopes* of the sequence satisfying

$$a_i - a_{i-1} \leq a_{i+1} - a_i.$$

By convention, we regard every sequence of length less than three as convex. This is in line with the characterization of convex sequences as those sequences whose piecewise-linear interpolation epigraph is convex.

Let us formally define the matrix consisting of all possible index combinations a and b.

▶ **Definition 2.** *Let* $[a,b] \in \bar{\mathbb{R}}^{(n+1)\times(n+1)}$ *denote the* tropical rank-1 matrix *associated with* $a \in \bar{\mathbb{R}}^{(n+1)}$ *and* $b \in \bar{\mathbb{R}}^{(n+1)}$*, defined through:*

$$[a,b]_{i,j} = a_i + b_j \text{ for } i,j = 0,\ldots,n.$$

▶ Remark 3. There are several inequivalent variants of the tropical rank of a matrix, each stemming from a different characterization of matrix rank in the usual sense. The notion of rank-1 matrices employed here corresponds to what is sometimes called the *Barvinok rank* of a matrix [21].

Define the entries in the k-th antidiagonal for $k = 0,\ldots,2n$ in $[a,b]$ as

$$[a,b]^{(k)} = \begin{cases} ([a,b]_{0,k}, [a,b]_{1,k-1}, \ldots, [a,b]_{k,0}) & \text{if } 0 \leq k \leq n, \\ ([a,b]_{n,k-n}, [a,b]_{n-1,k-n+1}, \ldots, [a,b]_{n-(k-n),2k-2n}) & \text{if } n < k \leq 2n. \end{cases}$$

Computing $a * b$ can be equivalently viewed as finding the minimum in each antidiagonal in $[a,b]_{i,j}$. In particular,

$$c_k = \min [a,b]^{(k)}$$

and thus,

$$a * b = c = (\min\ [a,b]^{(0)}, \ldots, \min\ [a,b]^{(2n)}).$$

The two following lemmas will come in handy for our algorithms.

▶ **Lemma 4.** *Let $a, b \in \bar{\mathbb{R}}^{n+1}$ be convex sequences. Then, for each k, the k-th antidiagonal $[a,b]^{(k)}$ is also a convex sequence.*

Proof. Let $\tilde{a}_k = (a_0, \ldots, a_k)$ for $0 \leq k \leq n$ and $\tilde{a}_k = (a_{k-n}, \ldots, a_n)$ for $n < k \leq 2n$, i.e., the subsequence of a used to form the antidiagonal $[a,b]^{(k)}$. We define analogously $\tilde{b}_k = (b_k, \ldots, b_0)$ for $0 \leq k \leq n$ and $\tilde{b}_k = (b_n, \ldots, b_{k-n})$ for $n < k \leq 2n$. It is clear that $\tilde{a}_k$ is convex for all k. The convexity of $\tilde{b}_k$ for all k follows geometrically from the fact that convexity of the epigraph is retained by reflection along a line parallel to the ordinate. The sum of $\tilde{a}_k$ and $\tilde{b}_k$ is convex since $\tilde{a}_k$ and $\tilde{b}_k$ are, and this sum is precisely $[a,b]^{(k)}$. ◀

▶ **Lemma 5.** *The minimum of a convex sequence $t \in \bar{\mathbb{R}}^{n+1}$ can be computed in time $O(\log n)$.*

Proof. Convex sequences are characterized by the differences between the values at adjacent positions increasing, and their minimum is attained where these differences pass from negative to positive. Since there are n differences and they form a non-decreasing sequence, this can be done in time $O(\log n)$ using binary search. ◀

▶ Remark 6. Note now that Lemmas 5 and 4 imply directly a $\tilde{O}(n)$ algorithm for $(\min, +)$-Convolution on convex sequences, by computing the minima of the $2n+1$ anti-diagonals in time $O(\log n)$. We extend this algorithm based on binary search on the anti-diagonals to more general, non-convex sequences. The same principle will be useful for algorithms for other problems related to $(\min, +)$-convolutions.

In addition to this method, we use a second approach, recently described in [12, 4], that can be employed to design a linear-time algorithm for convolving a convex sequence and an *arbitrary* sequence, convex or not. This is based on a variant of the matrix $[a,b]$, defined as follows:

▶ **Definition 7.** *Pad b with $n+1$ ∞-entries to the left, by setting $b_{-i} = \infty$ for $1 \leq i \leq n+1$, and let $\overline{[a,b]} \in \bar{\mathbb{R}}^{(n+1)\times(n+1)}$ $\overline{[a,b]}_{k,i} = a_i + b_{k-i}$. We call $\overline{[a,b]}$ the* shifted rank-1 matrix *associated to a and b.*

We then make the following observation, as done in [12]:

▶ **Lemma 8.** *Let b be convex. Then $\overline{[a,b]}$ is* Monge*, that is*

$$\overline{[a,b]}_{i,j} + \overline{[a,b]}_{i+1,j+1} \leq \overline{[a,b]}_{i+1,j} + \overline{[a,b]}_{i,j+1}.$$

Proof. First, note that padding b with ∞-entries does not impact convexity. Then, we have that

$$\begin{aligned}
&\overline{[a,b]}_{i,j} + \overline{[a,b]}_{i+1,j+1} - \overline{[a,b]}_{i+1,j} - \overline{[a,b]}_{i,j+1} = \\
&a_i + b_{j-i} + a_{i+1} + b_{j-i} - a_{i+1} - b_{j-i-1} - a_i - b_{j-i+1} = \\
&2b_{j-i} - b_{j-i-1} - b_{j-i+1},
\end{aligned}$$

and $2b_k \leq b_{k-1} + b_{k+1}$ is precisely the definition of b being convex. ◀

▶ Remark 9. Note that instead of searching for minima of anti-diagonals in $[a,b]$, the equivalent task for $\overline{[a,b]}$ is finding the minimum of each row, which, for Monge matrices, can be accomplished in linear time via the SMAWK algorithm [1].

3 Convexity Measures

In the following, we introduce two related, natural definitions for quantifying convexity. Both of these measures come with different advantages and can be used in different settings, witnessing their independent relevance.

The first convexity measure called *convex sequence number* captures the smallest number of cuts to divide a sequence $t \in \bar{\mathbb{R}}^{n+1}$ into convex sub-sequences in a straightforward manner. In turn, the *convex rank* of t is the number of convex sequences of length $n+1$ such that their index-wise minimum equals t. While the convex sequence number is easy to compute, it may be large compared to the convex rank whenever the convex length-$n+1$ sub-sequences are "entangled." We elaborate on an example below. However, we do not know how to compute the convex rank exactly. Instead, we show how to efficiently compute an $O(\log(n))$-approximation algorithm for the convex rank, which is indeed sufficient to (nearly) maintain the running time guarantees of our algorithms.

Convex Sequence Number

Let us define the convex sequence number s formally.

▶ **Definition 10.** *Let $t = (t_0, \ldots, t_n) \in \bar{\mathbb{R}}^{n+1}$ be a sequence. If there exist indices $I = (i_0, i_1 \ldots, i_{s-2})$ with $i_0 < i_1 < \cdots < i_{s-2}$ such that the sequences $(t_0, \ldots, t_{i_0})$, $(t_{i_0+1}, \ldots t_{i_1})$, $\ldots$, $(t_{i_{s-2}+1}, \ldots, t_n)$ are convex and $|I|$ is of minimum cardinality, then we call $s = |I| + 1$ the convex sequence number of t.*

That is, the convex sequence number measures the smallest amount of cuts we have to make in t such that all s sub-sequences are convex, which we believe to be a natural way of measuring how convex a sequence is that is accessible on an intuitive level. We refer to the indices in I as *cuts*. If it is not obvious from the context, we use the convention to denote the convex sequence number of a sequence t by s_t.

Indeed, we can compute the convex sequence number efficiently.

▶ **Theorem 11.** *There is a linear-time algorithm that outputs the minimum number of cuts for a decomposition of a sequence $t \in \bar{\mathbb{R}}^{n+1}$ into convex sub-sequences.*

Proof. We proceed in a greedy fashion. That is, we pass through the sequence and check if each new point added to the sequence satisfies the convexity property.

In detail, we start with the first sub-sequence. Every two next points can always be added (or less, if we already arrived at the end of the instance). For every next new point with index $i+1$, we check whether the second-order difference $t_{i-1} - 2t_i + t_{i+1}$ is non-negative. If so, we add the point and continue. Otherwise, we define $i+1$ as the cut and repeat the procedure.

Regarding minimality, this is indeed optimal as we cannot add any further point to the current sub-sequence and each new point is independent of the points selected before the direct two predecessors. As for correctness, note that the resulting sub-sequences are convex by construction.

It is clear that this procedure takes a linear amount of steps in the input length. ◀

The following property is crucial for our algorithms.

▶ **Lemma 12.** *Let $a, b \in \bar{\mathbb{Z}}^{n+1}$. The parameter convex sequence number is subadditive on each antidiagonal, that is,*

$$s_{[a,b]^{(k)}} \leq s_a + s_b$$

holds for all k.

Proof. Let $I = i_0 < \ldots < i_{s_a-2}$ be the cuts of a decomposition into convex sub-sequences of a, and similarly, $J = j_0 < \cdots < j_{s_b-2}$ for b.

Let $\ell_0 < \ell_1 < \ldots < \ell_{s-2}$ be the union of I and J in increasing order. Note that we can still assume this arrangement to be strictly increasing, since taking the union of two sets removes any potential duplicates. Then, clearly, it holds that $s \leq s_a + s_b$.

It remains to show that $\ell_0, \ldots, \ell_{s-2}$ are indeed the cuts of $[a, b]^{(k)}$ into convex sub-sequences, and thus $s \geq s_{[a,b]^{(k)}}$. Since a and b are per definition convex on each of their sub-sequences, they are also convex on every contiguous subset of them. That is, on each sub-sequence induced by the breakpoints ℓ, a and b are both convex. Hence, $[a, b]^{(k)}$ is convex on these sub-sequences as well, which was to show. ◀

▶ Remark 13. We may at this point already observe the following generalization of the convolution algorithm for purely convex sequences described in Remark 6: Namely, there is an algorithm for $(\min, +)$-Convolution on two sequences $a, b \in \bar{\mathbb{Z}}^{n+1}$ running in time $\tilde{O}((s_a + s_b) \cdot n)$, by employing Theorem 11 and Lemma 12, and observing that the merged index sets in the proof of the latter can be computed in linear time in n.

Convex Rank

For convex sequences, also the procedure from the previous remark can be adapted to use the SMAWK algorithm. To this end, let $b^{(1)}, \ldots, b^{(s_b)} \in \bar{\mathbb{R}}^{n+1}$ be the pieces of the convex sequence decomposition of b, with each $b^{(i)}$ padded with ∞-entries outside the indices corresponding to the i-th index interval in the decomposition of b. Let $c = a * b$ as before. Then, observe that

$$c_k = \min_{s=1,\ldots,s_b} (a * b^{(s)})_k$$

holds for all k, that is, c is given as the point-wise minimum of $a * b^{(1)}, \ldots, a * b^{(s_b)}$. Using the fact that $b^{(s)}$ is convex by definition for each s, we can then apply the SMAWK-based algorithm from Remark 9 to compute c in time $O(\min\{s_a, s_b\} \cdot n)$ This observation can be generalized as follows.

▶ **Lemma 14.** *Let $a, b^{(1)}, \ldots, b^{(r)} \in \bar{\mathbb{R}}^{n+1}$ be any convex sequences, and let $b \in \bar{\mathbb{R}}^{n+1}$ be defined by setting $b_i = \min_\rho b_i^{(\rho)}$, that is, b is the point-wise minimum of the $b^{(\rho)}$. Let $c = a * b$ and $c^{(\rho)} = a * b^{(\rho)}$. Then,*

$$c_k = \min_\rho c_k^{(\rho)}$$

for all k, that is, c is the point-wise minimum of the $c^{(\rho)}$.

Proof. Follows directly from the fact that $x + \min\{y, z\} = \min\{x + y, x + z\}$ for all $x, y, z \in \bar{\mathbb{R}}$. ◀

This observation motivates directly and naturally the following definition:

▶ **Definition 15.** *Let $t = (t_0, \ldots, t_n) \in \bar{\mathbb{R}}^{n+1}$ be a sequence. If there exists a set $T = \{t^{(0)}, t^{(1)} \ldots, t^{(r-1)}\}$ of convex sequences with $t_i \in \bar{\mathbb{R}}^{n+1}$ such that the index-wise minimum satisfies $t_k = \min\{t_k^{(0)}, \ldots, t_k^{(r-1)}\}$, then we call T a* convex r-decomposition *of t. The minimum r such that there exists a convex r-decomposition for t we call the* convex rank *of t.*

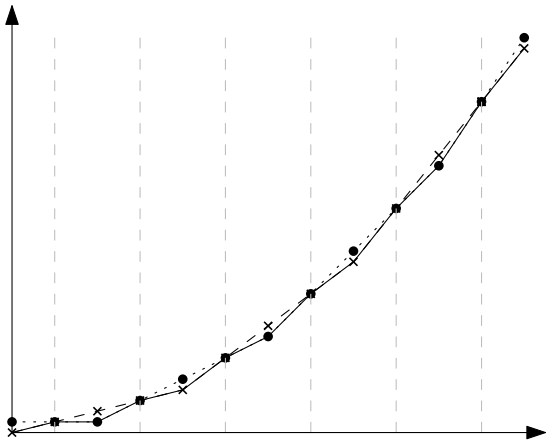

Figure 1 The solid line is the piece-wise linear extension of a non-convex sequence c. The dashed and dotted lines define the piece-wise linear extension of two convex sequences a and b whose point-wise minimum is c (the values of a and b are indicated by disks and crosses, respectively). Dashed, gray vertical lines signify the cut points of the convex sequence number of the sequence c.

Beyond the motivation through the structural observation in Lemma 14, it may require further elaboration to make plausible the supposition that the convex rank is indeed a natural measure of convexity. In essence, it can be viewed as one answer to the question: How many convex sequences are needed to build t from them? Any answer to this question depends on what operation constitutes the formal meaning of "building" a sequence from others.

In the general context of mathematical structures, this is formalized as having a set S with a binary operation $\oplus$ (the "building" operation), such that S contains some distinguished subset $X \subset S$ of particular interest. Then, it is natural to ask for the *X-rank* of an arbitrary element $s \in S$: If $s \in X$, its X-rank is one, and in general it is the minimum number r needed to write $s = x_1 \oplus \cdots \oplus x_r$ with $x_i \in X$ for all $1 \leq i \leq r$. Examples of this abound in the familiar setting of $\oplus$ being ordinary addition over a linear space: If S is a linear space of matrices, X can be taken to be the set of matrices of the form $A = uv^T$ for vectors u and v of the appropriate dimension; in this case, X-rank recovers the ordinary matrix rank. A more involved example from algebraic geometry is furnished by *secant varieties* of a variety X, arising as (the closure of projectivized) ordinary sums of points on the variety X, recovering symmetric, anti-symmetric and tensor (border) rank (when choosing X to be the Veronese, Grassmannian and Segre, respectively), see [8] for ample background on this particular class of examples.

In the more closely related context of tropical convolutions, the operation $\oplus$ being the point-wise minimum of its operands is the natural "additive" operation, forming a semiring together with the operation of tropical convolution. For example, the *Barvinok rank* of a tropical matrix (see [5] and the treatment in [21]), is the smallest number of tropical rank-one matrices needed to express a given matrix as their pointwise minimum. Work by Develin [20] deepens the analogy to ordinary ranks through a tropical analogue of secant varieties. In all of the cases outlined, X-rank corresponds to the complexity of some object with respect to a representation (as a sum) by the set X. Given this mathematical context, it is a very natural thing to measure the degree of convexity through the concept of X-rank of a sequence, where X is the set of convex sequences.

Relation to Convex Sequence Numbers

From its definition, it is clear that the convex rank of a sequence is at most its convex sequence number. However, the two measures can be arbitrarily far apart: Roughly speaking, the critical instances are those where two non-consecutive sub-sequences of t together form a convex sub-sequence (padding the missing indices accordingly). In particular, two convex sequences can be intertwined arbitrarily often. Choosing points which are always at the point-wise minimum of the two sequences, we can produce an arbitrarily large convex sequences number as every time the sequences are intertwined further, their intersection point defines a new cut in the sense of convex sequence numbers. Indeed, consider any prefix of the infinite sequences which are defined for $i \geq 0$ as

$$\begin{aligned} a_{2i} &= i^2 + (i \bmod 2), \\ b_{2i} &= i^2 + 1 - (i \bmod 2), \\ b_{2i+1} = a_{2i+1} &= i^2 + i + 1. \end{aligned}$$

An easy calculation shows that both a and b define convex sequences, but their point-wise minimum has unbounded convex sequence number. See Figure 1 for an illustration.

Algorithms and Properties

An obvious question is how to compute the convex rank of a sequence. First, note the following.

▶ **Proposition 16.** *There is an algorithm running in time $O(r^n)$ to decide whether or not a given sequence $a \in \bar{\mathbb{Z}}^{n+1}$ has convex rank at most r.*

Proof. Let $a \in \bar{\mathbb{Z}}^{n+1}$ and r be given. By definition, for every $i = 0, \ldots, n$ in any convex r-decomposition $a^{(1)}, \ldots, a^{(r)}$ of a there must be some ρ such that $a_i = a_i^{(\rho)}$. Now, we may guess this ρ for every i and obtain r sequences $\hat{a}^{(1)}, \ldots, \hat{a}^{(r)}$ defined partially only at those indices i for which the current guess assumed a given $\hat{a}^{(\rho)}$ to satisfy $a_i = \hat{a}_i^{(\rho)}$. However, it is easy to extend these partial definitions in a piece-wise linear manner (and with ∞ from the left and the right) and check if the resulting sequences are convex and define a as their point-wise minimum. Clearly, if all $\hat{a}^{(\rho)}$ are convex and yield a point-wise, a is of convex rank at most r. If this is not the case for any guess of assignments, then a is of convex rank at least $r + 1$. ◀

In addition to this brute-force approach, we now show how we can approximate the convex rank of a sequence t with a greedy method efficiently, yielding the following theorem. For the remainder of the paper, we write r_t to denote the convex rank of a sequence t.

▶ **Theorem 17.** *There is an $O(\log(n))$-approximation algorithm running in time $O(r_t \cdot n^4 \log n)$ that outputs for a sequence $t \in \bar{\mathbb{R}}^{n+1}$ its decomposition into convex sequences of length $n + 1$ such that their index-wise minimum equals t.*

The algorithm is a greedy algorithm enriched with a dynamic program. The intuition is as follows: We consider the problem as a sort of arithmetic variant of Set Cover. In particular, we are given a sequence of numbers t, and the goal is to "cover" this sequence of numbers with convex sequences $t^{(0)}, \ldots, t^{(r-1)}$, such that for each index i, none of the elements $t_i^{(j)}$ in any sequences j in the covering can be strictly less than t_i. The approach is similar to the greedy approximation algorithm for Set Cover. However, making locally optimal decisions in each step requires a separate dynamic program.

Proof of Theorem 17. We make the assumption that t contains no ∞-entries: trailing and leading ∞-entries can simply be removed from t without changing the optimum. Furthermore, if there is an ∞ in the interior of t, this splits t into two disjoint parts, $t(1)$ and $t(2)$, such that $t = (t(1), \infty, t(2))$. Any convex sequence $t^{(i)}$ that can contribute to a solution (that is, has $t^{(i)}_j = t_j$ for at least one j) can be less than ∞ on at most one of $t(1)$ or $t(2)$, so we may treat $t(1)$ and $t(2)$ as separate instances, and their optimal solutions are the unions of any two optimal solutions of $t(1)$ and $t(2)$, respectively. This procedure can be repeated for any remaining ∞-entries of t.

Now, at each step of our greedy algorithm, we aim to solve the following problem. Given a set of indices $I \subseteq \{0, \dots, n\}$, we wish to compute a convex sequence $t^{(j)} \in \bar{\mathbb{R}}^{n+1}$ that satisfies $t^{(j)}_i \geq t_i$ for all $i \in \{0, \dots, n\}$, and furthermore, meets the following criterion: Let $J_j \subseteq \{0, \dots, n\}$ be the set of indices where equality holds, i.e., $t^{(j)}_i = t_i$ if and only if $i \in J_j$. Then, our goal is to find a sequence $t^{(j)}$ maximizing $|J_j \setminus I|$. The set I will take on the following role in the greedy procedure: At every step, we keep track of the indices in the original sequence t that already have been covered by the selection of sequences up until this point. The set $J_j \setminus I$ is then the set of indices that are covered by the newly constructed sequence $t^{(j)}$.

For this intermediate problem, we now construct a dynamic program that solves it optimally. First, we argue that we can make the following assumption on any optimal solution $t^{(j)}$ without loss of generality: Firstly, the piece-wise linear extension of $t^{(j)}$ is not linear precisely at the indices J_j. Secondly, $t^{(j)}_i = \infty$ for all values of i outside of $[\min J_j, \max J_j]$. Let us refer to such sequences $t^{(j)}$ as *t-compatible.* This can be seen as follows. Consider some convex sequence $t^{(j)}$ that satisfies $t^{(j)}_i \geq t_i$ at all i, and has $t^{(j)}_i = t_i$ at indices $i \in J_j = (j_1, \dots, j_p)$, where $j_1 < \ldots < j_p$. Observe that replacing $t^{(j)}_i$ with ∞ both strictly before and after j_1 and j_p retains convexity of $t^{(j)}$. We further replace the segment of the piece-wise linear extension of $t^{(j)}$ between j_ℓ and $j_{\ell+1}$ with the straight line segment connecting the points $(j_\ell, t^{(j)}_{j_\ell})$ and $(j_{\ell+1}, t^{(j)}_{j_{\ell+1}})$ contained in this epigraph. By the geometric characterization of convexity of a sequence as convexity of the epigraph of its piece-wise linear extension, and in turn by the definition of convexity as containing all line segments between any pair of contained points, the sequence $t^{(j')}$ obtained by this operation has $t^{(j')} \geq t^{(j)} \geq t$ at every point. Moreover, replacing $t^{(j)}$ by $t^{(j')}$ corresponds to intersecting the epigraph of the piece-wise linear extension of $t^{(j)}$ with a collection of half-planes. Therefore, $t^{(j')}$ is a convex sequence satisfying $t^{(j')} \geq t^{(j)} \geq t$ (point-wise), and the restrictions of $t^{(j')}, t^{(j)}$ and t to J_j are all equal. By construction, $t^{(j')}$ is t-compatible.

Consider now Δ, the set of normalized differences between any two (possibly non-consecutive) values of t, that is, $\Delta = \left\{ \frac{t_i - t_j}{|i-j|} \mid i \neq j \right\} \cup \{\pm\infty\}$. In our dynamic program, we keep track of the following data: For each index i and every $\delta \in \Delta$, we are interested in t-compatible sequences $t^{(j)}$ maximizing $|J_j \setminus I|$ among all choices of $t^{(j)}$ such that the following holds: (1) $t^{(j)}_i = t_i$, $t^{(j)}_\ell = \infty$ for all $\ell > i$, and (2) $t^{(j)}_i - t^{(j)}_{i-1} \leq \delta$ if $i > 0$. For each $i > 0$ and δ, let $T[i, \delta]$ contain such a sequence. If indeed these conditions are satisfied, then $\bigcup_i T[i, \infty]$ contains an optimal solution. Now, towards constructing T that contains such an optimal sequence at every index, let first $T[0, \delta] = (t_0, \infty, \dots, \infty)$ for all $\delta > -\infty$, and $T[i, -\infty] = (\infty, \dots, \infty, t_i, \infty, \dots, \infty)$ for all i. Note that the condition $t^{(j)}_i - t^{(j)}_{i-1} \leq -\infty$ forces $t^{(j)}_i < \infty$ and $t^{(j)}_{i-1} = \infty$. Compute then, for every i, the set $V(i)$ of all indices *visible from* i, that is, all indices $j < i$ such that the straight line segment between t_i and t_j is contained in the epigraph of the piece-wise linear extension of t. Then, for $\delta > -\infty$, $T[i+1, \delta]$ is given by the optimum sequence $t^{(j)}$ contained in $\bigcup_{j \in V(i+1)} T[j, \delta]$, extended between j and $i+1$ with the straight-line segment connecting (j, t_j) and (i, t_{i+1}). In particular, $t^{(j)}_{i+1} - t^{(j)}_i = \delta$ in the sequence corresponding to this straight-line extension.

The sequences are t-compatible by construction: The sequence has a possible leading and trailing stretch of ∞, and at all points in between, it is constructed by forming straight-line connections between points of the form (i, t_i) and (j, t_j). Furthermore, the sequences are convex, because the slopes between consecutive line segments are chosen to be non-decreasing. Optimality follows inductively. Now, applying this algorithm greedily yields the desired approximation bound, with an identical analysis as for Set Cover [17].

As for the running time, the table T has $O(n^3)$ entries; computing the sets Δ, as well as $V(i)$ over all i, takes $O(n^2)$ time. Computing a single table entry therefore requires $O(n)$ time because each $V(i)$ is of size $O(n)$, and over all $O(n^3)$ table entries yields an algorithm running time in $O(n^4)$. Performing this greedily at most $O(r_t \log n)$ times gives the claimed running time bound. ◀

▶ Remark 18. The greedy set cover approach has proved useful in a vast number of combinatorial and geometric problems. Notably, e.g. the so-called *convex partition number* [3] admits a similar, if much more direct approximation via the greedy set cover heuristic, which is however known to NP-hard to compute. Another concept related at least in spirit to convex rank is the notion of Barvinok rank, where one asks for the minimum number of tropical rank-one matrices needed to express a given matrix as their entry-wise minimum. This quantity, in turn, is hard to compute (and even approximate) [30]. One pressing question raised by these results is the complexity of exactly computing the convex rank of a sequence.

As it will turn out, the running time for computing the decomposition is the bottleneck in the algorithms. The blow-up for the algorithms itself is negligible though, as it only adds a factor of $O(\log n)$. So, to distinguish between computing the decomposition and the running times of the algorithms, we suppose in the following that the decomposition is given. Such assumptions are commonly used since at least 1993 [18] in the regime of fine-grained complexity and fixed-parameter tractability to distinguish the running times with respect to some parameter and the corresponding computation of the decomposition, see, e.g., the parameters and algorithms for treewidth [9], cliquewidth [18], and twinwidth [7] (which is even NP-hard to compute optimally) among others.

▶ **Lemma 19.** *Let $a, b \in \bar{\mathbb{Z}}^{n+1}$. It holds that*

$$r_{[a,b]^{(k)}} \leq r_a \cdot r_b$$

for all k.

Proof. Denote by $A = \{a^{(0)}, a^{(1)} \ldots, a^{(r_a-1)}\}$ the convex sequences of the decomposition of a, and similarly, $B = \{b^{(0)}, b^{(1)} \ldots, b^{(r_b-1)}\}$ for b.

Set $C = \{a^{(\ell)} * b^{(m)} \mid a^{(\ell)} \in A, b^{(m)} \in B)\}$. We claim that C corresponds to a decomposition of $[a, b]^{(k)}$ into convex sequences.

Let $[a, b]^{(k)} = c = (c_0, \ldots, c_{2n})$ be the entries of the antidiagonal. Each entry c_k is a sum of two sequences $a_i^{(\ell)} \in A$ and $b_{k-i}^{(m)} \in B$ as all entries of a and, respectively, b are preserved in (at least) one of the sequences of the decompositions.

All entries corresponding to one of such combinations $a^{(\ell)}, b^{(m)}$ form a convex sequence as adding two convex sequences remains convex.

It holds that $|C| = r_a \cdot r_b > r_{[a,b]^{(k)}}$. This concludes the proof. ◀

4 Convolution Problems

In this section, we present efficient algorithms for the convolution problems under the paradigm of both convexity measures. Note that $r_a \leq s_a$ implies that, whenever the dependence on r_a in an algorithm is linear, this also implies an algorithm with linear dependence on s_a.

However, there are cases where the algorithms we obtain have running times depending on r_a e.g. quadratically, in which case a separate treatment of parameterizations by s_a still makes sense.

$(\min, +)$-Convolution

We start with the $(\min, +)$-Convolution problem with respect to the convex rank, expediting Lemma 14.

▶ **Theorem 20.** *There is an algorithm for* $(\min, +)$-Convolution *on two sequences* $a, b \in \bar{\mathbb{Z}}^{n+1}$ *running in time* $O(r_a \cdot n)$*, provided a convex rank decomposition* $a^{(1)}, \ldots, a^{(r_a)}$ *of* a *is given.*

Proof. From Lemma 14, it suffices to compute $a^{(\rho)} * b$ for all $\rho = 1, \ldots, r_a$, which can be accomplished in time $O(n)$ using the SMAWK algorithm as pointed out in Remark 9. Taking point-wise minima can be done in time $O(r_a \cdot n)$, and this shows the claimed running time. ◀

▶ Remark 21. This algorithm applies to the case of max-convolution of concave rank decomposition, yielding the same running time bounds. In case the decomposition is not given, we note that the overhead of applying the decomposition algorithm of Theorem 17 to the entire input naively, adding a $O(n^4 r_t)$ preprocessing step, can be ameliorated to some degree: Split a and b into n^c consecutive parts of length n^{1-c} each. Then, preprocess each part separately in time $O(n^{4(1-c)} r_t)$, and use the $O(n^{1-c} r_t)$-time algorithm for each of the n^{2c} pairs of such parts to find the solution of the original instance in time $O(n^{4-3c} + n^{c+1})$, which is minimized for $c = 3/4$, giving an algorithm running in time $O(n^{7/4} r_t)$.

Observe that there is a direct lower bound under the $(\min, +)$-convolution conjecture: There is no algorithm solving $(\min, +)$-Convolution in time less than $((s_a + s_b)n)^{1-\epsilon} \cdot \text{polylog}(d)$ on integral inputs a, b of maximal absolute value d, for any constant $\epsilon > 0$: Suppose that there exists an algorithm solving $(\min, +)$-Convolution in time $((s_a + s_b)n)^{2-\epsilon} \cdot \text{polylog}(d)$ on integral inputs a, b of maximal absolute value d, for any constant $\epsilon > 0$. As each sequence a (and b) can have a convex sequence number of at most $n/2$ (every two consecutive points form one convex sub-sequence), this would mean that we can solve $(\min, +)$-Convolution in time

$$((s_a + s_b)n)^{1-\epsilon} \cdot \text{polylog}(d) \leq ((n/2 + n/2)n)^{1-\epsilon} \leq n^{2-\epsilon'} \cdot \text{polylog}(d)$$

on integral inputs a, b of maximal absolute value d, for some constant $\epsilon' > 0$, contradicting the $(\min, +)$-convolution conjecture.

Minimum Consecutive Subsums

We consider the Minimum Consecutive Subsums problem on convex inputs. Even though there exists a reduction to $(\min, +)$-Convolution, see [26], it does not preserve the convexity of the input. In particular, the reduction computes a such that the i-th entry is the sum of the first i elements in the (in our case convex) input sequence t. For b, the reduction computes the sum of the last i elements as its i-th entry. While the first sequence remains convex, the second becomes concave. Convolving a convex with a concave sequence results in an arbitrary output with respect to the convexity and thus, our algorithm for $(\min, +)$-Convolution cannot be used to obtain an algorithm with the desired running time bound. However, we observe other structural properties of potential solutions, so that we can indeed solve this problem optimally within the same running time bounds as above yielding the following two theorems.

▶ **Theorem 22.** *There is an algorithm for* Minimum Consecutive Subsums *on a sequence* $t \in \bar{\mathbb{Z}}^{n+1}$ *running in time* $\tilde{O}(s_t n)$.

Proof. We first compute the decompositions of t into at most s_t convex sub-sequences in time $O(n)$ using Theorem 11. Next, we divide each convex sub-sequence into two parts (one of which may be empty), namely a monotonically decreasing sub-sequence (preceding the minimum), and a monotonically increasing part (following the minimum). Call the set of monotone sequences $t^{(1)}, \dots, t^{(k)}$ with $k \leq 2s_t$.

When executing the algorithm, we maintain a table M with n entries where the ith entry corresponds to the best solution found so far for a consecutive subsum of length i. Initialize each value with ∞ (or a sufficiently large value). We then sweep once through the whole sequence t and compute for each point t_j the minimum consecutive subsum starting at t_j, but only for all *reasonable* lengths (defined below) $i \in \{1, \dots, n\}$.

More formally, for each point t_j and each monotone sequence $t^{(k)}$, find its reasonable interval $I^{(k)} = [t^{(k)}_\ell, t^{(k)}_{\ell+1}, \dots, t^{(k)}_{\ell_k}]$ such that $t^{(k)}_\ell - t_{j-1} < 0$ (shifting the whole sum one index to the left is not improving) and $t_j - t^{(k)}_{\ell_k+1} < 0$ (shifting the whole sum one index to the right is not improving). Update all entries in the table M which corresponds to the length of consecutive sums from t_j to $t^{(k)}_p$ for all j, k and $t^{(k)}_p \in I^{(k)}$ if they are smaller than the current entry in M. Finally, output M as the solution. During execution, edge cases are simulated by padding with $\pm\infty$ at the end/beginning of monotonically increasing/decreasing sequences.

Correctness follows easily, as each potential interval holding the minimum is considered. Regarding the running time, it is crucial to see that each monotone piece will be divided into non-overlapping intervals by different t_j. Thus, while considering all start points t_j, we overall go through all points once. Hence, computing the reasonable intervals for each t_j and k, and the corresponding consecutive subsum overall costs time $\tilde{O}(s_t n)$ (using a standard trick, we assume that we computed all initial partial sums of t in a single pass, which makes the interval sums $\sum_{i=i_1}^{i_2} t_i$ accessible in time $\tilde{O}(1)$). ◀

▶ **Theorem 23.** *There is an algorithm for* Minimum Consecutive Subsums *on a sequence* $t \in \bar{\mathbb{Z}}^{n+1}$ *running in time* $\tilde{O}(r_t^2 n)$ *given the convex rank decomposition of* t.

Proof. The algorithm proceeds basically as in Theorem 22. However, we now have to deal with multiple convex sequences. This means that for every point t_j, we consider the reasonable intervals of all convex sequences. Again, there will be exactly one interval for each convex sequence and they are non-overlapping. This leads to a blow-up of a factor of r_t, as we now have $r_t n$ instead of n points overall. Also note that once we determined the start index and end index in the interval, we sum up the actual values in t and not the ones from the decomposition into convex sequences. ◀

▶ Remark 24. As we only care about the monotone sub-sequences of the sequence, the concave case can be solved equivalently. Further, by defining the reasonable intervals such that $t^{(k)}_\ell - t_{j-1} > 0$ and $t_j - t^{(k)}_{\ell+1} > 0$, and initializing the table with $-\infty$ entries, we can also solve the maximising version of the problem called Maximum Consecutive Subsums.

Super Additivity Testing

Turning our attention to Super Additivity Testing, we can immediately deduce the following by applying our previous results.

▶ **Theorem 25.** *There is an algorithm for* Super Additivity Testing *on a sequence* $a \in \bar{\mathbb{Z}}^{n+1}$ *running in time* $O(r_a \cdot n)$ *given the convex rank decomposition of* a*.*

Proof. This follows from choosing $b = a$, applying Theorem 20, and then checking whether the result is at least a_i for each i. ◀

▶ Remark 26. As $(\min, +)$-Convolution is used as a subroutine, we can easily see that these algorithms also work for concave inputs.

Convolution 3-Sum

While the previous problems relied on the SMAWK algorithm, we will now employ the strategy laid out in Remarks 6 and 13.

▶ **Theorem 27.** *There is an algorithm for* Convolution 3-Sum *on a sequence* $a \in \bar{\mathbb{Z}}^{n+1}$ *running in time* $\tilde{O}(s_a \cdot n)$*.*

Proof. We first compute the decompositions of a, b into at most s_a convex sub-sequences in time $O(n)$ using Theorem 11. Then, for every anti-diagonal $[a, a]^{(k)}$ of $[a, a]$, the breakpoints of a decomposition of $[a, a]^{(k)}$ into convex sub-sequences is immediate from the proof of Lemma 12. Instead of asking for the minimum (as for $(\min, +)$-Convolution), the Convolution 3-Sum problem asks whether a specific element a_k is contained in the k-th anti-diagonal $[a, a]^{(k)}$ (although finding the minimum of a convex sub-sequence is still part of our algorithm). We can answer this question in $O(\log n)$ time for each of the $O(s_a)$ convex sub-sequences of $[a, a]^{(k)}$, by performing, first, a binary search to find the minimum of each convex sub-sequences, as in the case of $(\min, +)$-Convolution. Each convex sub-sequence is thereby divided into two parts (one of which may well be empty), namely a monotonically decreasing sub-sequence (preceding the minimum), and a monotonically increasing part (following the minimum). On each of those (at most) two parts, we may then perform an ordinary binary search in order to determine the presence or absence of the element a_k in the current convex sub-sequence of $[a, a]^{(k)}$. Correctness and running time follow immediately. ◀

Interestingly, this algorithm cannot be adapted for the parameter convex rank as before. This is due to the issue that we do preserve the real minimum values, but cannot guarantee that each value in the kth antidiagonal corresponds to some sum $t_i + t_j$ for $i + j = k$. Testing this would give an increase in the running time by a factor of n as each antidiagonal could have up to n positions of the desired value, which would be worse than the trivial known algorithm for this problem. It remains an interesting question whether there exists an $O(r_t n)$ algorithm for Convolution 3-Sum.

5 Open Questions

The results in the present paper suggest further research directions: Some of the algorithms could not be shown to be tight as known reduction do not retain the convexity of the instances, see [19]. Also, excluding a running time of the form $O(n + k^2)$ for both convexity measures k would be of great interest. Further, whether or not the quadratic dependency on the convex rank in the algorithms is necessary is yet to be determined. It remains open if we can compute the convex rank exactly in polynomial time or if this problem is NP-hard.

References

1 Alok Aggarwal, Maria M. Klawe, Shlomo Moran, Peter W. Shor, and Robert E. Wilber. Geometric applications of a matrix-searching algorithm. *Algorithmica*, 2:195–208, 1987. doi:10.1007/BF01840359.

2 Amihood Amir, Moshe Lewenstein, and Ely Porat. Faster algorithms for string matching with k mismatches. *J. Algorithms*, 50(2):257–275, 2004. doi:10.1016/S0196-6774(03)00097-X.

3 Esther M. Arkin, Sándor P. Fekete, Ferran Hurtado, Joseph S. B. Mitchell, Marc Noy, Vera Sacristán, and Saurabh Sethia. On the reflexivity of point sets. In Frank K. H. A. Dehne, Jörg-Rüdiger Sack, and Roberto Tamassia, editors, *Algorithms and Data Structures, 7th International Workshop, WADS 2001, Providence, RI, USA, August 8-10, 2001, Proceedings*, volume 2125 of *Lecture Notes in Computer Science*, pages 192–204. Springer, 2001. doi:10.1007/3-540-44634-6_18.

4 Kyriakos Axiotis and Christos Tzamos. Capacitated dynamic programming: Faster knapsack and graph algorithms. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 19:1–19:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.19.

5 Alexander I. Barvinok, David S. Johnson, Gerhard J. Woeginger, and Russell Woodroofe. The maximum traveling salesman problem under polyhedral norms. In Robert E. Bixby, E. Andrew Boyd, and Roger Z. Ríos-Mercado, editors, *Integer Programming and Combinatorial Optimization, 6th International IPCO Conference, Houston, Texas, USA, June 22-24, 1998, Proceedings*, volume 1412 of *Lecture Notes in Computer Science*, pages 195–201. Springer, 1998. doi:10.1007/3-540-69346-7_15.

6 Mohammad Hossein Bateni, Mohammad Taghi Hajiaghayi, Saeed Seddighin, and Cliff Stein. Fast algorithms for knapsack via convolution and prediction. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 1269–1282. ACM, 2018. doi:10.1145/3188745.3188876.

7 Pierre Bergé, Édouard Bonnet, and Hugues Déprés. Deciding twin-width at most 4 is np-complete. In *ICALP*, volume 229 of *LIPIcs*, pages 18:1–18:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

8 Alessandra Bernardi, Enrico Carlini, Maria Virginia Catalisano, Alessandro Gimigliano, and Alessandro Oneto. The hitchhiker guide to: Secant varieties and tensor decomposition. *Mathematics*, 6(12), 2018. doi:10.3390/math6120314.

9 Hans L. Bodlaender, Pål Grønås Drange, Markus S. Dregi, Fedor V. Fomin, Daniel Lokshtanov, and Michal Pilipczuk. A c^k n 5-approximation algorithm for treewidth. *SIAM J. Comput.*, 45(2):317–378, 2016.

10 David Bremner, Timothy M. Chan, Erik D. Demaine, Jeff Erickson, Ferran Hurtado, John Iacono, Stefan Langerman, Mihai Patrascu, and Perouz Taslakian. Necklaces, convolutions, and X+Y. *Algorithmica*, 69(2):294–314, 2014. doi:10.1007/s00453-012-9734-3.

11 Karl Bringmann and Alejandro Cassis. Faster 0-1-knapsack via near-convex min-plus-convolution. *CoRR*, abs/2305.01593, 2023. doi:10.48550/arXiv.2305.01593.

12 Timothy M. Chan. Approximation schemes for 0-1 knapsack. In Raimund Seidel, editor, *1st Symposium on Simplicity in Algorithms, SOSA 2018, January 7-10, 2018, New Orleans, LA, USA*, volume 61 of *OASIcs*, pages 5:1–5:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/OASIcs.SOSA.2018.5.

13 Timothy M. Chan and Qizheng He. Reducing 3sum to convolution-3sum. In Martin Farach-Colton and Inge Li Gørtz, editors, *SOSA*, pages 1–7. SIAM, 2020. doi:10.1137/1.9781611976014.1.

14 Timothy M. Chan and Moshe Lewenstein. Clustered integer 3sum via additive combinatorics. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *STOC 2015*, pages 31–40. ACM, 2015. doi:10.1145/2746539.2746568.

15 Timothy M. Chan and R. Ryan Williams. Deterministic apsp, orthogonal vectors, and more: Quickly derandomizing razborov-smolensky. *ACM Trans. Algorithms*, 17(1):2:1–2:14, 2021. doi:10.1145/3402926.

16 Shucheng Chi, Ran Duan, Tianle Xie, and Tianyi Zhang. Faster min-plus product for monotone instances. In Stefano Leonardi and Anupam Gupta, editors, *STOC*. ACM, 2022. doi:10.1145/3519935.3520057.

17 V. Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235, 1979.

18 Bruno Courcelle. Monadic second-order logic and hypergraph orientation. In *LICS*, pages 179–190. IEEE Computer Society, 1993.

19 Marek Cygan, Marcin Mucha, Karol Wegrzycki, and Michal Wlodarczyk. On problems equivalent to (min, +)-convolution. *ACM Trans. Algorithms*, 15(1):14:1–14:25, 2019. doi:10.1145/3293465.

20 Mike Develin. Tropical secant varieties of linear spaces. *Discrete & Computational Geometry*, 35:117–129, 2006.

21 Mike Develin, Francisco Santos, and Bernd Sturmfels. On the rank of a tropical matrix. *Combinatorial and computational geometry*, 52:213–242, 2005.

22 Archontia C. Giannopoulou, George B. Mertzios, and Rolf Niedermeier. Polynomial fixed-parameter algorithms: A case study for longest path on interval graphs. *Theor. Comput. Sci.*, 689:67–95, 2017. doi:10.1016/j.tcs.2017.05.017.

23 Jan M. Hochstein and Karsten Weihe. Maximum s-t-flow with k crossings in $O(k^3 n \log n)$ time. In Nikhil Bansal, Kirk Pruhs, and Clifford Stein, editors, *SODA*, pages 843–847. SIAM, 2007. URL: http://dl.acm.org/citation.cfm?id=1283383.1283473.

24 Ioannis Koutis, Gary L. Miller, and Richard Peng. A nearly-m log n time solver for SDD linear systems. In Rafail Ostrovsky, editor, *FOCS*, pages 590–598. IEEE Computer Society, 2011. doi:10.1109/FOCS.2011.85.

25 Marvin Künnemann, Ramamohan Paturi, and Stefan Schneider. On the fine-grained complexity of one-dimensional dynamic programming. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *ICALP*, volume 80 of *LIPIcs*, pages 21:1–21:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.21.

26 Eduardo Sany Laber, Wilfredo Bardales Roncalla, and Ferdinando Cicalese. On lower bounds for the maximum consecutive subsums problem and the (min, +)-convolution. In *IEEE*, pages 1807–1811. IEEE, 2014. doi:10.1109/ISIT.2014.6875145.

27 Yves Lucet. Faster than the fast legendre transform, the linear-time legendre transform. *Numer. Algorithms*, 16(2):171–185, 1997. doi:10.1023/A:1019191114493.

28 George B. Mertzios, André Nichterlein, and Rolf Niedermeier. The power of linear-time data reduction for maximum matching. *Algorithmica*, 82(12):3521–3565, 2020. doi:10.1007/s00453-020-00736-0.

29 Mihai Patrascu. Towards polynomial lower bounds for dynamic problems. In Leonard J. Schulman, editor, *STOC*, pages 603–610. ACM, 2010. doi:10.1145/1806689.1806772.

30 Yaroslav Shitov. The complexity of tropical matrix factorization. *Advances in Mathematics*, 254:138–156, 2014.

31 Ryan Williams. Faster all-pairs shortest paths via circuit complexity. In David B. Shmoys, editor, *STOC*, pages 664–673. ACM, 2014. doi:10.1145/2591796.2591811.