On Minimizing Wiggle in Stacked Area Charts

Alexander Dobler
TU Wien, Austria

Martin Nöllenburg
TU Wien, Austria

Abstract

Stacked area charts are a widely used visualization technique for numerical time series. The x-axis represents time, and the time series are displayed as horizontal, variable-height layers stacked on top of each other. The height of each layer corresponds to the time series values at each time point. The main aesthetic criterion for optimizing the readability of stacked area charts is the amount of vertical change of the borders between the time series in the visualization, called *wiggle*. While many heuristic algorithms have been developed to minimize wiggle, the computational complexity of minimizing wiggle has not been formally analyzed. In this paper, we show that different variants of wiggle minimization are NP-hard and even hard to approximate. We also present an exact mixed-integer linear programming formulation and compare its performance with a state-of-the-art heuristic in an experimental evaluation. Lastly, we consider a special case of wiggle minimization that corresponds to the fundamentally interesting and natural problem of ordering a set of numbers as to minimize their sum of absolute prefix sums. We show several complexity results for this problem that imply some of the mentioned hardness results for wiggle minimization.

2012 ACM Subject Classification Mathematics of computing → Permutations and combinations; Theory of computation → Theory and algorithms for application domains; Theory of computation → Problems, reductions and completeness

Keywords and phrases Stacked area charts, NP-hardness, Mixed-integer linear programming

Digital Object Identifier 10.4230/LIPIcs.WADS.2025.22

Related Version *Full Version*: `https://arxiv.org/abs/2506.21175` [5]

Supplementary Material *Software (Source code and experimental evaluation)*: `https://doi.org/10.5281/zenodo.15745936`

Funding *Alexander Dobler*: Supported by the Vienna Science and Technology Fund (WWTF) under grant [10.47379/ICT19035].
Martin Nöllenburg: Supported by the Vienna Science and Technology Fund (WWTF) under grant [10.47379/ICT19035].

1 Introduction

Stacked area charts are a widely used method for visualizing numerical time series across discrete time points, such as population data of countries over time or movie revenues over multiple weeks [4, 9]. In stacked area charts, the x-dimension depicts time, and the time series are stacked as variable-height strips on top of each other without gaps, such that their heights depict their values (see Figure 1a). The lowest border of the stacked time series is a straight horizontal line.

A primary aesthetic criterion for stacked area charts is *wiggle* – the aggregated amount of vertical change of the borders between the time series in the visualization (Figure 1b). Further, the wiggle is usually weighted by the time series values, resulting in *weighted wiggle* (for a formal definition refer to Section 2). For a stacked area chart of a specific dataset, the amount of wiggle solely depends on the vertical ordering of the time series. The task of minimizing wiggle has already been tackled with heuristic methods [4, 13]. But despite the

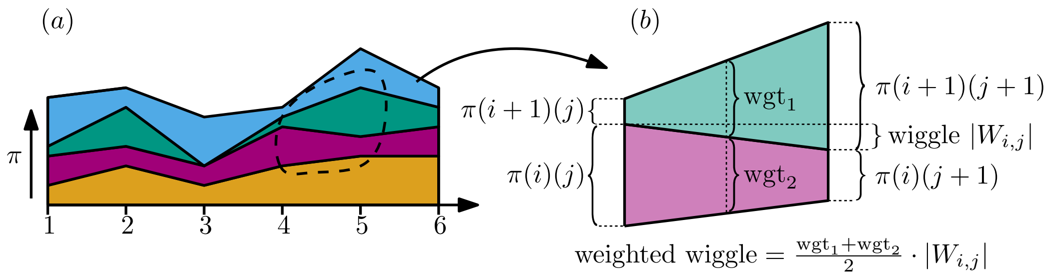


Figure 1 (a) A stacked area chart of four time series on six time points. The time series are ordered according to π from bottom to top. (b) Illustration of the wiggle between time points 4 and 5 on the border between the green and violet time series. For the weighted wiggle, $|W_{i,j}|$ is multiplied by the average of $\pi(i)(j)$, $\pi(i)(j+1)$, $\pi(i+1)(j)$, and $\pi(i+1)(j+1)$. The total (weighted) wiggle is the sum of all (weighted) wiggle values.

popularity of stacked area charts in mainstream visualizations, no work was done with regard to the computational complexity of minimizing wiggle. This paper fills the gap and presents several complexity lower bounds for minimizing wiggle and weighted wiggle. We also compare a state-of-the-art heuristic for minimizing weighted wiggle with a new mixed-integer linear programming formulation on a set of real-world data; the evaluation shows that the heuristic performs well with regard to wiggle minimization and that the mixed-integer program can solve small to medium-size instances exactly. Further, a special case of minimizing wiggle is discussed that results in a fundamentally interesting problem – ordering a set of numbers such that the accumulation of all absolute prefix sums is minimized. Despite its natural problem definition, the problem has not been studied yet. Its relevance comes from its relation to several ordering problems that minimize accumulated cost, such as scheduling problems. Results for this problem are also used to show some hardness results for minimizing wiggles.

Related work. The origin of stacked area charts is unclear, as they are a very natural and frequently used visualization for a commonly occurring type of data. The most notable work for popularizing research on stacked area charts is from 2008 by Byron and Wattenberg [4], since receiving more than 600 citations. Given the extensive literature on stacked area charts, we focus on studies specifically addressing wiggle minimization and similar aesthetic criteria in related visualizations. Byron and Wattenberg [4] only deal with data that represent movie revenues. Such data have very specific properties – a movie starts airing, its revenues grow quickly, and then the revenues slowly decrease. Thus, they present a very specific ordering approach that orders the movies by their first screening. They also consider a similar type of visualization, called *streamgraphs*. The only difference between streamgraphs and stacked area charts is that the lower border of a streamgraph does not have to be a horizontal line, introducing another degree of freedom. Still, wiggle minimization is defined equivalently for streamgraphs. Byron and Wattenberg present a similar ordering procedure for streamgraphs. Greffard and Kuntz [7] present an approach for weighted squared wiggle minimization in streamgraphs that first defines pairwise dissimilarity values between time series and then applies traveling salesperson approaches. Di Bartolomeo and Hu [2] present a heuristic and local search algorithm for weighted wiggle minimization in streamgraphs, which also work for stacked area charts. Bu et al. [3] present an ordering algorithm for streamgraphs that is based on clustering. They also optimize for minimizing what they call *sine illusion effect* – time series borders mimicking a sine wave. He and Li [8] present an ordering approach for stacked area charts based on a traveling salesperson formulation. Their approach aims to minimize the covariance between adjacent time series in the stacked area chart. Lastly, Mathiesen and Schulz [13] present a heuristic algorithm that is able to minimize a weighted sum of quality criteria for stacked area charts, including wiggle. Their approach is an improvement over an algorithm by Di Bartolomeo and Hu [2], and they demonstrate its effectiveness in an experimental evaluation.

The problem of ordering a set of numbers to minimize the accumulation of all absolute prefix sums is related to some classic problems from scheduling. Tsai [12] studies the problem where instead of minimizing the accumulation, one wants to minimize the maximum absolute prefix sum. They claim strong NP-hardness (a formal proof is absent). Kellerer et al. [10] consider the same problem as Tsai with the additional constraint that each prefix sum must be positive, and show constant-factor approximation algorithms. Further, they explicitly pose the open problem of minimizing the sum of all prefix sums. In their definition, though, the ordering has to satisfy that each prefix sum is positive, which is not the case for us.

Our contribution. We consider wiggle minimization and weighted wiggle minimization in stacked area charts from a computational complexity perspective. For this, we define the two computational problems p-WiggleMin and Weighted-p-WiggleMin in Section 2. The value p in both problem definitions corresponds to the exponent of each wiggle value in the objective function, in turn capturing variants of wiggle minimization such as the weighted squared wiggle minimization from Byron and Wattenberg [4]. Further, we introduce and investigate a special case of instances leading to the problem Min-$\sum$|Prefixsum| – ordering a set of numbers to minimize the sum of absolute prefix sums.

In Section 3, we show that Min-$\sum$|Prefixsum| is strongly NP-hard and that both wiggle problems with $p = 1$ are strongly NP-hard, even if the number of time points is constant. Next, we also consider special cases of Min-$\sum$|Prefixsum|, where there is only one positive or one negative element in the input. Section 4 shows that both wiggle problems for arbitrary p are strongly NP-hard, and hard to approximate under specific complexity assumptions. Further, a lower bound on the approximation ratio of a known greedy heuristic used in [2] and [13] is shown. Lastly, Section 5 presents a mixed-integer linear program for Weighted-1-WiggleMin and compares it with the state-of-the-art heuristic of Mathiesen and Schulz [13] in an experimental evaluation on real-world data.

Due to space constraints, statements marked with ★ are proved in the appendix.

2 Preliminaries and Problem Definitions

We define $[n] = \{1, \dots, n\}$. A permutation π of a multiset $S = \{s_1, \dots, s_n\}$ is a bijection $\pi : [n] \mapsto S$. We sometimes treat π as the list $(\pi(1), \pi(2), \dots, \pi(n))$. Further, we define $\text{pos}_\pi(s) = i$ if and only $\pi^{-1}(s) = i$.

An *ℓ-time series* f is a function $f : [\ell] \to \mathbb{R}_0^+$. We refer to the elements of its image as *data points.* A set F of ℓ-time series, is *balanced* if $\sum_{f \in F} f(i) = \sum_{f \in F} f(j)$ for all $i, j \in [\ell]$.

A computational problem is *strongly* NP*-hard* if it remains NP-hard even if its numerical parameters are integers that are polynomial in the input size.

Problem Definitions

In most problems discussed in this paper, we are given a set $F = \{f_1, \dots, f_n\}$ of ℓ-time series. Given a permutation π of F, $i \in \{0\} \cup [n]$, and $j \in [\ell - 1]$, we define the *wiggle* value $W^\pi_{i,j} = \sum_{k=1}^{i} (\pi(k)(j+1) - \pi(k)(j))$. This results in the following two problem variants of *wiggle minimization for stacked area charts*, the first unweighted, and the second weighted by the time series data points - as is more common in the stacked area charts literature [2, 7, 13] (see also Figure 1b). For both problems, p is a positive integer corresponding to the exponent of the wiggle values. The second problem is equivalent to the minimization of *flatness* in [13].

▶ **Problem 1** (p-WiggleMin)**.** Given a set $F = \{f_1, \ldots, f_n\}$ of ℓ-time series, find a permutation π of F such that $\sum_{i=1}^{n} \sum_{j=1}^{\ell-1} |W_{i,j}^{\pi}|^p$ is minimized.

▶ **Problem 2** (Weighted-p-WiggleMin)**.** Given a set $F = \{f_1, \ldots, f_n\}$ of ℓ-time series, find a permutation π of F such that

$$\sum_{i=1}^{n} \sum_{j=1}^{\ell-1} \frac{\pi(i)(j) + \pi(i)(j+1) + \pi(i+1)(j) + \pi(i+1)(j+1))}{4} |W_{i,j}^{\pi}|^p$$

is minimized. Above, $\pi(n+1)$ is the time series containing only zeroes.

If $\ell = 2$, then 1-WiggleMin is equivalent (see Lemma 4) to the following fundamentally interesting problem, which we also study in this paper.

▶ **Problem 3** (Min-$\sum$|PrefixSum|)**.** Given a multiset $S = \{s_1, \ldots, s_n\}$ of real numbers, find a permutation π of S such that $\sum_{i=1}^{n} \left| \sum_{j=1}^{i} \pi(j) \right|$ is minimized.

Given a solution π of Min-$\sum$|PrefixSum|, we define for $i = 0, 1, \ldots, n$, $P_i^{\pi} = \sum_{j=1}^{i} \pi(j)$.

Next, we want to show relationships between the previously defined problems.

▶ **Lemma 4** (★)**.** *Given an instance $S = \{s_1, \ldots, s_n\}$ of Min-$\sum$|PrefixSum|, there exists an instance $F = \{f_1, \ldots, f_n\}$ of 1-WiggleMin on two time points such that S has a solution with value x if and only if F has a solution with value x. If $\sum_{s \in S} s = 0$, then F is balanced. Further, data point values in F are bounded by a polynomial of the values in S.*

Conversely, for an instance F of 1-WiggleMin on two time points, there exists an instance S of Min-$\sum$|PrefixSum| such that S has a solution with value x if and only if F has a solution with value x.

▶ **Lemma 5** (★)**.** *Let $p \geq 1$ be an integer. Given a balanced instance $F = \{f_1, \ldots, f_n\}$ of p-WiggleMin on ℓ time points, there is an instance $F' = \{f_1', \ldots, f_n'\}$ of Weighted-p-WiggleMin on $4\ell + 1$ time points and a constant C such that F has a solution with value x if and only if F' has a solution with value Cx. Further, the values of the data points in F' are polynomially bounded w.r.t. the values of the data points in F.*

A trivial algorithm for the mentioned problems would need to go over all permutations, and thus need $\mathcal{O}(n!|I|)$ time when $|I|$ is the instance size. A simple improvement is to use dynamic programming by computing the optimal solution for each subset of time series (or elements in S), leading to an algorithm needing time $\mathcal{O}(2^n |I|^c)$ for some constant c. In the following sections, we want to find out whether polynomial time exact, or approximation algorithms for the problems are likely.

3 Complexity of Min-$\sum$|PrefixSum| and (Weighted)-1-WiggleMin

In this section, we consider the problem Min-$\sum$|PrefixSum|. We will show several complexity results, some of which will imply results for 1-WiggleMin and Weighted-1-WiggleMin.

3.1 Strong NP-hardness

Most of this section is devoted to showing strong NP-hardness of Min-$\sum$|PrefixSum|. The proof idea is to reduce instances of a known NP-hard problem to instances S of Min-$\sum$|PrefixSum|, such that S has a solution value which is equal to a lower bound if and only if the instance which we reduced from is a yes instance. This lower bound is stated next.

▶ **Lemma 6.** *A lower bound for the objective function of* Min-$\sum$|Prefixsum| *is* $\sum_{s\in S}|s|/2$.

Proof. Let π be some solution of Min-$\sum$|Prefixsum|. It is easy to see that for $i = 1, \ldots, n$,

$$|P_i^\pi| + |P_{i-1}^\pi| \geq |\pi(i)|.$$

Thus,

$$2\sum_{i=0}^{n}|P_i^\pi| \geq |P_0^\pi| + |P_n^\pi| + 2\sum_{i=1}^{n-1}|P_i^\pi| \geq \sum_{i=1}^{n}|s_i|$$ ◀

We call a solution π achieving this bound *bound-achieving*. The next lemma states a sufficient condition that implies that the solution value exceeds the previously stated lower bound.

▶ **Lemma 7.** *Let π be a solution of* Min-$\sum$|Prefixsum| *and $\sum_{i=1}^{n} s_i = 0$. There exists $j \in [n]$ such that $|P_j^\pi| + |P_{j-1}^\pi| > |\pi(j)|$ if and only if $\sum_{i=1}^{n}|P_i^\pi| > \sum_{i=0}^{n}|s_i|/2$.*

Proof. The backwards direction is clear. For the forward direction, we have

$$2\sum_{i=0}^{n}|P_i^\pi| = 2\sum_{i=1}^{n-1}|P_i^\pi| > \sum_{i=1}^{n}|s_i|.$$ ◀

Additionally, in our reduction, we will consider instances S of Min-$\sum$|Prefixsum| that have specific properties. These properties are described below.
(IP1) $\sum_{i=1}^{n} s_i = 0$.
(IP2) $n \equiv 0 \pmod 3$, i.e., n is a multiple of 3.
(IP3) S has exactly $\frac{2n}{3}$ positive elements and $\frac{n}{3}$ negative elements.
Hence, S also has no elements equal to zero. We call an instance that satisfies **(IP1)-(IP3)** *nice*, and observe the following properties that will be useful for the reduction.

▶ **Lemma 8.** *Let π be a bound-achieving solution of a nice instance S. At least one of $\pi(1), \pi(2)$ is negative and at least one of $\pi(n-1), \pi(n)$ is negative.*

Proof. If both $\pi(1), \pi(2)$ are positive, then $\pi(2) < |P_1^\pi| + |P_2^\pi|$. Lemma 7 gives us a contradiction, as we assumed that π is bound-achieving.

If both $\pi(n-1), \pi(n)$ are positive, then we have, as $P_n^\pi = 0$, that $P_{n-1}^\pi = -\pi(n)$ and $P_{n-2}^\pi = -(\pi(n) + \pi(n-1))$. It follows that $|P_{n-2}^\pi| + |P_{n-1}^\pi| > |\pi(n-1)|$, a contradiction. ◀

▶ **Lemma 9.** *Let π be a bound-achieving solution of a nice instance S. If for $i \in [n-1]$, $\pi(i)$ and $\pi(i+1)$ are both positive or both negative, then $P_i^\pi = 0$.*

Proof. If $\pi(i)$ and $\pi(i+1)$ are both positive, and $P_i^\pi \neq 0$ then we have two cases.
- If $P_i^\pi > 0$ then $|P_i^\pi| + |P_{i+1}^\pi| > \pi(i+1)$, a contradiction.
- If $P_i^\pi < 0$ then $|P_{i-1}^\pi| + |P_i^\pi| > \pi(i)$, a contraction.

When both $\pi(i)$ and $\pi(i+1)$, are negative, the proof works the same. ◀

The following is a direct consequence of the last lemma.

▶ **Corollary 10.** *Let π be a bound-achieving solution of a nice instance S. There exists no $i \in [n-2]$ such that $\pi(i), \pi(i+1), \pi(i+2)$ are all negative or all positive.*

With these lemmas out of the way, we can show that π has a unique structure with regard to the signs of its elements.

▶ **Lemma 11.** *Let π be a bound-achieving solution of a nice instance S. Then, for $i \in [n]$,*

- *$\pi(i)$ is positive if and only if $i \equiv 0 \pmod 3$ or $i \equiv 1 \pmod 3$,*
- *$\pi(i)$ is negative if and only if $i \equiv 2 \pmod 3$, and*
- *if $i \equiv 0 \pmod 3$, then $P_i^\pi = 0$.*

Proof. We show the claim by induction on i. For the base case, $i = 1$, assume for a contradiction that $\pi(1)$ is negative. Together with Lemma 8 and the pigeonhole principle, it follows that there must be three consecutive positive elements, a contradiction to Corollary 10.

For the induction step $i > 1$, we have the following cases.

- If $i \equiv 0 \pmod 3$ then $\pi(i)$ must be positive; otherwise, for the remaining $n - i$ following elements, we have $(n-i)/3 - 1$ negative elements and $2(n-i)/3 + 1$ positive elements. By the pigeonhole principle, there are three consecutive positive elements, a contradiction. Further $P_i^\pi = 0$ must hold: For $i = n$ it holds trivially. Otherwise, if P_i^π was not zero, then $\pi(i+1)$ must be negative by Lemma 9. For there to be no three consecutive positive elements in the remaining array, $\pi(n-1)$ and $\pi(n)$ must be positive, a contradiction.
- If $i \equiv 1 \pmod 3$ then $\pi(i)$ is positive by the same argumentation as in the base case.
- If $i \equiv 2 \pmod 3$ then $\pi(i)$ is clearly negative; otherwise, as P_{i-1}^π is positive, $|P_i^\pi| + |P_{i+1}^\pi| > |\pi(i)|$ would hold; a contradiction (see Lemma 7). ◀

With this groundwork, we are able to show the following result.

▶ **Theorem 12.** *The decision variant of* Min-$\sum$|Prefixsum| *is strongly* NP-*complete.*

Proof. NP-membership is obvious, thus we continue with NP-hardness. We reduce from the strongly NP-hard problem Numerical 3-Dimensional Matching [17]. The problem input consists of three multisets X, Y, and Z, each containing k integers. Let $b = (\sum_{e \in X \cup Y \cup Z} e)/k$. The problem is to decide whether there exists a subset M of $X \times Y \times Z$ such that

1. each element from X, Y, and Z appears in exactly one triple, and
2. $x + y + z = b$ holds for each triple $(x, y, z) \in M$.

We can assume that each element from $X \cup Y \cup Z$ is positive (otherwise add to all elements $-\alpha + 1$, where $\alpha = \min X \cup Y \cup Z$ and obtain an equivalent instance with regard to the solution). We also assume that each element from X, Y, and Z is smaller than b. Now define the multisets $X' = \{x \mid x \in X\}$, $Y' = \{y + 2b \mid y \in Y\}$, and $Z' = \{z - 3b \mid z \in Z\}$. Let $n = |X' \cup Y' \cup Z'|$. Notice that $\sum_{e \in X' \cup Y' \cup Z'} e = 0$, and X', Y' and Z' are pairwise disjoint. We now define the instance S of Min-$\sum$|Prefixsum| as $S = X' \cup Y' \cup Z'$. Notice that S is a nice instance satisfying **(IP1)-(IP3)**.

We claim that (X, Y, Z) is a yes-instance of Numerical 3-Dimensional Matching if and only if there exists a solution π of S that is bound-achieving. We prove both directions.

"$\Rightarrow$". Let M be a subset of $X \times Y \times Z$ constituting a solution to the instance of Numerical 3-Dimensional Matching. We constructively define π by going through the triples (x, y, z) in M in any order. Let (x, y, z) be the ith triple. We define $\pi(3i-2) = x, \pi(3i-1) = z - 3b$, and $\pi(3i) = y + 2b$. By the definition of X', Y', and Z' such a permutation always exists. Now notice for $j = 0, 1, \ldots, n$ that P_j^π is

- 0, if and only if $j \equiv 0 \pmod 3$,
- positive if and only if $j \equiv 1 \pmod 3$, and
- negative if and only if $j \equiv 2 \pmod 3$.

Thus, we have $|P_j^\pi| + |P_{j-1}^\pi| = |\pi(j)|$ for all $j \in [n]$. It follows that $\sum_{j=1}^n |P_j^\pi| = \sum_{s \in S} |s|/2$.

"$\Leftarrow$". Assume that π is bound-achieving. We construct M, starting with $M = \emptyset$. Because S is nice, we have that Lemma 11 holds for π. For each $i \in [k]$ we have the following. Let $u := \pi(i-2), v := \pi(i-1)$, and $w := \pi(i)$. Because of Lemma 11 we have that $u+v+w = 0$, v is negative, and u and w are positive. Hence, $v \in Z'$ holds. Now we claim that exactly one of u and w is from X' and exactly one is from Y'. Indeed, if both u and w were from X', then $u + w < 2b < |v|$ and $u + v + w = 0$ is impossible. Similarly, if both u and w were from Y', then $u + w > 4b > |v|$ and $u + v + w = 0$ is impossible. Thus, w.l.o.g. assume that $u \in X'$ and $w \in Y'$. Add to M the triple $(u, w - 2b, v + 3b)$. It is easy to verify that after adding this triple for each $i \in [k]$, M verifies that (X, Y, Z) is a yes-instance of NUMERICAL 3-DIMENSIONAL MATCHING. ◀

Lemma 4 and Lemma 5 imply the following corollary.

▶ **Corollary 13.** *The decision variants of* 1-WIGGLEMIN *and* WEIGHTED-1-WIGGLEMIN *are strongly* NP*-complete, even for a constant number of time points.*

3.2 Restricted instances of MIN-$\sum$|PREFIXSUM|

We further ask for which types of instances MIN-$\sum$|PREFIXSUM| becomes tractable. In particular, we want to know whether restricting the number of negative (or positive) elements of S has an effect on the computational complexity of MIN-$\sum$|PREFIXSUM|. Indeed, if S only contains positive or only negative elements, then an optimal ordering π will simply order the elements of S by their absolute value. But what if S contains exactly one positive or exactly one negative element? We obtain the following peculiar results, which state that the sum of S has an effect on the complexity of MIN-$\sum$|PREFIXSUM| in this case.

▶ **Theorem 14** (★)**.** MIN-$\sum$|PREFIXSUM| *can be solved in time $\mathcal{O}(n \log n)$ if S contains exactly one negative (positive) element, and $\sum_{i=1}^{n} s_i = 0$.*

▶ **Theorem 15** (★)**.** *The decision variant of* MIN-$\sum$|PREFIXSUM| *is* NP*-complete, even if S contains exactly one negative (positive) element.*

We dedicate the rest of the section to show both results, starting with Theorem 14.

Proof sketch of Theorem 14. We show the result for S containing a single negative element x, the result for a single positive element is shown equivalently. Now consider some permutation π of S, and let $k = \mathrm{pos}_\pi(x)$. We observe that the objective value can be split into absolute prefix sums before and after position k as follows.

$$\begin{aligned}\sum_{i=1}^{n} \Big|\sum_{j=1}^{i} \pi(j)\Big| &= \sum_{i=1}^{k-1}\sum_{j=1}^{i} \pi(j) + \sum_{i=1}^{n-k}\sum_{j=1}^{i} \pi(n-j+1)\\ &= \sum_{i=1}^{k-1} \pi(i)\cdot(k-i) + \sum_{i=1}^{n-k} \pi(n-i+1)\cdot(n-k-i+1).\end{aligned}$$

Thus, we obtain a weighted sum of the positive elements. The coefficient of an element depends only on its position in relation to the position of the negative element. This leads to a simple algorithm, which is also given as pseudocode in the long version of this paper: The permutation is chosen such that the negative element is in the middle. The smallest positive elements are placed at positions 1 and n, the next smallest at positions 2 and $n-1$, etc. ◀

To show Theorem 15, we reduce from the following problem.

▶ **Problem 16** (OneInTwoPartition). Given is a set X containing m pairs of positive integers. No integer appears twice in the union of all the integers. The question is: Are there two sets X_1, X_2 of size m such that both sets contain exactly one element from each pair in X, no element is contained in both sets, and $\sum_{x\in X_1} x = \sum_{x\in X_2} x$?

Clearly, OneInTwoPartition is NP-hard by a straightforward reduction from Partition [6]: Let $Y = \{y_1, \ldots, y_n\}$ be an instance of Partition, asking for a subset $Y' \subseteq Y$ with $\sum_{y\in Y'} y' = \frac{1}{2}\sum_{y\in Y} y$. We can assume all integers in Y to be positive. Let $b = \sum_{y\in Y} y$. The reduced instance consists of the pairs $(i \cdot b + y_i, i \cdot b)$ for $i = 1, \ldots, n$. The correctness of this reduction is immediate. We are ready to prove Theorem 15.

Proof sketch of Theorem 15. Note that the proof is rather technical; to understand it in full detail, the interested reader should refer to the full description in full version of the paper. We reduce from OneInTwoPartition. Consider an instance $X = \{p_1, p_2, \ldots, p_m\}$ of OneInTwoPartition, where $m > 1$. Define $M = \sum_{(x,y)\in X} \frac{x+y}{2}$. We define the following three sets.

$$S_1 = \{m(x + iM) \mid i \in [m], p_i = (x, y)\}$$
$$S_2 = \{m(y + iM) \mid i \in [m], p_i = (x, y)\}$$
$$S_3 = \{imM + 1 \mid i \in [m]\}$$

We continue with further definitions.

$$M_1 = \frac{\sum_{s\in S_1} s + \sum_{s\in S_2} s}{2} = mM + mM\sum_{i=1}^{m} i$$
$$M_2 = \sum_{s\in S_3} s = m + mM\sum_{i=1}^{m} i$$
$$S = S_1 \cup S_2 \cup S_3 \cup \{-(M_1 + M_2)\}$$

For each $(x, y) \in X$ there are unique values $x', y' \in S$ such that for some i, $x' = m(x + iM)$ and $y' = m(y + iM)$. We say that x' is the *correspondent* of x and y' is the *correspondent* of y, and vice versa. We observe that $S \setminus \{-(M_1 + M_2)\}$ can be partitioned uniquely into sets $t_1, t_2, \ldots, t_m$ such that for each t_i we have that

- t_i contains exactly three elements,
- t_i contains one element from S_1, one from S_2, and one from S_3,
- the correspondent of $\{x\} = S_1 \cap t_i$ and the correspondent of $\{y\} = S_2 \cap t_i$ form a pair in X, and
- each integer from t_i is in the interval $[imM, (i + 1)mM)$.

We set S as the instance of Min-$\sum$|Prefixsum| and claim that X is a yes-instance of OneInTwoPartition if and only if there exists a permutation π of S such that

$$\sum_{i=1}^{n} |\sum_{j=1}^{i} \pi(j)| \le \sum_{i=1}^{m}(m - i + 1)\sum_{x\in t_i} x.$$

We sketch the more interesting direction of the proof, assuming that π is a solution achieving this objective, showing how to construct X_1 and X_2. The idea now is that π will order the elements as shown in Figure 2. Namely, the prefix sum P_{2m+1} will be zero, and the last m elements will be ordered increasingly. Further, $\pi(2m + 1 + i)$ will be from set t_i for each $i \in [n]$. This further implies that the last m elements sum to M_1, which is a

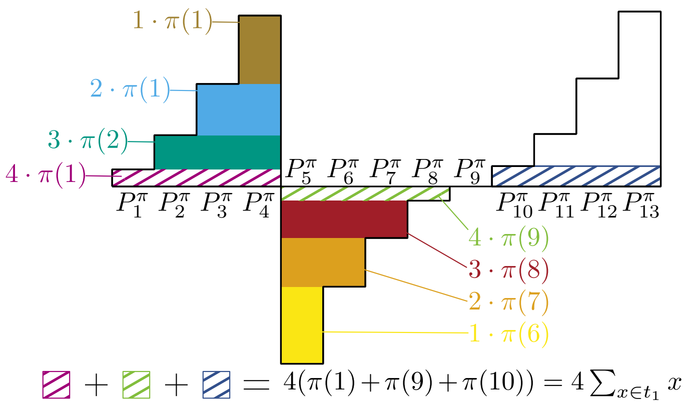

$1 \cdot \pi(1)$
$2 \cdot \pi(1)$
$3 \cdot \pi(2)$
$4 \cdot \pi(1)$
P_1^π P_2^π P_3^π P_4^π
P_5^π P_6^π P_7^π P_8^π P_9^π
P_{10}^π P_{11}^π P_{12}^π P_{13}^π
$4 \cdot \pi(9)$
$3 \cdot \pi(8)$
$2 \cdot \pi(7)$
$1 \cdot \pi(6)$
▨ + ▨ + ▨ $= 4(\pi(1)+\pi(9)+\pi(10)) = 4\sum_{x \in t_1} x$

Figure 2 Schematization of the distribution of prefix sums for an optimal solution π of S in the proof of Theorem 15, where $m = 4$. The height corresponds to the prefix sum value, while values below the horizontal middle line are negative. The figure illustrates how the objective value (the area of the union of all enclosed regions) can be decomposed into a weighted sum of the positive elements. Further, the three smallest elements at positions $1, 9$, and 10 contribute to the sum with coefficient 4, resulting in the hatched regions. In an optimal solution, these elements correspond to the three smallest values, constituting t_1.

multiple of m. Together, this means that the last m elements cannot contain any elements from S_3. Now let X_1 be the set of correspondents of the last m elements in π, and let X_2 be the remaining elements from X. The above properties imply that $\sum_{x \in X_1} x = \sum_{x \in X_2} x$. ◀

Note that a pseudo-polynomial algorithm for MIN-$\sum$|PREFIXSUM| cannot be ruled out for the case of one negative (positive) element, as we did not show strong NP-hardness.

4 p-WIGGLEMIN and WEIGHTED-p-WIGGLEMIN

In this section, we show hardness result for p-WIGGLEMIN and WEIGHTED-p-WIGGLEMIN for arbitrary p. We present a reduction which has several complexity implications in Section 4.1. Section 4.2 contains results with regard to approximation lower bounds.

4.1 A Reduction Implying NP-hardness

The reduction is from the well-known problem MINIMUM LINEAR ARRANGEMENT [6] to p-WIGGLEMIN. The problem is defined as follows.

▶ **Problem 17** (MINIMUM LINEAR ARRANGEMENT (MLA))**.** Given an undirected graph $G = (V, E)$ and an integer k, does there exist a permutation π of V such that

$$\sum_{\{u,v\} \in E} |\mathrm{pos}_\pi(u) - \mathrm{pos}_\pi(v)| \leq k?$$

Consider a constant positive integer p. The reduction will work for any such p, and goes as follows. Let (G, k) be an instance of MLA with $V = \{v_1, \ldots, v_n\}$ and $E = \{e_1, \ldots, e_m\}$. We construct a set $F = \{f_1, \ldots, f_n\}$ of $(m+1)$-time series. We define the time series inductively based on the time point, starting with the first time point: let $f_i(1) = m$ for all $i = 1, \ldots n$. For time point $j > 1$ consider the edge $e_{j-1} = \{v_a, v_b\}$ with $a < b$. We define $f_i(j)$ as

$$f_i(j) = \begin{cases} f_i(j-1)+1 & \text{if } i = a, \\ f_i(j-1)-1 & \text{if } i = b, \\ f_i(j-1) & \text{otherwise.} \end{cases} \tag{1}$$

Now consider an arbitrary permutation π^V of V. Let π^F be a permutation of F that is defined such that $\mathrm{pos}_{\pi^V}(v_i) = \mathrm{pos}_{\pi^F}(f_i)$ for all $i \in [n]$. We have the following.

▶ **Lemma 18 (★).** *It holds that*

$$\sum_{i=1}^{n}\sum_{j=1}^{m} |W_{i,j}^{\pi^F}|^p = \sum_{\{u,v\}\in E} |pos_\pi(u) - pos_\pi(v)|. \tag{2}$$

Thus, a solution π^V with solution value k w.r.t. MLA corresponds to a solution π^F with solution value k w.r.t. p-WiggleMin. Hence, hardness and inapproximability results of MLA directly carry over to p-WiggleMin. Further, the instance F is balanced, so results also carry over to Weighted-p-WiggleMin by Lemma 5. We obtain the following by the NP-hardness of MLA [6].

▶ **Theorem 19.** *Let $p \geq 1$ be an arbitrary integer. The decision variants of p-WiggleMin and Weighted-p-WiggleMin are strongly NP-complete.*

4.2 Approximation Lower Bounds

In this section, we consider the complexity of approximating p-WiggleMin and Weighted-p-WiggleMin. Firstly, the reduction from the previous section implies two hardness results due to hardness results for MLA shown Ambühl et al. [1] and Raghavendra et al. [15].

▶ **Theorem 20.** *Let $p \geq 1$ be an arbitrary integer. Let $\epsilon > 0$ be an arbitrarily small constant. If there is a PTAS for p-WiggleMin or for Weighted-p-WiggleMin, then there is a (probabilistic) algorithm that decides whether a given SAT instance of size n is satisfiable in time 2^{n^ϵ}.*

Note that it is widely believed that such an algorithm for SAT is unlikely, thus it is unlikely that there exists a PTAS for both problems.

▶ **Theorem 21.** *Let $p \geq 1$ be an arbitrary integer. Under the Small-Set Expansion Hypothesis [14], there is no constant-factor approximation for p-WiggleMin and Weighted-p-WiggleMin.*

The Small-Set Expansion Hypothesis is a hypothesis that implies the Unique-Games Conjecture [11,14]. It was introduced by Raghavendra and Steurer in 2010 [14] and has since received much attention, as this conjecture would imply some hardness and inapproximability results, including the non-existence of a constant-factor approximation for treewidth [16].

A known greedy heuristic. Next, we want to look at a known greedy heuristic called **BestFirst** for Weighted-1-WiggleMin that has been applied in works on stacked area charts [2,13]. Given an instance $F = \{f_1, \ldots, f_n\}$ of Weighted-1-WiggleMin, the heuristic iteratively builds the order π, starting from the empty order. At step $i = 1, \ldots, n$ of the heuristic, the partial ordering of length $i-1$ is extended by appending a not yet chosen time series to the end that has the smallest increase in the objective value. Such heuristics equivalently exist for 1-WiggleMin and Min-$\sum$|PrefixSum|, we also call them **BestFirst**.

▶ **Theorem 22 (★).** *The* **BestFirst** *heuristic has approximation factor at least $\Omega(\sqrt[3]{n})$ for Weighted-1-WiggleMin, 1-WiggleMin, and Min-$\sum$|PrefixSum|.*

5 MILP and Experiments for Weighted-1-WiggleMin

Here, we present a mixed-integer linear program (MILP) for Weighted-1-WiggleMin and compare it to the state-of-the art heuristic from Mathiesen and Schulz [13]. This heuristic is different from the **BestFirst** heuristic from the last section and will be explained in more detail later. The aim of this comparison is to determine how an exact approach for Weighted-p-WiggleMin scales with respect to input size and to estimate how the heuristic of [13] compares with an exact algorithm with respect to solution quality. Note that the heuristic we compare with is strictly better than **BestFirst** as was demonstrated in an experimental evaluation by Mathisen and Schulz [13]. Due to space constraints, the description of the MILP can be found in the full version.

5.1 Experimental Setup

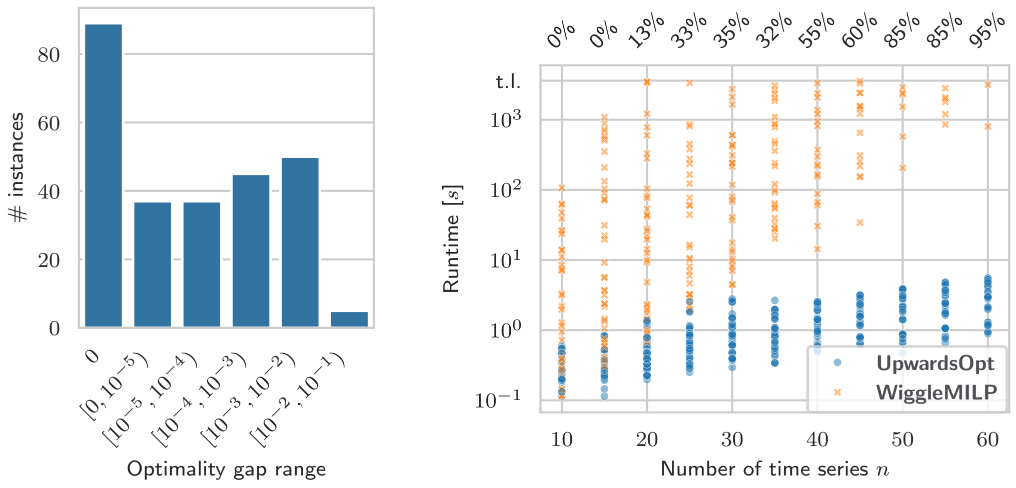
Tested algorithms. We compared two algorithms for Weighted-1-WiggleMin. The first, **WiggleMILP**, is an implementation of the MILP. The second, **UpwardsOpt**, is the best state-of-the art algorithm from [13]. It iteratively performs *moves* that improve the objective value. In a single move, a time series is removed from the current ordering and reinserted into the position that is best with regard to the objective value.

Instances. We obtained instances from real-world data, and these instances were also used in [13]. For example, the eight instances from [13] include unemployment rates of countries for a span of months, Covid values of countries for a span of days, and movie revenues for a span of weeks. These eight instances are very large, with 33–1,000 time series and 33–243 time points. We cannot expect that such instances can be solved to optimality by **WiggleMILP**, thus we sampled from these instances sub-instances as follows. For each instance I and each $k = 10, 15, \ldots, 60$, we picked uniformly at random k time series from I (if I contains at least k time series). These k time series then constitute an instance. We did this five times for each combination of k and I, resulting in 465 instances overall.

Hardware and setup. Both algorithms were implemented in Python 3.12.4. We used the original implementation of **UpwardsOpt** from [13] with the same parameters. **WiggleMILP** was implemented using the Python interface of Gurobi, using Gurobi v11.0.2. Due to large floating point values in the computations, the `NumericFocus` parameter of Gurobi was set to 3. The `MIPGap` parameter was set to 0, in order to find optimal solutions. Without this setup we ran into numeric issues in preliminary experiments. The experiments were run on a cluster using Intel Xeon E5-2640 v4, 2.40GHz 10-core processors, running Ubuntu 18.04.6 LTS. To simulate an end-user machine, the memory limit was set to 8GB and each algorithm was executed on a single thread. The time limit was set to one hour.

5.2 Results

In this section, we answer two questions: first, how does **UpwardsOpt** compare to **WiggleMILP** with respect to the solution quality; second, how scalable are both approaches with respect to the size of an instance. For the first question, we only considered 263 out of 465 instances where **WiggleMILP** produced the optimal solution – it neither timed out nor ran into the memory limit. For each instance where **WiggleMILP** produced the optimal solution, we compute the *optimality gap* g that is defined as $g = \mathrm{sol}_{\mathsf{UpwardsOpt}}/\mathrm{sol}_{\mathsf{WiggleMILP}} - 1$, where sol_x is the solution value of the algorithm x. Thus, a value of zero corresponds to **UpwardsOpt**



(a) Number of instances by the optimality gap achieved by **UpwardsOpt**.

(b) Scatter plot of runtimes by the number of time series n. The numbers at the top represent the percentage of instances with the respective value n that timed out for **WiggleMILP**.

Figure 3 Results of the experiments.

computing the optimal solution and larger values represent "how far away" **UpwardsOpt** is from the optimum. Figure 3a shows the distribution of optimality gaps. We observe that **UpwardsOpt** could solve 89 out of the 263 instances optimally, while the remaining optimality gaps are relatively low. This confirms that that **UpwardsOpt** is a good heuristic for WEIGHTED-p-WIGGLEMIN for real-world instances.

Next, in Figure 3b we present a scatter plot of runtimes. The y-axis shows runtime in a logscale. The x-axis corresponds to the instance size in terms of the number of time series. Each point in the plot corresponds to an instance-algorithm combination. The values at the top denote the percentage of instances with respective value n that timed out for **WiggleMILP** (the memory limit was never reached). There were no timeouts for **UpwardsOpt**. We observe that **UpwardsOpt** is relatively fast, while **WiggleMILP** runs for much longer and times out already for some instances with 20 time series.

Generally, we conclude that the state-of-the art heuristic **UpwardsOpt** is well-suited for WEIGHTED-1-WIGGLEMIN for real-world instances, even though the problem is computationally very hard in the theoretical sense.

6 Conclusion and Open Problems

We have investigated variants of wiggle minimization in stacked area charts from the theoretical side and showed computational lower bounds. Not only is it NP-hard to minimize wiggle, but it is also unlikely that approximations with good approximation guarantees exist. Nonetheless, we could show in an experimental evaluation that an existing heuristic is very good at minimizing weighted wiggle for real-world instances. Due to this being the first theoretical work on minimizing wiggle in stacked area charts, there remain some open problems.

- Are there constant-factor approximations for MIN-$\sum$|PREFIXSUM|?
- Are there stronger inapproximability results for p-WIGGLEMIN and WEIGHTED-p-WIGGLEMIN, i.e., under the assumption that P $\neq$ NP.

- We think that improvements to **WiggleMILP** are possible, and it might be interesting to investigate other exact wiggle minimization approaches.
- Finally, our results might be extended to wiggle minimization in streamgraphs. It seems likely that the hardness results in this paper carry over to streamgraphs, though, we were not able to show that yet.

References

1 Christoph Ambühl, Monaldo Mastrolilli, and Ola Svensson. Inapproximability results for maximum edge biclique, minimum linear arrangement, and sparsest cut. *SIAM J. Comput.*, 40(2):567–596, 2011. doi:10.1137/080729256.

2 Marco Di Bartolomeo and Yifan Hu. There is more to streamgraphs than movies: Better aesthetics via ordering and lassoing. *Comput. Graph. Forum*, 35(3):341–350, 2016. doi:10.1111/CGF.12910.

3 Chuan Bu, Quanjie Zhang, Qianwen Wang, Jian Zhang, Michael Sedlmair, Oliver Deussen, and Yunhai Wang. Sinestream: Improving the readability of streamgraphs by minimizing sine illusion effects. *IEEE Trans. Vis. Comput. Graph.*, 27(2):1634–1643, 2021. doi:10.1109/TVCG.2020.3030404.

4 Lee Byron and Martin Wattenberg. Stacked graphs - geometry & aesthetics. *IEEE Trans. Vis. Comput. Graph.*, 14(6):1245–1252, 2008. doi:10.1109/TVCG.2008.166.

5 Alexander Dobler and Martin Nöllenburg. On minimizing wiggle in stacked area charts. *CoRR*, abs/2506.21175, 2025. doi:10.48550/arXiv.2506.21175.

6 M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

7 Nicolas Greffard and Pascale Kuntz. Visualizing a set of multiple time series with an aggregate stacked graph. In Ebad Banissi, Mark W. McK. Bannatyne, Fatma Bouali, Remo Burkhard, John Counsell, Urska Cvek, Martin J. Eppler, Georges G. Grinstein, Weidong Huang, Sebastian Kernbach, Chun-Cheng Lin, Feng Lin, Francis T. Marchese, Chi Man Pun, Muhammad Sarfraz, Marjan Trutschl, Anna Ursyn, Gilles Venturini, Theodor G. Wyeld, and Jian J. Zhang, editors, *Proc. International Conference on Information Visualisation (IV'2015)*, pages 68–74. IEEE Computer Society, 2015. doi:10.1109/IV.2015.23.

8 Yutian He and Hongjun Li. Optimal layout of stacked graph for visualizing multidimensional financial time series data. *Inf. Vis.*, 21(1):63–73, 2022. doi:10.1177/14738716211045005.

9 Renée Huggett. Multiple line graphs (2). In *Graphs and Charts*, pages 43–46. Palgrave Macmillan UK, London, 1990. doi:10.1007/978-1-349-11245-6_10.

10 Hans Kellerer, Vladimir Kotov, Franz Rendl, and Gerhard J. Woeginger. The stock size problem. *Oper. Res.*, 46(3-Supplement-3):S1–S12, 1998. doi:10.1287/OPRE.46.3.S1.

11 Subhash Khot. On the power of unique 2-prover 1-round games. In John H. Reif, editor, *Proc. ACM Symposium on Theory of Computing (STOC'2002)*, pages 767–775. ACM, 2002. doi:10.1145/509907.510017.

12 Tsai Li-Hui. Sequencing to minimize the maximum renewal cumulative cost. *Oper. Res. Lett.*, 12(2):117–124, 1992. doi:10.1016/0167-6377(92)90073-C.

13 Steffen Strunge Mathiesen and Hans-Jörg Schulz. Aesthetics and ordering in stacked area charts. In Amrita Basu, Gem Stapleton, Sven Linker, Catherine Legg, Emmanuel Manalo, and Petrucio Viana, editors, *Proc. Diagrammatic Representation and Inference (Diagrams'2021)*, volume 12909 of *Lecture Notes in Computer Science*, pages 3–19. Springer, 2021. doi:10.1007/978-3-030-86062-2_1.

14 Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In Leonard J. Schulman, editor, *Proc. ACM Symposium on Theory of Computing (STOC'2010)*, pages 755–764. ACM, 2010. doi:10.1145/1806689.1806792.

15 Prasad Raghavendra, David Steurer, and Madhur Tulsiani. Reductions between expansion problems. In *Proc. Conference on Computational Complexity (CCC'2012)*, pages 64–73. IEEE Computer Society, 2012. doi:10.1109/CCC.2012.43.

16 Yu (Ledell) Wu, Per Austrin, Toniann Pitassi, and David Liu. Inapproximability of treewidth and related problems. *J. Artif. Intell. Res.*, 49:569–600, 2014. doi:10.1613/JAIR.4030.

17 Wenci Yu, Han Hoogeveen, and Jan Karel Lenstra. Minimizing makespan in a two-machine flow shop with delays and unit-time operations is NP-hard. *J. Sched.*, 7(5):333–348, 2004. doi:10.1023/B:JOSH.0000036858.59787.C2.