Optimizing Wiggle in Storylines

Alexander Dobler
TU Wien, Austria

Tim Hegemann
Universität Würzburg, Germany

Martin Nöllenburg
TU Wien, Austria

Alexander Wolff
Universität Würzburg, Germany

Abstract

A *storyline visualization* shows interactions between characters over time. Each character is represented by an x-monotone curve. Time is mapped to the x-axis, and groups of characters that interact at a particular point t in time must be ordered consecutively in the y-dimension at $x = t$. The predominant objective in storyline optimization so far has been the minimization of crossings between (blocks of) characters. Building on this work, we investigate another important, but less studied quality criterion, namely the minimization of *wiggle*, i.e., the amount of vertical movement of the characters over time.

Given a storyline instance together with an ordering of the characters at any point in time, we show that *wiggle count minimization* is NP-complete. In contrast, we provide algorithms based on mathematical programming to solve *linear wiggle height minimization* and *quadratic wiggle height minimization* efficiently. Finally, we introduce a new method for routing character curves that focuses on keeping distances between neighboring curves constant as long as they run in parallel.

We have implemented our algorithms, and we conduct a case study that explores the differences between the three optimization objectives. We use existing benchmark data, but we also present a new use case for storylines, namely the visualization of rolling stock schedules in railway operation.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Human-centered computing → Graph drawings

Keywords and phrases Storyline visualization, wiggle minimization, NP-complete, linear programming, quadratic programming, experimental analysis

Digital Object Identifier 10.4230/LIPIcs.GD.2025.39

Related Version *Full Version*: `https://arxiv.org/abs/2508.19802` [7]

Supplementary Material *Software (Source Code)*: `https://github.com/hegetim/narrativiz` [14] archived at `swh:1:dir:a7efa9c0ca674b6cf25b1c4b2c1f4793cc814723`

Funding *Alexander Dobler*: Vienna Science and Technology Fund (WWTF) grant [10.47379/ICT19035]. *Tim Hegemann*: Federal Ministry of Research, Technology and Space (BMFTR) grant [01IS22012C]. *Martin Nöllenburg*: Vienna Science and Technology Fund (WWTF) grant [10.47379/ICT19035].

Acknowledgements We thank Marie Schmidt for discovering the similarities between Storylines and Rolling Stock Schedules. We thank Rowan Hoogervorst and Boris Grimm for providing the Rolling Stock Schedules dataset and for their helpful feedback.

1 Introduction

A *storyline* can be seen as a temporal hypergraph; the vertices represent *characters* and the hyperedges, which correspond to given points in time, represent *meetings* (also called *interactions*) among the characters. A *storyline visualization* draws each character as an

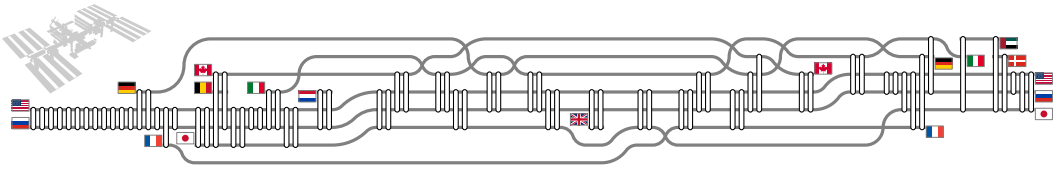


Figure 1 A storyline that visualizes the nationalities of the crew members of expeditions to the International Space Station. Linear wiggle height is minimized using our LP formulation.

x-monotone curve and each meeting as a vertical line segment at the x-coordinate that corresponds to the point in time when the meeting happens; see Figures 1 and 2a. Storyline visualizations have been made for books or movies [8, 13, 20, 22, 26] (where the meetings are the scenes of the story), but they have also been used to illustrate scientific collaboration [15] (where the meetings are joint publications), for genealogical data [17] (where the meetings are marriages and have size 2), or for tweets following certain topics [20].

In order to measure the quality of storyline visualizations, various metrics have been suggested. Most works have focused on reducing the number of crossings of the character curves (simple pairwise crossings [13, 19] or so-called block crossings [27, 28]). Others have also tried to reduce the number of wiggles [11, 20, 26] (that is, the number of turns) and/or the amount of vertical white-space [20, 26].

In this paper, we follow the storyline layout pipeline that has been suggested by Liu, Wu, Wei, Liu, and Liu [20, Fig. 2]. It breaks down the overall problem into four steps; 1. hierarchy generation, 2. ordering (crossing minimization), 3. alignment (wiggle minimization), and 4. compaction (white-space reduction). For step 1, Liu et al. assume that the characters form a given hierarchy that must be respected in step 2. We assume that steps 1 and 2 have been settled, optimally or heuristically. Note that crossing minimization is NP-hard [19, 27].

This paper focuses on wiggle minimization, which appears in step 3 and, in a different form, in step 4. We differentiate between three variants; in each of them, we are given, for every point in time, the vertical ordering of the characters, which for us is fixed. In *wiggle count minimization* (WCMin) the task is to find, for each point in time, y-coordinates for the character curves such that the total number of inflection points (points where the curvature of the character curves changes sign) is minimized. In *linear wiggle height minimization* (LWHMin), the total change in y-coordinate is minimized (see the example in Figure 1). Finally, in *quadratic wiggle height minimization* (QWHMin), the total sum of the squared wiggle heights is minimized.

Related work. In terms of wiggle minimization in storylines, the existing literature mostly proposes heuristic methods, does not formally define a wiggle metric, or considers models that differ from our setting. Ogawa and Ma [23] propose a greedy algorithm for storyline visualization that attempts to simultaneously minimize crossings and wiggle, however, wiggle is not formally defined. Tanahashi and Ma [26] present a genetic algorithm for drawing storylines. They make use of *slots*, which are horizontal strips of the visualization. A genome assigns each meeting to a slot. To evaluate a genome a pipeline approach is used, which rearranges lines inside slots, and then computes a fitness function measuring crossings and wiggle. Liu et al. [20] also present a pipeline approach for drawing storylines, where characters further have geographic locations. Crossings are minimized first by applying a variant of the barycenter heuristic [24] sequentially, then wiggle count is minimized heuristically by reducing the problem to the weighted longest common subsequence problem of two neighboring time steps. Lastly, quadratic wiggle height and whitespace is minimized using a quadratic program. The authors of [25] generalize the earlier approach of Tanahashi and Ma [26] to streaming

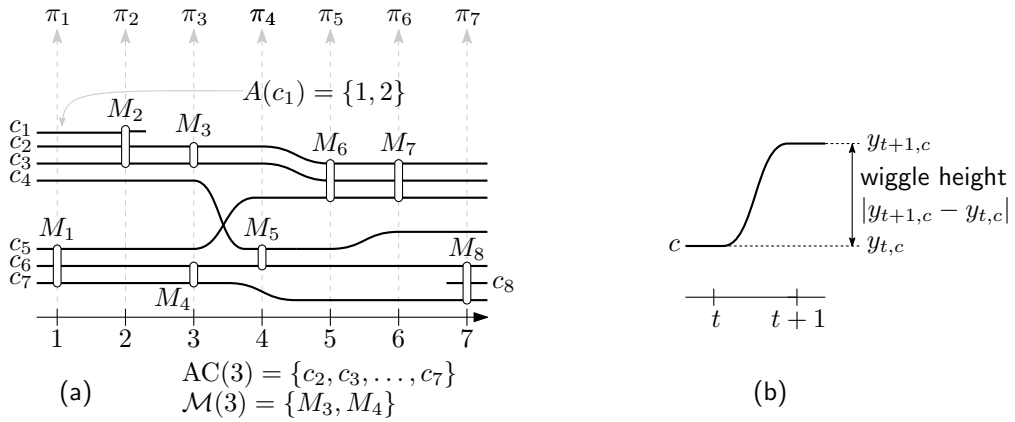


Figure 2 (a) Notation of storylines. (b) Wiggle height of character c between two time steps.

data, where time steps appear one by one. Arendt and Pirrung [1] consider a model where characters can split and merge. They first minimize crossings, and then reduce wiggle count minimization to the independent set problem, which they solve heuristically. Fröschl and Nöllenburg [10, 11] proposed a model for storyline visualization where characters are assigned to discrete cells in a matrix for each time step. They used ILP and SAT solvers to simultaneously minimize a combination of crossings, wiggle count, and linear wiggle height.

Wiggle minimization also plays a role in *streamgraphs* and *stacked area charts* [2, 5, 21], which are visualizations of multiple time series vertically stacked on top of each other without gaps. In these charts, wiggle is minimized combinatorially by computing the best stacking order of the time series.

Our contribution. is as follows.

- We present a linear program (LP) to solve LWHMin efficiently; see Section 3.1.
- We give a quadratic program (QP) to solve QWHMin efficiently; see Section 3.2.
- We prove that WCMin is NP-hard, but can be formulated as an integer linear program (ILP) and admits an efficient solution for two consecutive time steps; see Section 4.
- We present a new method for routing character curves that focuses on keeping distances between neighboring curves constant as long as they run in parallel; see Section 5.
- We report the results of a case study with 17 benchmark instances in which we compare optimal solutions for the three objectives qualitatively and quantitatively; see Section 6.1.
- We present the visualization of rolling stock schedules as a new use case for storylines; see Section 6.2. This can help railway experts to improve or compare such schedules.

Due to space constraints, statements marked with ($\star$) are proved in the full version of this paper [7].

2 Preliminaries

We use $[n]$ as shorthand for $\{1, 2, \ldots, n\}$. A *storyline instance* is a 4-tuple $(\mathcal{C}, T, \mathcal{M}, A)$, where $\mathcal{C} = \{c_1, \ldots, c_n\}$ is a set of *characters*, $T = [\ell]$ is a set of totally ordered *time steps* (or *layers*), and $\mathcal{M} = \{M_1, \ldots, M_m\}$ is a set of *meetings*; see Figure 2a. Each meeting $M \in \mathcal{M}$ has a corresponding time step $\mathrm{time}(M) \in [\ell]$ and consists of a set of characters $\mathrm{ch}(M) \subseteq \mathcal{C}$. Each character $c \in \mathcal{C}$ is *active* for a sequence $A(c) = \{i, i+1, \ldots, j\}$ of consecutive time steps. For each $t \in [\ell]$, we define the active character set $\mathrm{AC}(t) = \{c \in \mathcal{C} \mid t \in A(c)\}$ and the meeting set $\mathcal{M}(t) = \{M \in \mathcal{M} \mid \mathrm{time}(M) = t\}$.

An *ordered storyline instance* is furthermore given, for each $t \in [\ell]$, a permutation π_t of the characters $\mathrm{AC}(t)$. Here, for each meeting M with $\mathrm{time}(M) = t$, the characters $\mathrm{ch}(M)$ appear consecutively in π_t. We write $c \prec_t c'$ if c comes before c' in π_t. A *coordination* of a storyline instance defines for each t and each $c \in \mathrm{AC}(t)$ a y-coordinate $y_{t,c} \in \mathbb{R}$. The coordination is *valid* w.r.t. an ordered storyline instance if for each $t \in [\ell]$ and each pair $c, c' \in \mathrm{AC}(t)$ with $c \prec_t c'$, $y_{t,c} < y_{t,c'}$ holds. Given $\Delta, \overline{\Delta} \in \mathbb{R}_0^+$, a coordination is $(\Delta, \overline{\Delta})$-*nice* (or *nice* if Δ and $\overline{\Delta}$ are clear from the context) if it is valid and if, for every $t \in [\ell]$ and for every pair of characters c, c' that are consecutive in π_t, it holds that

- $|y_{t,c'} - y_{t,c}| = \Delta$ if c and c' are in the same meeting at time step t, and
- $|y_{t,c'} - y_{t,c}| \geq \overline{\Delta}$ otherwise.

A coordination is *integral* if its image contains only integers.

Given a coordination y of an ordered storyline instance, its *total (linear) wiggle height* (see Figure 2b) is defined as

$$\mathrm{LWH}(y) = \sum_{t=1}^{\ell-1} \sum_{c \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1)} |y_{t,c} - y_{t+1,c}|. \tag{1}$$

Its *total quadratic wiggle height* is defined as

$$\mathrm{QWH}(y) = \sum_{t=1}^{\ell-1} \sum_{c \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1)} (y_{t,c} - y_{t+1,c})^2. \tag{2}$$

Finally, its *total wiggle count* is defined as

$$\mathrm{WC}(y) = \left|\{(t,c) \mid 1 \leq t \leq \ell - 1, c \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1), y_{t,c} \neq y_{t+1,c}\}\right|. \tag{3}$$

Note that this matches our intuitive definition of wiggles as inflection points of the character curves.

▶ **Problem 1** (LWHMin)**.** Given an ordered storyline instance and $\Delta, \overline{\Delta} \in \mathbb{R}$, find a nice coordination minimizing the total wiggle height.

▶ **Problem 2** (QWHMin)**.** Given an ordered storyline instance and $\Delta, \overline{\Delta} \in \mathbb{R}$, find a nice coordination minimizing the total quadratic wiggle height.

▶ **Problem 3** (WCMin)**.** Given an ordered storyline instance and $\Delta, \overline{\Delta} \in \mathbb{R}$, find a nice coordination minimizing the total wiggle count.

3 Wiggle Height Minimization

In this section, we describe a linear program (LP) for LWHMin and a quadratic program (QP) for QWHMin. We also present observations about these programs.

3.1 Linear Program for LWHMin

Note that linear programs are polynomial-time solvable [16]. The linear program makes use of the following variables.

- for $t \in [\ell], c \in \mathrm{AC}(t)$, $y_{t,c}$ encodes the y-coordinate of character c at time step t.
- for $t \in [\ell - 1], c \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1)$, $w_{t,c}$ encodes the wiggle height of character c between time steps t and $t+1$.

For our LP formulation we define, for each $t \in [\ell]$, the sets

$$N(t) = \{(c, c') \mid c \text{ and } c' \text{ are characters that are consecutive in } \pi_t \text{ and } c \prec_t c'\},$$
$$N_{\mathcal{M}}(t) = N(t) \cap \{(c, c') \mid c \text{ and } c' \text{ are in the same meeting at time step } t\}, \text{ and}$$
$$N_A(t) = N(t) \setminus N_{\mathcal{M}}(t)$$

The LP is given as follows.

$$\text{Minimize} \sum_{t \in [\ell], c \in \mathrm{AC}(t)} w_{t,c}$$

$$\begin{aligned}
\text{subject to} \quad y_{t,c'} - y_{t,c} &= \Delta && \text{for all } t \in [\ell], (c, c') \in N_{\mathcal{M}}(t) && \text{(Cons-}\Delta\text{)}\\
y_{t,c'} - y_{t,c} &\geq \overline{\Delta} && \text{for all } t \in [\ell], (c, c') \in N_A(t) && \text{(Cons-}\overline{\Delta}\text{)}\\
y_{t,c} - y_{t+1,c} &\leq w_{t,c} && \text{for all } t \in [\ell - 1], c \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1) && \text{(W1)}\\
y_{t+1,c} - y_{t,c} &\leq w_{t,c} && \text{for all } t \in [\ell - 1], c \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1) && \text{(W2)}\\
y_{t,c} &\geq 0 && \text{for all } t \in [\ell], c \in \mathrm{AC}(t) && \text{(G0)}
\end{aligned}$$

Constraints (Cons-$\overline{\Delta}$) and (Cons-Δ) ensure that the computed coordination is $(\Delta, \overline{\Delta})$-nice. Together with the objective, (W1) and (W2) ensure that the wiggle value is computed correctly as $w_{t,c} = |y_{t,c} - y_{t+1,c}|$. The objective adds up the wiggle values. Our LP has the following nice property.

▶ **Proposition 1.** *If $\Delta, \overline{\Delta} \in \mathbb{N}$, then all extreme points defined by the wiggle height minimization polytope are integer.*

Proof. Assume assignments to y- and w-variables corresponding to an extreme point. It is clear that $\min\{y_{t,c} \mid t \in [\ell], c \in \mathrm{AC}(t)\} = 0$, as otherwise there is some small $\epsilon > 0$ such that each y-variable can be simultaneously increased or decreased by ϵ, contradicting that we are at an extreme point. It is also clear that, for each $t \in [\ell-1]$ and $c \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1)$, either (W1) or (W2) is binding. Now, let G be the graph defined by $V(G) = \{y_{t,c} \mid t \in [\ell], c \in \mathrm{AC}(t)\}$ and $E(G) = \{y_{t,c}y_{t',c'} \mid y_{t,c} - y_{t',c'} \in \mathbb{Z}\}$. If G is connected, we are done as we have already seen that the smallest y-variable is integer. Otherwise, consider some connected component C of G such that $V(C) \not\subseteq \mathbb{Z}$. Note that it is possible to add to all $y_{t,c} \in V(C)$ some small $\epsilon > 0$, resulting in a different point enclosed in the polytope. Equivalently, there exists $\epsilon' > 0$ that can be subtracted from all $y_{t,c} \in V(C)$, witnessing in fact that we are not at an extreme point. Thus, G is connected and the extreme point is integer. ◀

3.2 Quadratic Program for QWHMin

The quadratic program makes use of the same variables and constraints as the LP above. In fact, we just replace the objective function by the definition of quadratic wiggle height

$$\text{Minimize} \sum_{t \in [\ell], c \in \mathrm{AC}(t)} w_{t,c}^2.$$

It is easy to see that, in matrix form, the objective has diagonal entries that are either 0 or $w_{t,c}$ (for some $t \in [\ell - 1]$ and $c \in \mathrm{AC}(t)$). The entries of type $w_{t,c}$ are non-negative by definition, hence the matrix of the objective is positive semidefinite. QPs with such an objective and with linear constraints can be solved in polynomial time [4].

Note that we can replace each of the summands $w_{t,c}^2$ in the objective by $(y_{t,c} - y_{t+1,c})^2$, which is the same as $(y_{t+1,c} - y_{t,c})^2$ (for all $t \in [\ell - 1], c \in \mathrm{AC}(t) \cap \mathrm{AC}(t + 1)$). Then we can drop the w-variables and the constraints (W1) and (W2).

4 Wiggle Count Minimization

In this section we consider the number of wiggles in the visualization as optimization criterion. We show that for $\ell = 2$ time steps, WCMin is polynomial-time solvable, while it is NP-hard in the general case. We also present an integer linear program to solve WCMin.

4.1 Polynomial Cases

When ignoring the restriction that characters in meetings need to be equally spaced, finding a storyline visualization with the fewest number of wiggles is polynomial-time solvable. The problem is equivalent to finding the longest common subsequence between each pair of adjacent orderings π_t, π_{t+1}, $t \in [\ell - 1]$. Integer coordinates can be obtained by scaling all y-coordinates accordingly. A similar observation has been made in [20].

▶ **Observation 2.** *Given an ordered storyline instance, one can compute in polynomial time a valid integral coordination that minimizes the total wiggle count.*

Rather surprisingly, we obtain the following result. It shows that we can compute the "maximum wiggle-free" storyline – a set of characters that all can be realized without a single wiggle – in polynomial time. The proof is in the full version [7].

▶ **Theorem 3** ($\star$). *Let $(\mathcal{C}, T, \mathcal{M}, A, (\pi_t)_{t\in[\ell]})$ be an ordered storyline instance. Let $\mathcal{C}'$ be the set of characters in $\mathcal{C}$ that are active at all time steps, i.e., $A(c) = [\ell]$ for all $c \in \mathcal{C}'$. Then, given a pair $(\Delta, \overline{\Delta})$, a $(\Delta, \overline{\Delta})$-nice coordination with the largest wiggle-free subset of $\mathcal{C}'$ can be computed in $\mathcal{O}(|\mathcal{C}|^2 \cdot \ell)$ time.*

The next result follows directly, as for $\ell = 2$, the only characters that can have (at most one) wiggle, are active at both time steps.

▶ **Corollary 4.** WCMin *is solvable in time $\mathcal{O}(|\mathcal{C}|^2)$ for $\ell = 2$.*

4.2 NP-Hardness

In this section, we prove the following result with a gadget-based reduction. We first describe the source problem of the reduction, then the required gadgets, and finally the full reduction.

▶ **Theorem 5.** *The decision variant of* WCMin *is* NP*-complete.*

Our proof reduces from the problem Planar Monotone 3-Sat, which is defined as follows. Let ϕ be a Boolean 3-Sat formula in conjunctive normal form, let Υ be the set of variables of ϕ, and let Γ be the set of clauses of ϕ. The formula ϕ is *monotone* if every clause of ϕ

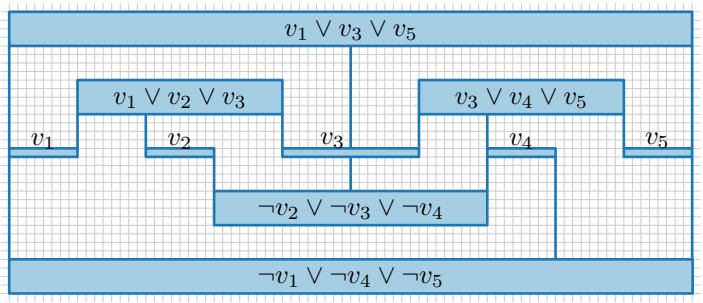


Figure 3 An instance of PM3-Sat.

contains either only positive or only negative literals. We define the bipartite variable-clause graph $G(\phi)$, consisting of vertex set $\Gamma \cup \Upsilon$ and an edge between a clause $\gamma \in \Gamma$ and a variable $v \in \Upsilon$ if and only if v occurs in c (positively or negatively). A *monotone planar rectilinear embedding* $\mathcal{E}$ of $G(\phi)$ is a planar embedding of $G(\phi)$ on an integer grid (see Figure 3) such that

- all vertices in $G(\phi)$ are represented by disjoint integer coordinate rectangles,
- the y-coordinates of all variable vertices are the same,
- all positive clause vertices are placed above the variable vertices,
- all negative clause vertices are placed below the variable vertices, and
- edges are straight vertical segments with integer x-coordinates connecting the borders of their corresponding rectangles.

▶ **Problem 4** (Planar Monotone 3-Sat (PM3-Sat))**.** Given a monotone 3-Sat formula ϕ and a monotone planar rectilinear embedding $\mathcal{E}$ of $G(\phi)$, decide whether ϕ is satisfiable.

PM3-Sat is NP-complete [6]. We can assume w.l.o.g. that the coordinates in the embedding $\mathcal{E}$ have the following special properties:

- The x-coordinates of edges and vertical borders of the vertex rectangles are multiples of 8.
- All edge lengths are multiples of 4.
- The clause vertices have height 4 and the variable vertices have height 1.

Notice that Figure 3 shows an instance satisfying these properties.

Now, given such an instance $I_{\text{SAT}} = (\phi, \mathcal{E})$ of PM3-Sat, we construct an instance $I_{\text{SL}} = (\mathcal{C}, T, \mathcal{M}, A, (\pi_t)_{t\in[\ell]}, \Delta, \overline{\Delta})$ of WCMin. We will give the precise number of wiggles W for the decision variant of WCMin later, and set $\Delta = \overline{\Delta} = 1$, thus, we can assume that coordinations of I_{SL} contain only integer coordinates (see Lemma 11).

To have a better correspondence between the two instances, we assume in the reduction that the storyline curves are not x-monotone, but are y-monotone from bottom to top. That is, we rotate the storyline drawing by 90 degrees counterclockwise. All figures for the reduction reflect this. We assume w.l.o.g. that the smallest y-coordinate of any vertex in $\mathcal{E}$ is 3. Let $y_{\max}$ be the largest y-coordinate. We set $T = \{1, \dots, y_{\max} + 2\}$. The main part of our reduction will be in $[3, y_{\max}] \subseteq T$, while the remaining four time steps will be used for an auxiliary bounding *frame*.

Our reduction consists of three components: variable gadgets, clause gadgets, and a rigid frame. For presentation purposes, we define the storyline instance graphically. That is, we define the instance in terms of one of its coordinations. The definitions of $\mathcal{C}, \mathcal{M}, A$, and $(\pi_t)_{t\in[\ell]}$ can then be inferred from this coordination. In the main part of the paper, we assume a black-box bounding frame, which we assume to have fixed coordinates in any coordination. Clearly, this frame must consist of meetings and characters, its realization is given in in the full version [7] due to space constraints. An instance of WCMin that results from the instance in Figure 3 can already be found in Figure 6 in the variant with a black-box bounding frame. We start with the description of the variable gadget.

Variable gadget. Assume all variables in $\mathcal{E}$ are embedded on the two y-coordinates y_{top} and y_{bot}. Now consider a variable $v \in \Upsilon$. We assume that the corresponding rectangle in $\mathcal{E}$ has width w_v, and its horizontal edges span from x_{left} to x_{right}. Then the corresponding variable gadget (see Figure 4) consists of a set C_v of w_v characters which are ordered as in Figure 4 and are active at time steps y_{bot} and y_{top}. This set of characters can be further partitioned into a set C_v^{pos} of *positive characters* (green in Figure 4) and a set C_v^{neg} of *negative characters* (red in Figure 4). Furthermore, using a frame as in the figure, the variable gadget is confined to a space of width $w_v + 2$, with coordinates from $x_{\text{left}} - 1$ to $x_{\text{right}} + 1$.

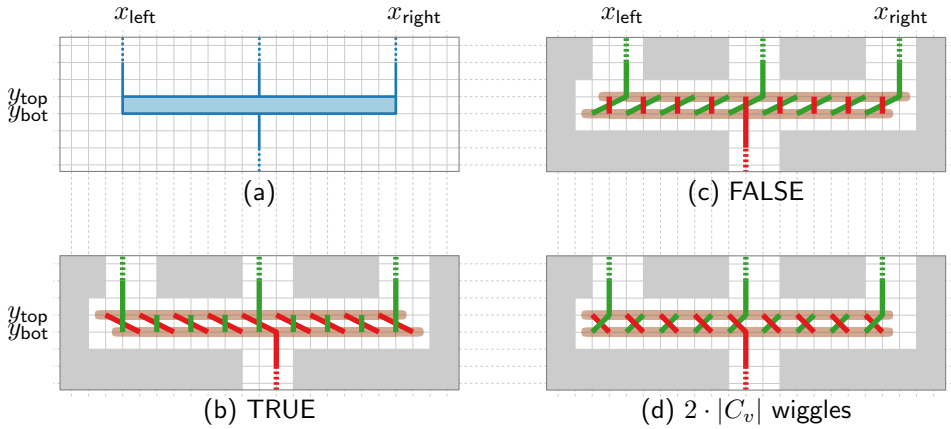


Figure 4 Illustration of a variable gadget. (a) The variable in the embedding $\mathcal{E}$. (b) The corresponding variable gadget in the TRUE state. (c) The variable gadget in the FALSE state. (d) The variable gadget in a state with too many wiggles. Character curves are green and red. meetings are brown horizontal blocks. The rigid frame is shown in gray. The dotted lines sketch the connections to the corresponding clause gadgets.

For each occurrence of a variable in a clause, a character is extended towards a clause gadget, as indicated with the dashed line ends in Figure 4. Positive characters are connected with positive clauses, negative characters with negative clauses. More precisely, for a vertical edge of $\mathcal{E}$ connecting v with a positive (negative) clause $\gamma \in \Gamma$ at x-coordinate $x_{\text{left}} + x_{v,\gamma}$ with $x_{v,\gamma} \in \mathbb{N}_0$, let $c_{v,\gamma} \in C_v^{\text{pos}}$ ($c_{v,\gamma} \in C_v^{\text{neg}}$) be the $(1+x_{v,\gamma}/2)$th (from left to right) positive (negative) character in C_v^{pos} (C_v^{neg}) (the values are always positive as x-coordinates of edges are multiples of eight). We call such a character *wire character*. This character is extended to time steps $y_{\text{top}}+1, y_{\text{top}}+2, \dots$ ($y_{\text{bot}}-1, y_{\text{bot}}-2, \dots$) until it reaches the corresponding clause gadget. This connection is along a corridor of width two defined by x-coordinates $x_{\text{left}} + x_{v,\gamma}$ and $x_{\text{left}} + x_{v,\gamma} + 1$ (see Figure 4). In a potential coordination, we say that $c_{v,\gamma}$ is *satisfying* if the x-coordinate of $c_{v,\gamma}$ at y_{top} (y_{bot}) is $x_{\text{left}} + x_{v,\gamma}$, and not satisfying otherwise. We say that $x_{\text{left}} + x_{v,\gamma}$ is the *satisfying x-coordinate* of $c_{v,\gamma}$.

Due to the fixed coordinates of frames, we observe the following.

▶ **Observation 6.** *The variable gadget has either $|C_v|$ (Figure 4b,c) or $2|C_v|$ (Figure 4d) wiggles between time steps y_{top} and y_{bot}.*

Furthermore, as desired, variable gadgets model a choice between TRUE and FALSE (see Figure 4b,c).

▶ **Observation 7.** *If the variable gadget has $|C_v|$ wiggles, then either all positive wire characters are satisfying and all negative wire characters are not satisfying, or vice versa.*

Clause gadget. At the heart of the reduction is the clause gadget, see Figure 5. We desribe the gadget for positive clauses only. The gadgets for negative clauses are the same but mirrored along the x-axis and can also be found in Figure 6. Thus, let γ be a positive clause with three variables $v_1, v_2, v_3 \in \Upsilon$, such that v_1 is the left, v_2 is the middle, and v_3 is the right variable. Assume that the clause rectangle in $\mathcal{E}$ is defined by the coordinates $[x_{\text{left}}, x_{\text{right}}] \times [y_{\text{bot}}, y_{\text{top}}]$. The clause gadget for γ is essentially placed into the rectangle

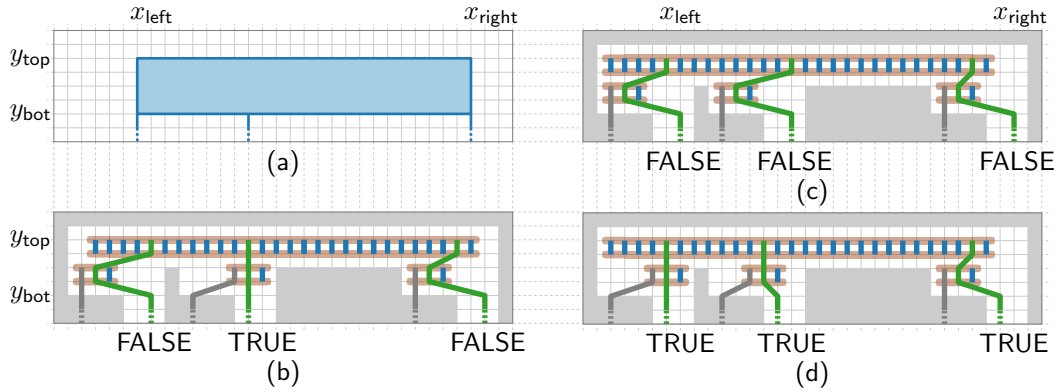


Figure 5 Illustration of a positive clause gadget. (a) The clause in the embedding $\mathcal{E}$. (b-d) The same clause gadget in three different states depending on whether its wire characters are satisfying (TRUE) or not (FALSE). A character belonging to the rigid frame is shown with the gray line.

$[x_{\text{left}} - 4, x_{\text{right}} + 1] \times [y_{\text{bot}}, y_{\text{top}}]$.[1] The rest of the clause gadget depends on the satisfying x-coordinates of the characters $c_{v_i,\gamma}$ with $i \in [3]$; see Figure 5. Thus, let x_1^s, x_2^s, x_3^s be the satisfying x-coordinates of $c_{v_1,\gamma}, c_{v_2,\gamma}, c_{v_3,\gamma}$, respectively. Each of the three wire characters is active until time step y_{top}, and it is part of some meetings with *clause gadget characters* (blue in Figure 5). For each wire character $c_{v_i,\gamma}$, we have the following construction.

- A *fixing character* which is a character from the frame fixed at $x_i^s - 4$ which is extended until $y_{\text{bot}} + 2$, see the gray line in Figure 5.
- The two *blocking meetings* of size three at time steps $y_{\text{bot}} + 1$ and $y_{\text{bot}} + 2$ that ensure that $c_{v_i,\gamma}$ must have at least one wiggle if its x-coordinate at y_{bot} is not satisfying.

Lastly, the clause gadget has two *choice meetings* of size $x_{\text{right}} - x_{\text{left}} + 4$ at y_{top} and $y_{\text{top}} - 1$. Each clause character $c_{v_i,\gamma}$ with $i \in [3]$ appears inside these meetings at position $x_i^s - x_{\text{left}} + i$. The remaining positions are filled by clause characters which have the same position in both meetings. Essentially, the addition of i, which slightly shifts the relation of satisfying x-coordinate and position in the meeting, ensures that we cannot prevent multiple wiggles by having multiple satisfying wire characters. The following two lemmas show that the clause gadget can be realized with the fewest wiggles if at least one of its characters is satisfying, and otherwise with one wiggle more. We assume that wire characters do not have wiggles between their variable gadget and y_{bot}, as they are either unnecessary or can be moved into the clause gadget.

▶ **Lemma 8.** *Assume that there exists a nice coordination in which none of $c_{v_1,\gamma}$, $c_{v_2,\gamma}$, and $c_{v_3,\gamma}$ is satisfying. Then a nice coordination of the clause gadget has at least six wiggles, and such a coordination with exactly six wiggles always exists.*

Proof. Figure 5 depicts such a nice coordination. We show that the number of wiggles of each wire character $c_{v_i,\gamma}$ with $i \in [3]$ and its corresponding fixing character is at least two between y_{bot} and y_{top}. If the number of wiggles is at least two between y_{bot} and $y_{\text{bot}} + 1$, we are done.

Otherwise, $c_{v_i,\gamma}$ has x-coordinate $x_i^s - 3$ at time step $y_{\text{top}} - 3$. Now notice that there is no nice coordination of the choice and blocking meetings such that $c_{v_i,\gamma}$ can be drawn without a wiggle between $y_{\text{top}} - 3$ and $y_{\text{top}} - 1$. ◀

[1] That is also partially why we need x-coordinates being multiples of eight.

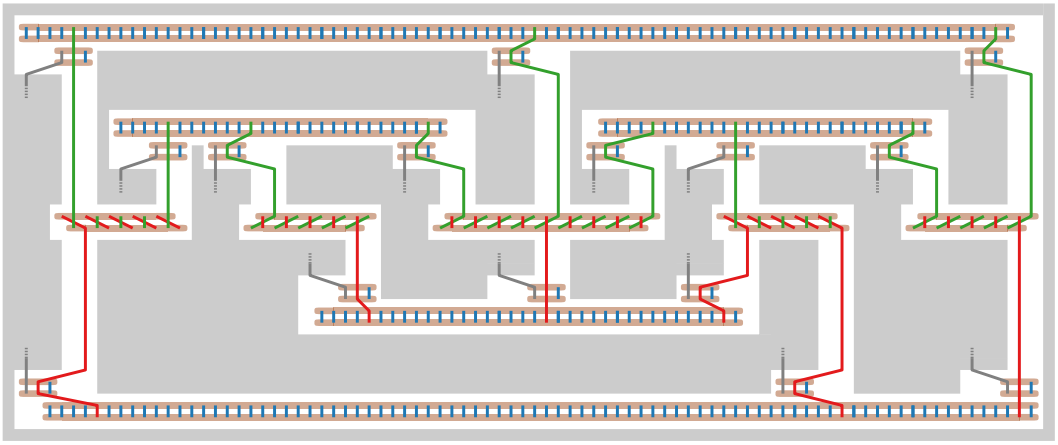


Figure 6 The complete reduction (with black-box frame) resulting from the instance in Figure 3.

▶ **Lemma 9.** *Assume a nice coordination in which some of $c_{v_i,\gamma}$ with $i \in [3]$ is satisfying. Then a nice coordination of the clause gadget has at least five wiggles, and such a coordination with exactly five wiggles always exists.*

Proof. For the existence, let $c_{v_j,\gamma}$ with $j \in [3]$ be some satisfying wire character. Simply draw $c_{v_j,\gamma}$ without wiggle, by adjusting the coordination of the choice meetings accordingly. The remaining two wire characters can be drawn with two wiggles, while their fixing character remains without wiggle between y_{bot} and y_{top} (see Figure 5a, where $j = 2$).

For the lower bound we have already seen in the proof of Lemma 8 that if a fixing character remains without wiggle between y_{bot} and y_{top}, then its corresponding wire character must have at least one wiggle between $y_{\text{top}} - 2$ and $y_{\text{top}} - 1$. Thus, for there to be at most 4 wiggles, two wire characters would need to be realized without wiggles. This is not possible because of the slight shift of positions of the wire characters in the choice meetings, a contradiction. ◀

We define k_{cr} as the number of crossings in the reduced storyline instance; in any coordination they will introduce a wiggle. Now, let $k = k_{\text{cr}} + 5|\Gamma|$. We claim the following; a proof without treating the rigid frame as black-box is given in the full version [7].

▶ **Lemma 10** (⋆). *I_{SAT} is a yes-instance of* Planar Monotone 3-Sat *if and only if I_{SL} has a nice coordination with at most k wiggles.*

Proof (assuming a black-box bounding frame). The forward direction follows immediately from our construction.

For the backward direction, assume a nice coordination of I_{SL} with at most k wiggles. Because k_{cr} wiggles are unavoidable, and we have at least 5 wiggles for each clause gadget by Lemmas 8 and 9, we must have exactly 5 wiggles per clause gadget. Further, each variable gadget has $|C_v|$ wiggles for each variable v. Hence, at least one wire character per clause gadget is satisfying, and we can use the variable assignment corresponding to satisfying characters to verify the satisfiability of the PM3-Sat instance by Observation 7. ◀

Proof of Theorem 5. For NP-membership we provide a polynomial certificate and a certifier for the decision problem of deciding whether there is a nice coordination with at most k wiggles. The certificate consists of a set of pairs $X \subseteq \mathcal{C} \times [\ell - 1]$ of size at least $\sum_{c \in \mathcal{C}}(|A(c)| - 1) - k$. Each pair $(c, t) \in X$ defines that c can be drawn without wiggle between time steps t and $t + 1$. The certifier can certify the certificate using a linear program. The linear program

is essentially the one from Section 3 using constraints (Cons-Δ) to (G0) and the additional constraints $y_{c,t} = y_{c,t+1}$ for each pair $(c,t) \in X$. If the polytope defined by the program is non-empty, then the certifier reports yes.

NP-hardness follows from Lemma 10 and the fact that PM3-Sat is NP-hard. Clearly, the reduction is polynomial. ◀

4.3 ILP Formulation for WCMin

We have established that WCMin is NP-complete. We now present an ILP formulation for WCMin (assuming $\Delta, \overline{\Delta} \in \mathbb{N}$) that we will use in Section 6 to construct storyline drawings with minimum wiggle count. The formulation requires the following lemma.

▶ **Lemma 11.** *Consider an ordered storylines instance and $\Delta, \overline{\Delta} \in \mathbb{N}$. For each $k \in \mathbb{N}$, if there is a nice coordination with k wiggles, there is one with at most k wiggles using integer coordinates.*

Proof. The proof is similar to the one of Proposition 1. We recommend reading that proof first. Consider a nice coordination with k wiggles. We build a graph G with $V(G) = \{y_{t,c} \mid t \in [\ell], c \in \mathrm{AC}(t)\}$ and $E(G) = \{y_{t,c}y_{t',c'} \mid y_{t,c} - y_{t',c'} \in \mathbb{Z}\}$. If G has multiple connected components, the y-coordinates corresponding to vertices in a single connected component can be increased until the number of connected components of G is decreased. When G is connected, all y-coordinates can be increased until they are all integers. ◀

The ILP makes use of the following variables (refer to Section 3 for details).

- $y_{t,c}$ for $t \in [\ell], c \in \mathrm{AC}(t)$, y-coordinate of character c at time step t.
- $z_{t,c}$ for $t \in [\ell-1], c \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1)$, equals 1 if character c wiggles between t and $t+1$, 0 otherwise.

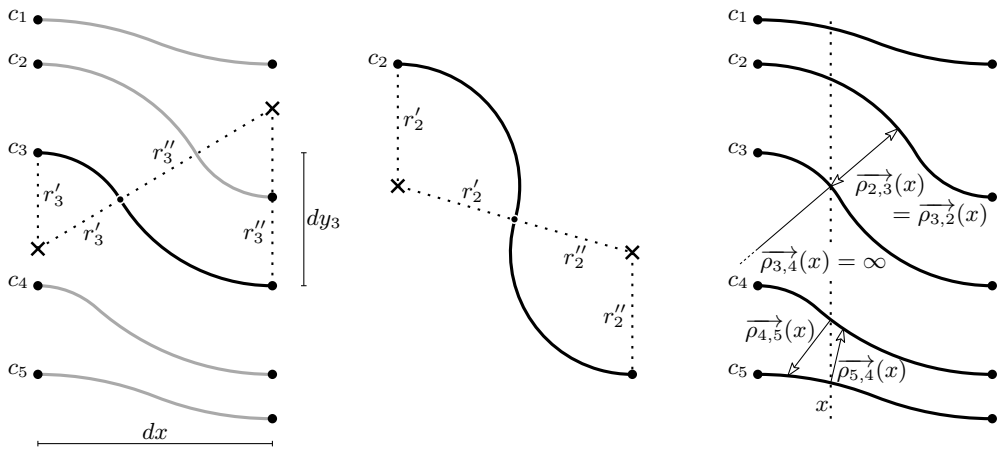
By Lemma 11 we can assume the y-coordinates of the characters to be integers. To enforce that $z_{t,c} = 0$ if and only if $y_{t,c} = y_{t+1,c}$ (i.e., there is no wiggle), we use a large constant $Y \in \mathbb{N}$; see constraints (M1) and (M2) below. A trivial upper bound for the y-coordinates is $\max\{\Delta, \overline{\Delta}\} \cdot \sum_{t \in [\ell]} |\mathrm{AC}(t)|$. We use this bound for Y. The sets $N_{\mathcal{M}}(t)$ and $N_A(t)$ are defined in Section 3. Now we can formulate the ILP:

$$\text{Minimize} \quad \sum_{t \in [\ell-1], c \in \mathrm{AC}(t)} z_{t,c}$$

$$\begin{aligned}
\text{subject to} \quad y_{t,c'} - y_{t,c} &= \Delta && \text{for all } t \in [\ell], (c,c') \in N_{\mathcal{M}}(t) && \text{(Cons-}\Delta\text{)}\\
y_{t,c'} - y_{t,c} &\geq \overline{\Delta} && \text{for all } t \in [\ell], (c,c') \in N_A(t) && \text{(Cons-}\overline{\Delta}\text{)}\\
y_{t,c} + Y \cdot z_{t,c} &\geq y_{t+1,c} && \text{for all } t \in [\ell-1], c \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1) && \text{(M1)}\\
y_{t,c} - Y \cdot z_{t,c} &\leq y_{t+1,c} && \text{for all } t \in [\ell-1], c \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1) && \text{(M2)}\\
y_{t,c} \in \mathbb{N}_0, \quad 0 \leq y_{t,c} &< Y && \text{for all } t \in [\ell], c \in \mathrm{AC}(t) && \text{(Int)}\\
z_{t,c} &\in \{0,1\} && \text{for all } t \in [\ell], c \in \mathrm{AC}(t) && \text{(Bin)}
\end{aligned}$$

Constraints (Cons-Δ) and (Cons-$\overline{\Delta}$) have been explained in Section 3. (Int) and (Bin) ensure that $y_{t,c}$ is a positive integer bounded by Y and $z_{t,c}$ is binary, respectively. The objective sums over all z-variables and therefore counts the number of wiggles.

As it turned out, optimal solutions created by state-of-the-art solvers produce unusably large wiggles with this formulation. We thus introduce a new variable h for the height of the drawing and, for each $t \in [\ell]$ and $c \in \mathrm{AC}(t)$, the new constraint $y_{t,c} \leq h$. By adding the term h/Y to the objective, we also minimize h. The adjusted formulation still solves WCMin.



(a) Wiggle geometry. **(b)** Not x-monotone. **(c)** Radial curve distance.

Figure 7 A wiggle is represented by a sequence of two circular arcs.

5 Routing Character Curves

In our storyline drawings, we represent each wiggle by a sequence of two circular arcs. In order to generate a smooth transition, we require the tangents at the arcs to be equal where the arcs cross over (see Figure 7a). In order to achieve visually pleasing transitions, we require that all arcs have at least a certain radius and that consecutive lines wiggle " in parallel" (a precise definition is given later in this section).

When visualizing the transition between a pair of consecutive time steps t and $t+1$, for each character c, the wiggle height $dy_{t,c} = |y_{t,c} - y_{t+1,c}|$ is given by the coordination. Given the radii $r'_{t,c}$ and $r''_{t,c}$ of the two arcs of a character curve c between time steps t and $t+1$, the required horizontal space dx_t for this configuration fulfills

$$dx_t^2 = 2(r'_{t,c} + r''_{t,c})dy_{t,c} - dy_{t,c}^2. \tag{4}$$

Refer to the full version [7] for a detailed derivation. We require each character curve to be x-monotone. If we choose dx_t badly, we may end up with a sequence of arcs that is not x-monotone (see Figure 7b). It is easy to see that a character curve is x-monotone if and only if $(r'_{t,c} + r''_{t,c}) \geq dy_{t,c}$. With Equation (4), we can state this as $dx_t^2 \geq dy_{t,c}^2$.

Between two time steps, two character curves run in parallel, if they do not cross and start and end with the same vertical distance. In order to have them wiggle " in parallel" , we use concentric arcs. Generally, for two x-monotone differentiable curves φ and φ', we define their *directed radial distance* at position x, denoted by $\overrightarrow{\rho_{\varphi,\varphi'}}(x)$, as the length of a normal line segment starting at $\varphi(x)$ and ending where it intersects φ' or infinity, if the normal line does not intersect φ'. See Figure 7c for an example. We define the *radial distance* of φ and φ' at x as $\rho_{\varphi,\varphi'}(x) = \min(\overrightarrow{\rho_{\varphi,\varphi'}}(x), \overrightarrow{\rho_{\varphi',\varphi}}(x))$ (see Figure 7c).

We call two x-monotone curves *related* if their axis-aligned bounding boxes overlap. We call them *upwards (downwards) co-oriented* if they are also y-monotone and their y-coordinates both increase (decrease). Let φ and φ' be two character curves between consecutive time steps t and $t+1$. By construction, φ and φ' are y-monotone. If they are also related, co-oriented, and do not cross each other, we require their radial distance to be monotone.

We define the set $N_{\uparrow,\leftarrow}(t)$ of all pairs of characters $c, \hat{c} \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1)$ such that

- $y_{t,c} < y_{t,\hat{c}}$,
- c and $\hat{c}$ are non-crossing, related, and upwards co-oriented, and
- $y_{t,\hat{c}} - y_{t,c} \leq y_{t+1,\hat{c}} - y_{t+1,c}$.

We define $N_{\downarrow,\leftarrow}(t)$ for downwards co-oriented characters similarly as well as $N_{\uparrow,\rightarrow}(t)$ and $N_{\downarrow,\rightarrow}(t)$ if $y_{t,\hat{c}} - y_{t,c} > y_{t+1,\hat{c}} - y_{t+1,c}$. Note that these sets contain only characters that wiggle (i.e. their y-coordinates are different in t and $t+1$). We use the following LP to determine dx_t^2 as well as the radii $r'_{t,c}$ and $r''_{t,c}$ for each time step $t \in [\ell - 1]$ and for each $c \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1)$.

$$\begin{aligned}
\text{Minimize} \quad & dx_t^2 \\
\text{subject to} \quad & dx_t^2 \geq \max_{c \in \mathrm{AC}(t)} dy_{t,c}^2 && && \text{(XM)} \\
& dx_t^2 = 2(r'_{t,c} + r''_{t,c}) dy_{t,c} - dy_{t,c}^2 && \text{for all } c \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1) && \text{(R1)} \\
& r'_{t,c}, r''_{t,c} \geq r_{\min} && \text{for all } c \in \mathrm{AC}(t) \cap \mathrm{AC}(t+1) && \text{(R2)} \\
& r'_{t,c} \geq r'_{t,\hat{c}} + (y_{t,c} - y_{t,\hat{c}}) && \text{for all } c, \hat{c} \in N_{\downarrow,\leftarrow}(t) && \text{(D1)} \\
& r''_{t,c} \leq r''_{t,\hat{c}} - (y_{t+1,c} - y_{t+1,\hat{c}}) && \text{for all } c, \hat{c} \in N_{\downarrow,\rightarrow}(t) && \text{(D2)} \\
& r'_{t,c} \leq r'_{t,\hat{c}} - (y_{t,c} - y_{t,\hat{c}}) && \text{for all } c, \hat{c} \in N_{\uparrow,\leftarrow}(t) && \text{(D3)} \\
& r''_{t,c} \geq r''_{t,\hat{c}} + (y_{t+1,c} - y_{t+1,\hat{c}}) && \text{for all } c, \hat{c} \in N_{\uparrow,\rightarrow}(t) && \text{(D4)}
\end{aligned}$$

The constraint (XM) ensures x-monotonicity between time steps t and $t+1$; (R1) ensures that the two arcs meet at a distinct point and their tangents match. The remaining quality criteria are addressed by (R2) (minimum radius) and (D1) to (D4) (monotone radial distance). Note that we can model each variable of kind dx_t^2 as a linear variable because we do not have any terms depend on dx_t (non-squared).

6 Case Studies

We conducted two case studies that showcase wiggle minimization and the geometric realization we proposed. Firstly, we compare the different metrics for storyline wiggle (see Section 2) using both previous datasets from the literature and new instances. Secondly, we present a new use case for storylines when visualizing rolling stock schedules.

6.1 Storyline Benchmark Data

We compiled a diverse set of benchmark instances for which we could create crossing minimal ordered storylines using the ILP from [8]. It consists of three novels (n_1 to n_3) originally from The Stanford GraphBase [18] in a modified version by [13], eight blockbuster movies (b_1 to b_5 from [8] and b_6 to b_8 from [25]), one instance of publication data (p_1) from [9], and five rolling stock schedules (t_1 to t_5; see Section 6.2). Refer to a table in the full version [7] for more details about these instances, including objective values and running times.

We implemented the LP solving LWHMin (see Section 3.1), the QP solving QWHMin (see Section 3.2), and the ILP for WCMin (see Section 4.3) including the additional secondary objective that minimizes drawing height. We refer to them as LP, QP, and ILP, respectively. We implemented base – a heuristic that simply centers the character curves vertically in each time step – as a baseline. Note that base produces drawings of minimum height.

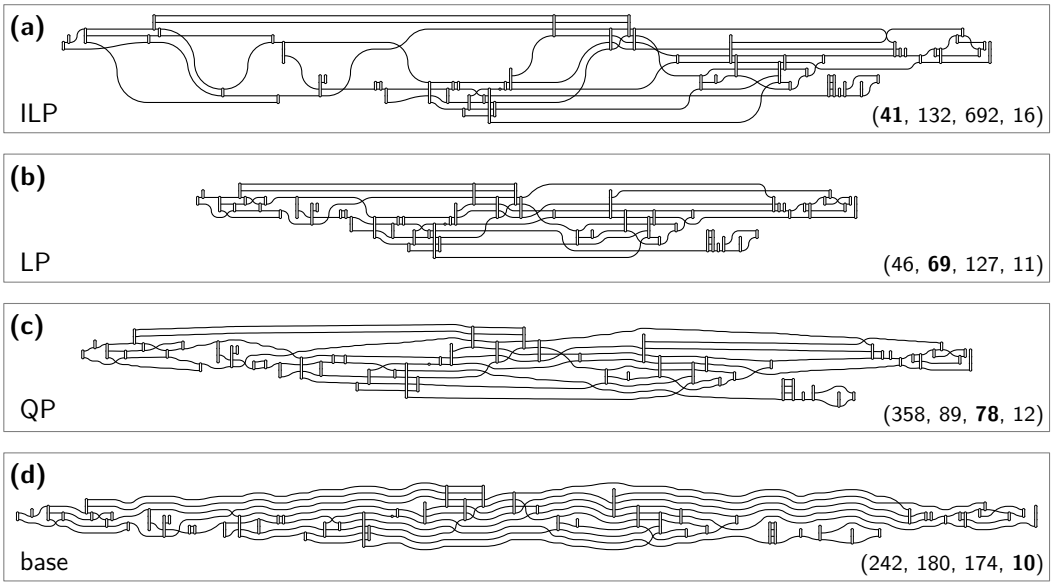


Figure 8 Storyline visualizations of the first chapter of Anna Karenina. The values in parentheses are (wiggle count, linear wiggle height, quadratic wiggle height, height of the bounding box).

Drawings of n_1 created with each of the four methods can be found in Figure 8. We see that minimizing the number of wiggles comes at a cost of some large wiggles and that producing many – thus small – wiggles results in an unnecessarily wide drawing when using our style. There could be a case for QP with an adjusted drawing style but for our current rendering, LP produces the best results.

Detailed results of the performance metrics for the dataset can be found in Figure 9. We see that optimizing for one objective can lead to poor performance in other metrics. Overall, however, LP produces good results more consistently than the other methods. This is why we used it for the drawings presented in our second case study.

6.2 Rolling Stock Scheduling

To our own surprise, storylines show up in a seemingly unrelated area, namely the visualization of rolling stock schedules. Rolling stock scheduling, which is an important subtask in railway optimization, is defined as follows. Given a set of train units, a set of possible compositions of train units, a set of trips given by the timetable, and passenger demands per trip, the task is to find a feasible assignment of train units (possibly in composition) to trips that minimizes a weighted sum of operating costs. These costs include, e.g., the total distance driven, the load factor, and the number of coupling operations. There are different approaches for visualizing rolling stock schedules (see [3] and [12, Fig. 1] for examples).

We visualize such a schedule as a storyline where each character curve represents the movement of a specific train unit over the planning period (usually one day of operation) and the composition of (typically two or three) train units is modeled as a meeting. In contrast to our basic model, such meetings now extend over several layers. Note that we use the term layer to avoid confusion with time steps in the timetable. In order to satisfy the additional constraints of "rolling stock" storylines, we needed to extend our basic model. We split each trip into at least two meetings, one at the departure time and one at the arrival time. We then group as many consecutive meetings (of pairwise different train units) into one layer as possible so the storyline stays compact. If departure and arrival meeting of one trip do not fall into consecutive layers, we insert an additional meeting for each intermediate layer.

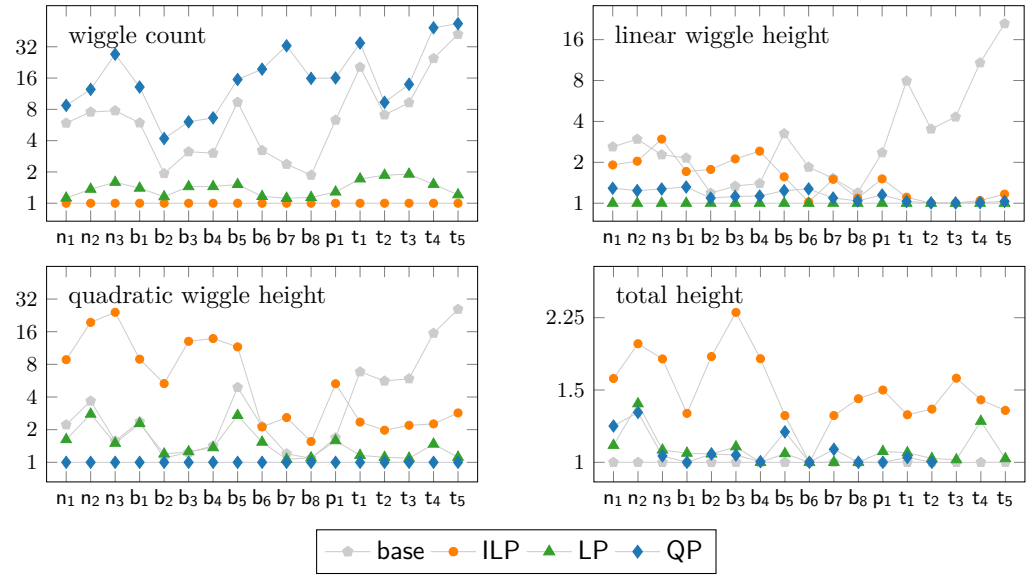


Figure 9 Metrics (relative to the respective optimum) of the wiggle minimization methods for our storyline benchmark set (**n**ovels, **b**lockbusters, **p**ublications, **t**rains). The y-axes are logarithmic.

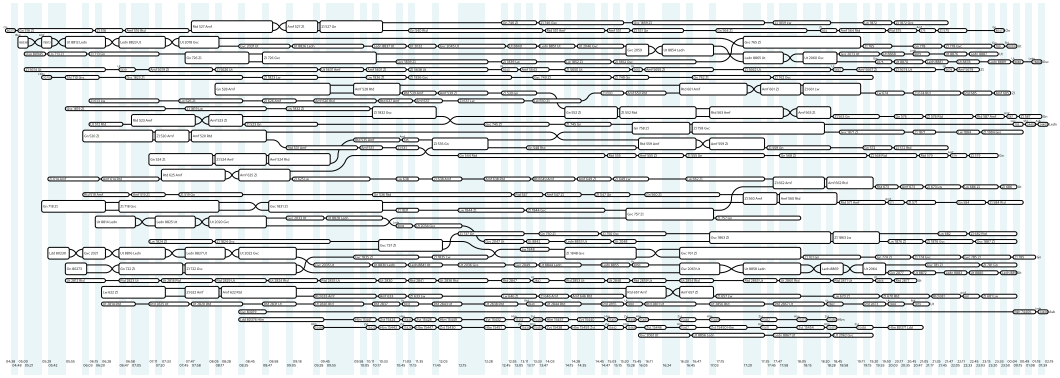


Figure 10 Storyline visualization of a rolling stock schedule using our drawing style.

Furthermore, we had to slightly adjust the ILP for crossing minimizing. We fixed the ordering of the characters within a meeting as they resemble the order of train units in a composition, which is relevant for planning. Additionally, we could have required that meetings belonging to the same trip must be drawn straight (as rectangles instead of as sequences of vertical bars) and that meetings cannot take part in a crossing, but for simplicity, we have not enforced this. However, we rewarded such solutions in the LP for linear wiggle height minimization using soft constraints. In the same way, we rewarded the vertical alignment of meetings where a train composed of the same units operates another trip.

An example of our visualization is shown in Figure 10 (larger versions are in the full version [7]). It depicts the same data set as [12, Fig. 1] (but unfortunately a different solution). Because of how we assign meetings to layers, the x-coordinates represent the departure and arrival times only vaguely. Also the width of a box does not represent the duration of a trip exactly. This was a deliberate (but optional) choice to achieve a more compact visualization.

7 Conclusion

We have given exact solutions for three variants of wiggle minimization. In particular, our efficient LP solution of LWHMin noticeably improves the aesthetic quality, while it does not cost much in terms of computation time (compared to crossing minimization).

We leave several questions open.

- We have shown that WCMin is NP-complete, but we can solve the restriction to two time steps efficiently. Is WCMin in FPT with respect to the number of time steps?
- Our geometric routing produces very readable drawings, but sometimes they are quite wide. Is there a more compact geometric solution that maintains the advantages of our method?
- Despite the rich literature on optimizing quality metrics in storyline visualization, there exists no formal user evaluation on the effects on readability of their optimization. In particular, do visualizations of rolling stock schedules as crossing- and wiggle-optimized storyline visualizations lead to improvements in train operation planning?

References

1 Dustin Arendt and Meg Pirrung. The "y" of it matters, even for storyline visualization. In Brian D. Fisher, Shixia Liu, and Tobias Schreck, editors, *Proc. Conference on Visual Analytics, Science and Technology (VAST 2017)*, pages 81–91, 2017. doi:10.1109/VAST.2017.8585487.

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 Ralf Borndörfer, Boris Grimm, Markus Reuther, and Thomas Schlechte. Optimization of handouts for rolling stock rotations. *J. Rail Transport Planning & Management*, 10:1–8, 2019. doi:10.1016/j.jrtpm.2019.02.001.

4 Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 7th edition, 2009.

5 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.

6 Mark de Berg and Amirali Khosravi. Optimal binary space partitions for segments in the plane. *Int. J. Comput. Geom. Appl.*, 22(3):187–206, 2012. doi:10.1142/S0218195912500045.

7 Alexander Dobler, Tim Hegemann, Martin Nöllenburg, and Alexander Wolff. Optimizing wiggles in storylines. *CoRR*, abs/2508.19802, 2025. arXiv:2508.19802.

8 Alexander Dobler, Michael Jünger, Paul J. Jünger, Julian Meffert, Petra Mutzel, and Martin Nöllenburg. Revisiting ILP models for exact crossing minimization in storyline drawings. In Stefan Felsner and Karsten Klein, editors, *Proc. Graph Drawing and Network Visualization (GD 2024)*, volume 320 of *LIPIcs*, pages 31:1–31:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.GD.2024.31.

9 Alexander Dobler, Martin Nöllenburg, Daniel Stojanovic, Anaïs Villedieu, and Jules Wulms. Crossing minimization in time interval storylines. In Clemens Huemer and Carlos Seara, editors, *Proc. European Workshop on Compututational Geometry (EuroCG 2023)*, pages 36:1–36:7, 2023. doi:10.48550/arXiv.2302.14213.

10 Theresa Fröschl. Minimizing wiggles in storyline visualizations. Master's thesis, Technische Universität Wien, 2018. doi:10.34726/hss.2018.53581.

11 Theresa Fröschl and Martin Nöllenburg. Minimzing wiggles in storyline visualizations. In Fabrizio Frati and Kwan-Liu Ma, editors, *Proc. Graph Drawing and Network Visualization (GD 2017)*, volume 10692 of *Lecture Notes in Computer Science*, pages 585–587. Springer, 2018. URL: https://www.ac.tuwien.ac.at/files/pub/fn-mwsv-18.pdf.

12 Boris Grimm, Rowan Hoogervorst, and Ralf Borndörfer. A comparison of two models for rolling stock scheduling. *Transport. Sci.*, 59:1101–1129, 2025. doi:10.1287/trsc.2024.0505.

13 Martin Gronemann, Michael Jünger, Frauke Liers, and Francesco Mambelli. Crossing minimization in storyline visualization. In Yifan Hu and Martin Nöllenburg, editors, *Proc. Graph Drawing and Network Visualization (GD 2016)*, volume 8901 of *Lecture Notes in Computer Science*, pages 367–381. Springer, 2016. doi:10.1007/978-3-319-50106-2_29.

14 Tim Hegemann. NarratiViz. Software, BMFTR grant 01IS22012C, swhId: swh:1:dir:a7efa9c0ca674b6cf25b1c4b2c1f4793cc814723 (visited on 2025-11-10). URL: https://github.com/hegetim/narrativiz, doi:10.4230/artifacts.25067.

15 Tim Hegemann and Alexander Wolff. Storylines with a protagonist. In Stefan Felsner and Karsten Klein, editors, *Proc. Graph Drawing and Network Visualization (GD 2024)*, volume 320 of *LIPIcs*, pages 26:1–26:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.GD.2024.26.

16 Narendra Karmarkar. A new polynomial-time algorithm for linear programming. *Comb.*, 4(4):373–396, 1984. doi:10.1007/BF02579150.

17 Nam Wook Kim, Stuart K. Card, and Jeffrey Heer. Tracing genealogical data with timenets. In Giuseppe Santucci, editor, *Advanced Visual Interfaces (AVI)*, pages 241–248. ACM Press, 2010. doi:10.1145/1842993.1843035.

18 Donald E. Knuth. *The Stanford GraphBase: A platform for combinatorial computing*. ACM Press, 1994.

19 Irina Kostitsyna, Martin Nöllenburg, Valentin Polishchuk, André Schulz, and Darren Strash. On minimizing crossings in storyline visualizations. In Emilio Di Giacomo and Anna Lubiw, editors, *Proc. Graph Drawing and Network Visualization (GD 2015)*, volume 9411 of *Lecture Notes in Computer Science*, pages 192–198. Springer, 2015. doi:10.1007/978-3-319-27261-0_16.

20 Shixia Liu, Yingcai Wu, Enxun Wei, Mengchen Liu, and Yang Liu. Storyflow: Tracking the evolution of stories. *IEEE Trans. Vis. Comput. Graph.*, 19(12):2436–2445, 2013. doi:10.1109/TVCG.2013.196.

21 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.

22 Randall Munroe. Movie narrative charts. Diagram available at https://xkcd.com/657/, 2009. Accessed 2017/04/03.

23 Michael Ogawa and Kwan-Liu Ma. Software evolution storylines. In Alexandru C. Telea, Carsten Görg, and Steven P. Reiss, editors, *Proc. ACM Symposium on Software Visualization (SoftVis 2010)*, pages 35–42, 2010. doi:10.1145/1879211.1879219.

24 Kozo Sugiyama, Shojiro Tagawa, and Mitsuhiko Toda. Methods for visual understanding of hierarchical system structures. *IEEE Trans. Syst. Man Cybern.*, 11(2):109–125, 1981. doi:10.1109/TSMC.1981.4308636.

25 Yuzuru Tanahashi, Chien-Hsin Hsueh, and Kwan-Liu Ma. An efficient framework for generating storyline visualizations from streaming data. *IEEE Trans. Vis. Comput. Graph.*, 21(6):730–742, 2015. doi:10.1109/TVCG.2015.2392771.

26 Yuzuru Tanahashi and Kwan-Liu Ma. Design considerations for optimizing storyline visualizations. *IEEE Trans. Vis. Comput. Graph.*, 18(12):2679–2688, 2012. doi:10.1109/TVCG.2012.212.

27 Thomas C. van Dijk, Martin Fink, Norbert Fischer, Fabian Lipp, Peter Markfelder, Alexander Ravsky, Subhash Suri, and Alexander Wolff. Block crossings in storyline visualizations. *J. Graph Alg. Appl.*, 21(5):873–913, 2017. doi:10.7155/jgaa.00443.

28 Thomas C. van Dijk, Fabian Lipp, Peter Markfelder, and Alexander Wolff. Computing storylines with few block crossings. In Fabrizio Frati and Kwan-Liu Ma, editors, *Proc. Graph Drawing and Network Visualization (GD 2018)*, volume 10692 of *Lecture Notes in Computer Science*, pages 365–378. Springer, 2018. doi:10.1007/978-3-319-73915-1_29.