Transportation Research Part C 84 (2017) 92–123

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Multi-level condition-based maintenance planning for railway infrastructures – A scenario-based chance-constrained approach

Zhou Su [a,*], Ali Jamshidi [b], Alfredo Núñez [b], Simone Baldi [a], Bart De Schutter [a]

[a] Delft Center for Systems and Control, Mekelweg 2, Delft, The Netherlands
[b] Section of Railway Engineering, Stevinweg 1, Delft, The Netherlands

A R T I C L E I N F O

Article history:
Received 28 October 2016
Received in revised form 21 August 2017
Accepted 22 August 2017
Available online 3 September 2017

Keywords:
Model predictive control
Condition-based maintenance
Railway infrastructure
Time-instant optimization
Chance-constrained optimization

A B S T R A C T

This paper develops a multi-level decision making approach for the optimal planning of maintenance operations of railway infrastructures, which are composed of multiple components divided into basic units for maintenance. Scenario-based chance-constrained Model Predictive Control (MPC) is used at the high level to determine an optimal long-term component-wise intervention plan for a railway infrastructure, and the Time Instant Optimization (TIO) approach is applied to transform the MPC optimization problem with both continuous and integer decision variables into a nonlinear continuous optimization problem. The middle-level problem determines the allocation of time slots for the maintenance interventions suggested at the high level to optimize the trade-off between traffic disruption and the setup cost of maintenance slots. Based on the high-level intervention plan, the low-level problem determines the optimal clustering of the basic units to be treated by a maintenance agent, subject to the time limit imposed by the maintenance slots. The proposed approach is applied to the optimal treatment of squats, with real data from the Eindhoven-Weert line in the Dutch railway network.

© 2017 Elsevier Ltd. All rights reserved.

1. Introduction

Maintenance is crucial for the proper functioning and lifetime extension of a railway network, which is composed of various infrastructures with different functions. The Dutch railway network, one of the most intensive railway networks in Europe, consists of tracks (6830 km), tunnels (5100), overhead wiring (4500 km), switches (7508), signaling systems, stations (388) and safety control systems. The degradation of an infrastructure can severely impair the performance of the whole railway network. One example is a squat, a typical type of Rolling Contact Fatigue (RCF), that accelerates rail degradation, which can potentially lead to derailment if not treated properly (Sandström and Ekberg, 2009). Grinding is effective for the treatment of early-stage squats, while rail replacement would be the only solution for severe-stage squats. Another example is ballast degradation, which can affect track geometries, causing unreliable support for sleepers and potential rail buckling and derailment (Ling et al., 2014; He et al., 2015). In this case, tamping is applied to correct the track geometry.

* Corresponding author.
E-mail addresses: Z.Su-1@tudelft.nl (Z. Su), A.Jamshidi@tudelft.nl (A. Jamshidi), A.A.NunezVicencio@tudelft.nl (A. Núñez), S.Baldi@tudelft.nl (S. Baldi), B.DeSchutter@tudelft.nl (B. De Schutter).

http://dx.doi.org/10.1016/j.trc.2017.08.018
0968-090X/© 2017 Elsevier Ltd. All rights reserved.

1.1. Maintenance for infrastructures

Maintenance can be either reactive or proactive. A shift from reactive maintenance to proactive solutions can be identified in several European countries in recent years (Zoeteman, 2001; Al-Douri et al., 2016). Condition-based maintenance (Kobbacy and Murthy, 2008; Ben-Daya et al., 2016), a proactive maintenance strategy where decision making is based on the observed "condition" of an asset, has received growing attention in various industrial fields (Jardine et al., 2006; Fararooy and Allan, 1995). We apply the concept of maintenance optimization (Dekker, 1996) in the planning of maintenance interventions[1] for railway infrastructures based on an explicit mathematical model describing the deterioration process of the condition (Scarf, 1997). This is different from data-driven approaches based on an expert system (Guler, 2012) or machine learning techniques (Li et al., 2014), where no explicit model of the deterioration dynamics is required (Scarf, 1997). Markov decision process and its variations are the most popular stochastic models used in maintenance planning of transportation infrastructures (Smilowitz and Madanat, 2000; Durango-Cohen and Madanat, 2008). A bi-variate Gamma process considering both longitudinal and transverse level is used to schedule tamping intervention for a French high-speed line in Mercier et al. (2012). In Zhang et al. (2013), a Weibull distribution is assumed for the condition deterioration time probability density function, and an optimal timetable of maintenance activities is determined for a regional railway network considering multiple tamping crews to minimize the negative effects on train schedule and maintenance cost. Not all stochastic approaches model the deterioration dynamics as a stochastic process (Frangopol et al., 2004). One example is the grey-box model proposed in Quiroga and Schnieder (2012) for the ageing process of track geometry. Deterministic models are relatively few in literature. A linear model is applied in Wen et al. (2016) to describe the degradation of track quality, and a quality-dependent-recovering upper bound is used to ensure that the improvement of track quality by tamping can never outperform the previous operation. An exponential model is used in Famurewa et al. (2015) to describe the nominal degradation of track geometry, where the improvement brought by tamping is modeled by an empirical regression model.

In this paper, we focus on the optimal planning of maintenance interventions for a railway infrastructure using a condition-based maintenance strategy. Our aim is to develop a comprehensive, systematic approach that is able to support the maintenance decision making for a wide range of railway infrastructures. In particular, we consider a railway infrastructure as a multi-component system (Nicolai and Dekker, 2008) with independent deterioration dynamics, which can be either linear or nonlinear. Moreover, we do not restrict the condition to take only discrete values, thus avoiding the suppression of the rich deterioration dynamics brought by the discretization of the originally continuous condition in most stochastic models (Frangopol et al., 2004).

1.2. Maintenance intervention planning

An optimization-based, multi-level approach is developed in this paper for the optimal planning of maintenance interventions for railway infrastructures like rail and ballast. A schematic plot for the multi-level approach is provided in Fig. 1. Three optimization problems, namely the intervention planning problem, the slot allocation problem, and the clustering problem, are solved at the high, middle, and low level, respectively. Based on the component-wise discrete-time prediction model of the condition of the infrastructure, at each time step the high-level intervention planning problem determines the optimal maintenance intervention for each component over a given prediction horizon. The sampling time, i.e. the length of each time step, is usually larger than one month because of the slow deterioration dynamics of a railway infrastructure. If a maintenance intervention is suggested at any time step at the high level, it should be performed within a traffic-free time slot (4–8 h at night) to avoid any disruption to the train service. However, it is not always possible to complete an intervention within such a short time slot, and a new operation must then be scheduled into a new time slot to finish the required intervention, resulting in an additional setup cost including machinery, logistic, personnel, etc. This gives rise to the middle-level slot allocation problem, which determines the time slots that optimize the trade-off between traffic disruption and the total setup cost associated with each maintenance slot, while guaranteeing that the total duration of the resulting maintenance slots is no less than the estimated maintenance time. According to the intervention plan, the low-level clustering problem then groups the basic units into clusters that can be treated within the allocated time slots. If the resulting clusters cannot cover all the basic units that need to be treated according to the high-level intervention plan because of insufficient time slots, then the slot allocation problem is solved again with a longer estimated maintenance time. This iterative procedure between the slot allocation problem and the clustering problem repeats until all the basic units that need to be treated are covered by a cluster. This cluster-wise work plan is then applied to the infrastructure, and the condition of each component is then regularly measured or updated by estimation.

1.3. State-of-the-art

We use Model Predictive Control (MPC) (Camacho and Alba, 2013; Rawlings and Mayne, 2009) as the basic scheme for the long-term optimal planning of maintenance interventions over a finite planning horizon. MPC has been applied to various

[1] Renewal is also considered as a maintenance intervention.

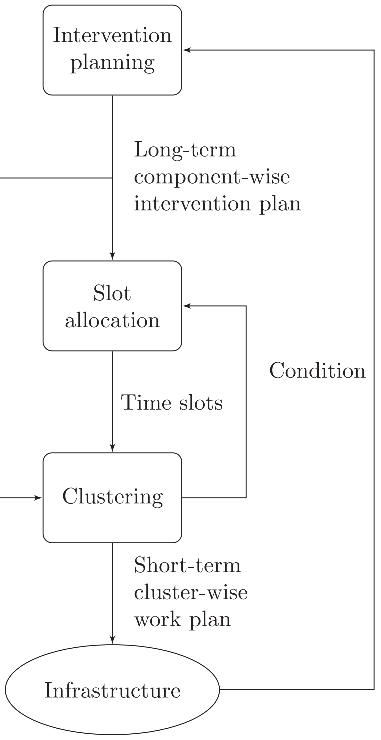


Fig. 1. Schematic plot of the proposed multi-level approach for optimal condition-based maintenance planning.

real-world decision making problems including risk management in semiconductor manufacturing and irrigation canals (Zafra-Cabeza et al., 2008; Zafra-Cabeza et al., 2011), inventory management in supply chains (Nandola and Rivera, 2013), and maintenance planning of railway infrastructures (Su et al., 2015). These real-world problems typically involve systems with both continuous and discrete dynamics. Such systems are usually modeled using the Mixed Logical Dynamical (MLD) framework (Bemporad and Morari, 1999), and a sequence of discrete control actions is determined by solving a mixed integer programming problem at each time step. An alternative technique, Time-Instant Optimization (TIO) (De Schutter and De Moor, 1998), is used in this paper. In the TIO-MPC scheme, a sequence of continuous time instants indicating the occurrence of each control action is optimized, resulting in a continuous, albeit non-smooth, optimization problem at each time step. For some real-world applications, like water level control for irrigation canals (van Ekeren et al., 2013; Sadowska et al., 2014), the number of admissible control actions, e.g. opening and closing of barriers, is relatively small even for a long prediction horizon. In this case, only a small number of time instants are needed to indicate when each control action occurs. This usually results in an optimization problem easier to solve than the large mixed integer programming problem in MLD-MPC. A comparison between the MLD and the TIO framework is given in Fig. 2, where $u(k)$ denotes the action performed at time step k, while t_{maint} and t_{renewal} are vectors containing all the time instants of the maintenance and renewal actions.[2] In this example, at most two maintenance actions and one renewal action can be performed within the 12-month prediction horizon. The MLD-MPC controller needs to optimize a discrete sequence of length 12 specifying which action (maintenance, renewal, or doing nothing) to be applied at each time step. However, the TIO-MPC controller only needs to optimize a continuous sequence of length 3, containing the time instants at which the two maintenance actions and one renewal action are performed, respectively. In this example, the TIO framework is a better choice than the MLD framework as it only needs a small number of continuous decision variables.

Maintenance decision making can be influenced by different randomness like model uncertainties, missing data, measurement error, etc. (Madanat, 1993). Robust control (Morari and Zafiriou, 1989), which guarantees good control performance and constraint satisfaction within a specific range of uncertainties, should be considered. Popular robust control approaches, like the min-max approach (Campo and Morari, 1987; Gruber et al., 2013), are often conservative, as the worst-case scenario does not always occur. To avoid conservatism, chance constraints (Prekopa, 1970) are considered, which guarantee that the constraint are satisfied with a probability no less than a confidence level. Chance-constrained MPC, which replaces all constraints with uncertainties by chance constraints, and optimizes the expectation of the objective function, has

[2] We use the notation $(v)_i$ to indicate the i-th element of the vector v.

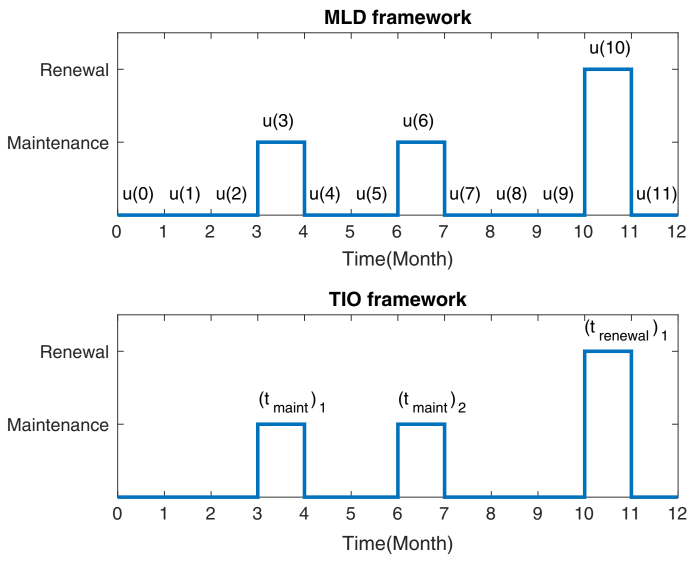


Fig. 2. Comparisons between MLD and TIO framework illustrated by a small example, in which at most two maintenance actions and one renewal actions can be performed to an infrastructure within the one-year prediction window.

been successfully applied to the management of drinking water networks (Grosso et al., 2014) and stock management in hospital pharmacy (Jurado et al., 2016). A scenario-based approach is adopted for tractability.

Maintenance interventions should be performed within traffic-free time slots to avoid disturbance or disruption to the current time table. However, unexpected incidents are unavoidable. Slight perturbations can be handled by robust time tables (e.g. Bešinović et al. (2016), Goverde et al. (2016)), but larger ones require real-time replanning of time tables (Quaglietta et al., 2016). An accurate estimation of the length of disruption (e.g. the disruption model described in Zilko et al. (2016)) can greatly benefit the decision-making in disruption management. From the maintenance agent's perspective, negative consequences of unexpected events, e.g. delaying or canceling of train services, can be minimized by considering cost of traffic disruption in the scheduling of maintenance activities. Maintenance scheduling considering disruption to train traffic has been extensively discussed in literature. An integer programming problem is formulated in Higgins (1998) to schedule different types of routine maintenance activities for a single railway line, minimizing disruption to and from train traffic, and the weighted completion time of all maintenance activities. A mixed integer programming problem is formulated in Budai et al. (2006) to minimizing track possession cost and maintenance costs for a single railway line, considering both routine activities and projects like grinding or tamping. The optimal long-term scheduling of railway maintenance projects is formulated as a time-space network model (Peng et al., 2011; Peng and Ouyang, 2012) to minimize the total travel costs of maintenance crews and disruption to train traffic for a large-scale railway network. A mixed integer linear programming model is developed in Caetano and Teixeira (2016) for optimal scheduling of ballast, rail, and sleeper renewal operations for a network, minimizing the expected life-cycle cost and track unavailability costs caused by renewal operations. In Peng and Ouyang (2014), the optimal clustering of maintenance jobs into projects to reduce the total maintenance cost for a railway network is formulated as a vehicle routing problem with side constraints. A metaheuristic using simulated annealing technique is developed in Santos and Teixeira (2012) to determine the optimal length of track to be treated by a tamping machine.

1.4. Contributions and structure of the paper

The major contributions of this paper include:

1. A multi-level decision making approach is developed taking into account both long-term and short-term objectives in condition-based maintenance planning, as well as the disruption to train traffic.
2. We apply a tractable scenario-based chance-constrained scheme for the high-level MPC controller to improve the robustness of the resulting intervention plan.
3. A case study with real data is performed for optimal treatment of squats for the Eindhoven-Weert line in the Netherlands.

Unlike previous studies on optimal planning of railway preventive maintenance activities (e.g. (Budai et al., 2006; Peng et al., 2011)), where condition deterioration is not explicitly considered, the proposed approach uses a condition-based maintenance strategy, in which both the condition deterioration and maintenance costs are minimized. Moreover, unlike other offline optimal condition-based maintenance planning approaches (e.g. (Wen et al., 2016)), where one optimization problem is solved for the whole planning horizon, we propose an online model-based approach, where the prediction is made based on the "actual" condition calculated from real-time measurements to avoid accumulation of estimation errors.

This paper is organized as follows: the general deterioration process of a railway infrastructure and the chance-constrained TIO-MPC scheme are briefly described in Section 2. The formal deterioration model is presented in Section 3, and the multi-level approach is formally explained in Section 4. A case study on the optimal treatment of squats is performed in Section 5, and conclusions and future directions are provided in Section 6.

2. Preliminaries

2.1. Railway defects and maintenance interventions

In this section, we briefly explain generic defects for railway infrastructures and the corresponding maintenance interventions. Note that in this paper, full renewal is also considered as a maintenance intervention. Fig. 3 shows the deterioration process of a generic defect (Esveld, 2001), e.g. a ballast defect, for one component of a railway asset, e.g. a section of ballast. The condition of the component is represented by one single measurable factor. The natural degradation[3] is shown by the green dashed line, which will eventually hit the operational limit, triggering a great hazard like derailment, if no adequate intervention is performed. An intervention is performed to improve the condition, when the predicted condition is close to the maintenance limit, which is usually much smaller than the operational limit to allow for sufficient safety margin. An intervention brings an abrupt improvement of the condition, quantified by the vertical drop of the degradation level after the intervention. Full renewal can always restore a component to an "as good as new" condition. In comparison, corrective maintenance, like tamping and grinding, is inefficient, in the sense that the level of improvement that can be achieved by a corrective maintenance becomes less the more it is applied. This is also shown in the monotonically increasing dashed red line connecting the conditions immediately after each maintenance intervention. Moreover, the deterioration also becomes faster after each maintenance intervention, demonstrated by the more steep natural degradation between two consecutive maintenance interventions. Finally, renewal becomes the only option when a maintenance intervention becomes so inefficient that it can no longer improve the condition.

2.2. MPC

In this section, we briefly explain the basic mechanism of chance-constrained TIO-MPC, the formal description of which is presented in the high-level problem in Section 4.1. Based on a dynamic model for the process and the current state, an MPC controller predicts the optimal sequence of control actions that optimizes an objectives function subject to constraints for a given prediction horizon N_P. According to the receding horizon principle, only the first entry of the sequence of control actions is applied to the system, and the controller moves to the next time step, solving a new optimization problem using an updated state.

The generic deterioration process of a railway infrastructure contains both discrete and continuous dynamics, as shown in Fig. 3. MLD-MPC recasts the hybrid dynamics model into an MLD system, and solves a mixed integer programming problem to determine a sequence of discrete control inputs. TIO-MPC, on the other hand, determines a sequence of continuous time instants at which each maintenance intervention, e.g. corrective maintenance and renewal, take place within the prediction period.[4] A nonlinear conversion to the original discrete control inputs can be made by rounding the continuous time instants to the nearest discrete time steps. Thus a continuous nonlinear optimization problem must be solved at each time step of TIO-MPC. This nonlinear optimization problem is intrinsically non-smooth because of the rounding. Although non-smooth optimization problems are in general difficult to solve, the number of decision variables in TIO-MPC is usually small, as the maximum number of interventions is small even for a long prediction horizon, e.g. at most two grindings per year. In this case, TIO-MPC is more promising than MLD-MPC in terms of computational efficiency.

The original hybrid model and the converter that converts time instants to control inputs form the TIO prediction model, which is stochastic in practice due to the presence of various uncertainties. A chance-constrained optimization problem is solved, where the expectation of the objective function is optimized, and the satisfaction of each stochastic constraint is guaranteed with a probability no less than a confidence level.[5] The expectation and probability are both computed over the complete set of realizations of uncertainties over the entire prediction period. This is difficult when the set of uncertainties

[3] The natural degradation is only assumed to be exponential in this example for demonstration purpose. In general, it can be described by any monotonically increasing function.

[4] TIO-MPC and MLD-MPC are not equivalent. The former has less degree of freedom, when the maximum number of interventions is strictly less than the prediction horizon.

[5] We consider joint chance constraint, in which a common confidence level must be guaranteed for all stochastic constraints.

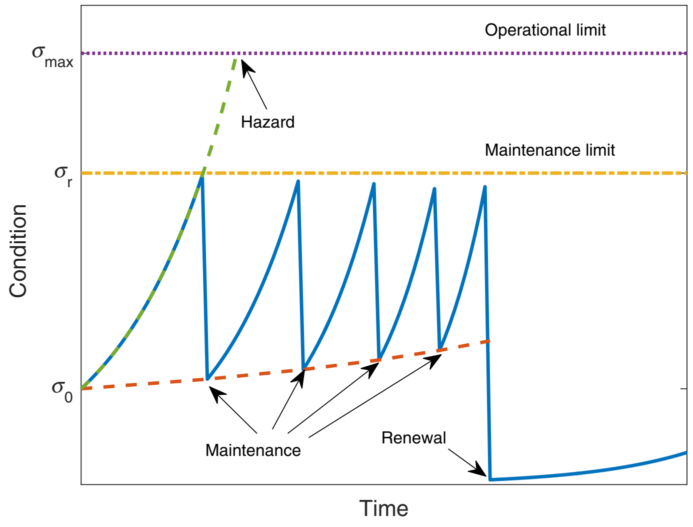


Fig. 3. Deterioration process of a generic defect with an exponential deterioration. A higher value indicates a worse condition. The initial condition is denoted by σ_0, while σ_r and σ_{max} represent the maintenance and operational limit, respectively.

is large, and no assumption is made on the uncertainties. A scenario-based approach is usually adopted, which will be explained in Section 4.1.2.

3. Deterioration model

In this section, we develop a discrete-time state space model to describe the deterioration process of a generic defect of a railway infrastructure, e.g. ballast defect or squat growth in a certain length of track in the railway network. The infrastructure is composed of n components, e.g. n segments of track, with independent condition deterioration dynamics. Let the vector

$$x_j(k) = \begin{bmatrix} x_j^{con}(k) \\ x_j^{aux}(k) \end{bmatrix} \in \mathcal{X}_j$$

denote the state of the j-th component of the infrastructure at time step k. The state of the j-th component includes its condition x_j^{con}, as well as other auxiliary variables collected in the vector x_j^{aux}. These auxiliary variables, e.g. the condition after the last maintenance intervention, are necessary to model the inefficiency of corrective maintenance.

Let $\mathcal{U} = \{a_0, \ldots, a_N\}$ denote the set of all possible actions[6] (including no maintenance) that can be applied to a component, where N is the number of possible interventions. The first action a_0 represents no maintenance, and the last action a_N represents full renewal. Let $u_j(k) \in \mathcal{U}$ denote the maintenance action applied to the j-th component at time step k. Furthermore, we define $u(k) = [u_1(k)^T \ldots u_n(k)^T]^T \in \mathcal{U}^n$ as the maintenance action performed on the multi-component infrastructure at time step k.

The deterioration process is also affected by various uncertainties like measurement error and model inaccuracies. We denote $\theta_j(k) \in \Theta_j$ as the realization of the uncertainties related to the deterioration of component j. Similarly, we define $\theta(k) = [\theta_1^T(k) \ldots \theta_n^T(k)]^T \in \Theta$ as the realization of the uncertainties for the whole asset. The following generic model is proposed to describe the stochastic deterioration process of the j-th component of the asset:

$$\begin{aligned} x_j(k+1) &= f_j(x_j(k), u_j(k), \theta_j(k)) \\ &= \begin{cases} f_j^0(x_j(k), \theta_j(k)) & \text{if } u_j(k) = a_0 \text{ (no maintenance)} \\ f_j^q(x_j(k), \theta_j(k)) & \text{if } u_j(k) = a_q \quad \forall q \in \{1, \ldots, N-1\} \\ f_j^N(\theta_j(k)) & \text{if } u_j(k) = a_N \text{ (full renewal)} \end{cases} \\ &\forall j \in \{1, \ldots, n\}. \end{aligned} \tag{1}$$

[6] For the sake of simplicity we consider the same set of available actions for each component of the infrastructure.

The natural degradation f_j^0 only depends on the current condition and uncertainties. Full renewal f_j^N restores the component to an "as good as new" condition, regardless of the current condition or the history of maintenance interventions. However, renewal is also stochastic, and the exact condition after a renewal is uncertain. The effect of other interventions on a component depends both on the current condition and the history of maintenance.

The deterioration dynamics of the whole asset can then be written as:

$$x(k+1) = f(x(k), u(k), \theta(k)) \tag{2}$$

where $f = [f_1^{\mathrm{T}} \dots f_n^{\mathrm{T}}]^{\mathrm{T}}$ is a vector-valued function.

In practice, constraints must be considered for each individual component. We call these constraints local constraints, as they are only dependent on the condition, action, and uncertainties of the individual component. One crucial local constraint is that the condition of each component should not exceed the maintenance limit. In addition to local constraints for individual components, we also consider global constraints on the whole asset. Such global constraints usually arise from limited resources available for maintenance and renewal of the asset, e.g. budget and working time limits. To summarize, we define the constraints that need to be considered at time step k for the whole system as:

$$g(x(k), u(k), \theta(k)) \leqslant 0 \tag{3}$$

In summary, the deterioration process of the infrastructure can then be described by the whole-system dynamics (2) subject to the constraint (3).

4. Multi-level intervention planning

A multi-level decision making scheme is developed for the optimal planning of maintenance interventions. As explained in Section 1, the motivation to adopt a multi-level scheme includes different time scales of the deterioration process and traffic schedule, and computational tractability. A flow chart is presented in Fig. 4 to illustrate the proposed multi-level scheme. The condition of the railway infrastructure is monitored at every time step. The current measurements for each basic unit is collected and processed, then aggregated to represent the condition of each component. Based on the current condition of each component, the high-level MPC controller determines the maintenance plan for the current time step that optimizes the trade-off between condition deterioration and maintenance cost. If no intervention is suggested for any component for the current time step, the MPC controller moves to the next time step. The condition of the infrastructure is updated by new measurements, or estimates if no new measurements are available. The iterative procedure between the middle-level and low-level problems is triggered when an intervention, e.g. grinding or tamping, is suggested at the high level for at least one component for the current time step. Let $\widehat{T}_{\mathrm{Maint}}$ denote the estimated time to complete the suggested intervention. The optimal maintenance slots are allocated at the middle level, minimizing the trade-off between the total setup cost for the maintenance slots and the cost of disruption to the railway traffic, guaranteeing that the resulting maintenance time is no less than the estimated maintenance time $\widehat{T}_{\mathrm{Maint}}$. The resulting maintenance slots are then fed to the low level. Depending on their location and condition, the basic units inside the components that require the corresponding intervention are grouped into clusters, that must be treated within the time slots determined at the middle level. If the resulting clusters at the low level cannot cover all the basic units, depending on the number and severities of the remaining basic units, an evaluation is made to determine whether to solve the middle-level problem again with a larger estimated maintenance time in order to cover the remaining basic units. A value v_{rem} is calculated for the remaining basic units, which is the ratio of the accumulated severities of the remaining basic units and the accumulated severities of all the basic units that needed to be treated. An additional maintenance time $\Delta\widehat{T}$ is also estimated, depending on the number of the remaining basic units. A factor μ is assigned to $\Delta\widehat{T}$ to convert the additional maintenance time into a "cost". The detailed explanation on how to compute v_{rem} and $\Delta\widehat{T}$ is given in Section 4.3. If $v_{\mathrm{rem}} - \mu\Delta\widehat{T} \leqslant 0$, indicating that the benefit of covering the remaining basic units does not justify the cost of the required additional maintenance time, the iterative procedure terminates and the uncovered basic units remain uncovered. If $v_{\mathrm{rem}} - \mu\Delta\widehat{T} > 0$, indicating that it is worthwhile to cover the remaining basic units with additional maintenance time, the middle-level problem is solved again with a longer estimated maintenance time $\widehat{T}_{\mathrm{Maint}} + \Delta\widehat{T}$.

The vector ξ contains the positions of all basic units. The conditions of all basic units at time step k are collected in the vector $w(k)$. In particular, let $w(0)$ denote the initial conditions of the basic units. Let $t_{\mathrm{sl}}(k)$ and $\phi(k)$ denote the solutions of the middle-level and the low-level problem, i.e. the resulting time slots and clusters, at time step k, respectively. The multi-level framework can be demonstrated by the pseudocode in Algorithm 1.

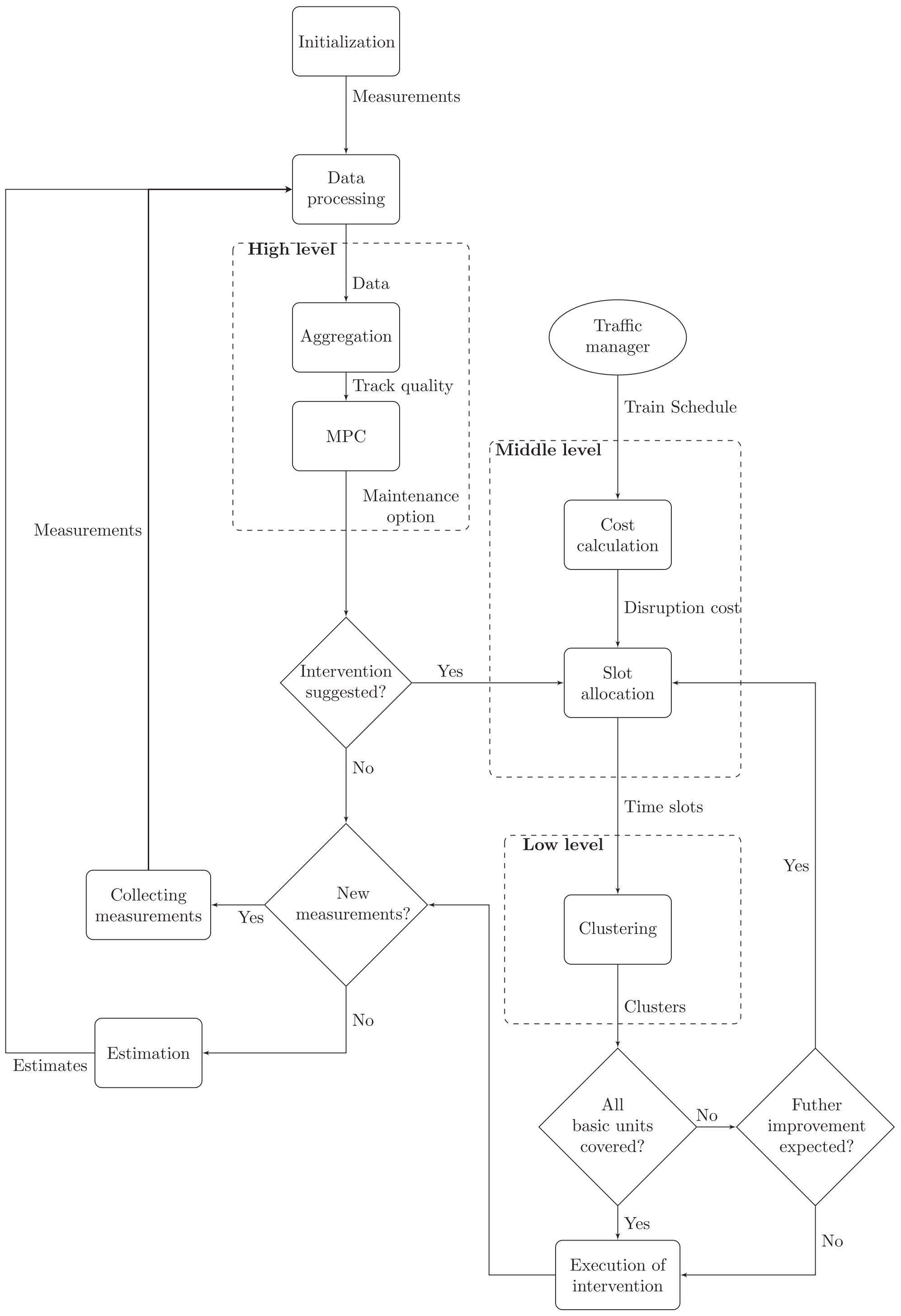


Fig. 4. Flowchart of the proposed multi-level approach, which can be implemented for online condition-based maintenance planning.

Algorithm 1. Procedure of the multi-level approach.

```
Function HighLevel_MPC($w(0)$, $\xi$)
    // Initialize the condition of the components
    $k \leftarrow 1$;
    $w(k) \leftarrow w(0)$;
    $x(k) \leftarrow$ Aggregate($w(k)$, $\xi$);
    while $k \leq k_{\text{end}}$ do
        // Solve the MPC optimization problem at time step $k$
        $u(k) \leftarrow$ MPC_Optimize($x(k)$);
        /* Middle- and low-level problems are triggered whenever an
           intervention is suggested                                  */
        for each intervention $a_l$ do
            if any $u_j(k) = a_l$ then
                /* Solve the middle-level problem to determine the time slots
                   for intervention $a_l$                                 */
                $t_{\text{sl}}(k) \leftarrow$ MiddleLevel($u(k)$, $a_l$, $\hat{T}_{\text{Maint}}$);
                /* Solve the low-level problem to determine the clusters for
                   intervention $a_l$                                     */
                $\phi(k) \leftarrow$ LowLevel($w(k)$, $\xi$, $t_{\text{sl}}(k)$, $a_l$);
                while a basic unit outside $\phi(k)$ do
                    Compute $\nu_{\text{rem}}$ and $\Delta\hat{T}$ using (38) and (39);
                    if $v_{\text{rem}} - \mu\Delta\hat{T} > 0$ then
                        $\hat{T}_{\text{Maint}} \leftarrow \hat{T}_{\text{Maint}} + \Delta\hat{T}$;
                        $t_{\text{sl}}(k) \leftarrow$ MiddleLevel($u(k)$, $a_l$, $\hat{T}_{\text{Maint}}$);
                        $\phi(k) \leftarrow$ LowLevel($w(k)$, $\xi$, $t_{\text{sl}}(k)$, $a_l$);
                    else
                        break
                end
                // Apply intervention $l$ to the infrastructure
                Intervention($\phi(k)$, $a_l$);
            // Update conditions of basic units
            if New measurements available then
                $w(k+1) \leftarrow$ New measurements;
            else
                $w(k+1) \leftarrow$ Simulate($w(k)$, $\xi$, $\phi(k)$);
        end
        $x(k+1) \leftarrow$ Aggregate($w(k+1)$, $\xi$);
        $k \leftarrow k+1$;
    end
```

4.1. High-level intervention planning problem

A scenario-based chance-constrained MPC controller is developed for the deterioration process described by (2) and (3) to determine the optimal maintenance and renewal interventions for each component of the asset. A TIO approach is applied to transform the MPC optimization problem with both continuous and discrete decision variables to a continuous non-smooth optimization problem. First, we recast the hybrid dynamic model (2) and (3) into a TIO prediction model. Then we select a set of representative scenarios from the entire set of realizations of the uncertainties within the prediction period. Chance-constrained MPC is then applied to the scenario-based TIO prediction model.

4.1.1. TIO prediction model

A TIO prediction model is developed based on the original hybrid deterioration model (2) and (3), which contains both continuous and discrete dynamics. Let N_{P} and N_{C} denote the length of the prediction and control horizons, respectively. Denote $\hat{x}(k+l|k)$ as the estimated state at time step $k+l$ based on the information available at time step k. The sequence of estimated states, control inputs and uncertainties within the prediction period can be defined as:

$$\begin{aligned}\tilde{x}(k) &= [\hat{x}^{\mathrm{T}}(k+1|k)\dots\hat{x}^{\mathrm{T}}(k+N_{\mathrm{P}}|k)]^{\mathrm{T}}\\ \tilde{u}(k) &= [u^{\mathrm{T}}(k)\dots u^{\mathrm{T}}(k+N_{\mathrm{P}}-1)]^{\mathrm{T}}\\ \tilde{\theta}(k) &= [\theta^{\mathrm{T}}(k)\dots\theta^{\mathrm{T}}(k+N_{\mathrm{P}}-1)]^{\mathrm{T}}\end{aligned}$$

The N_{P}-step prediction model can then be formulated as:

$$\tilde{x}(k) = \tilde{f}(x(k), \tilde{u}(k), \tilde{\theta}(k)) \tag{4}$$

$$\tilde{g}(x(k), \tilde{u}(k), \tilde{\theta}(k)) \leqslant 0 \tag{5}$$

where the function $\tilde{f}$ can be derived from recursive substitution of (2), as in standard MPC. The function $\tilde{g}$ can be derived similarly.

Recall that $u(k) \in \mathcal{U} = \{a_0, \dots, a_N\}$, where the option a_0 indicates “no intervention” and the last intervention a_N is full renewal. TIO-MPC first fixes the maximum number of times that each of the N interventions can be performed within the prediction period, and optimizes the continuous time instants to perform the interventions. Formally, let $v_{j,q}$ denote the maximum number of occurrences of intervention q for component j with the prediction period. All the relative time instants at which intervention q occurs in the prediction horizon for component j at time step k are collected in the vector $t_{j,q,k}$ of length $v_{j,q}$. The sequence of time instants to be optimized at time step k can then be written as:

$$\tilde{t}(k) = [\underbrace{t_{1,1,k}^{\mathrm{T}}\dots t_{1,N,k}^{\mathrm{T}}}_{t_1^{\mathrm{T}}(k)}\dots\underbrace{t_{n,1,k}^{\mathrm{T}}\dots t_{n,N,k}^{\mathrm{T}}}_{t_n^{\mathrm{T}}(k)}]^{\mathrm{T}} \tag{6}$$

The sequence of time instants $\tilde{t}(k)$ can be converted into the sequence of control inputs $\tilde{u}(k)$ by rounding to the nearest discrete time steps. The rounding function is always non-smooth, so TIO-MPC must solve a non-smooth optimization problem at each time step even for systems with linear dynamics.

For the situation of $N_{\mathrm{C}} < N_{\mathrm{P}}$, we have

$$u_j(k+l) = a_0 \quad \forall l \in \{N_{\mathrm{C}}\dots N_{\mathrm{P}}-1\}, \forall j \in \{1,\dots,n\}$$

This means no intervention is applied beyond the control horizon.

An example to explain the sequence of continuous time instants $\tilde{t}(k)$ and how it is converted to the sequence of discrete control inputs $\tilde{u}(k)$ is given in Fig. 5. In this example, the resulting sequence of time instants for component j at time step k is $t_j^{\mathrm{T}}(k) = [t_{j,1,k}^{\mathrm{T}}\, t_{j,2,k}^{\mathrm{T}}]^{\mathrm{T}}$. Among the two time instants for intervention a_1, only $(t_{j,1,k})_1$, which is located within the control period (indicated by the dashed vertical line), is rounded to the nearest relative time step l, indicating $u_j(k+l) = a_1$. The second time instant $(t_{j,1,k})_2$, is neglected as it is outside the control period $N_{\mathrm{C}}T_s$. Similarly, the first and only time instant for a_2 is within the control period, thus we have $u_j(k+m) = a_2$ as m is the time step closest to $(t_{j,2,k})_1$.

The following linear constraints should be considered for the time instants:

$$(t_{j,q,k})_1 \geqslant t_{j,q}^{\min} \tag{7}$$

$$(t_{j,q,k})_{v_{j,q}} \leqslant t_{j,q}^{\max} \tag{8}$$

$$(t_{j,q,k})_{i+1} - (t_{j,q,k})_i \geqslant \Delta t_{j,q}^{\min} \quad \forall i \in \{1,\dots,v_{j,q}-1\} \tag{9}$$

$$t_{j,q}^{\max} = N_{\mathrm{C}}T_s + v_{j,q}\Delta t_{j,q}^{\min} \tag{10}$$

$$\forall j \in \{1,\dots,n\} \quad \forall q \in \{1,\dots,N\}.$$

Constraints (7) and (8) specify the lower and upper bound of the time instants for intervention q on component j. The lower bound $t_{j,q}^{\min}$ is especially useful to address the issue on early planning. For example, if every maintenance intervention must be

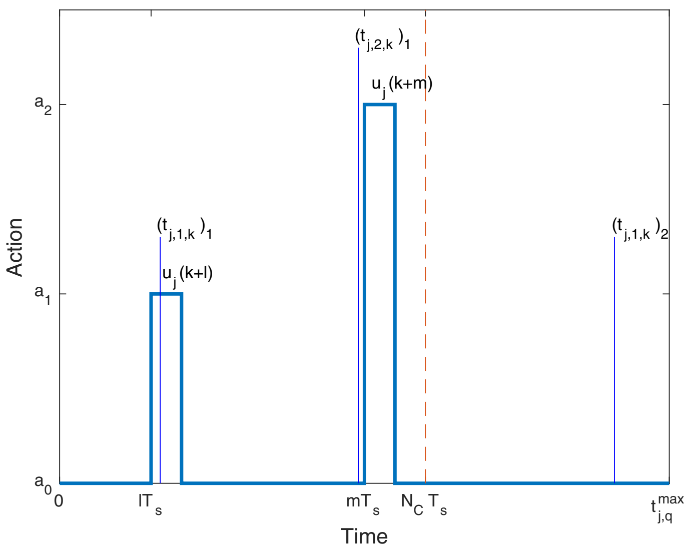


Fig. 5. Example illustrating the time instants in TIO-MPC. In this example, we have $u_j \in \{a_0, a_1, a_2\}$. Interventions a_1 and a_2 can be applied to component j at most twice and once, respectively.

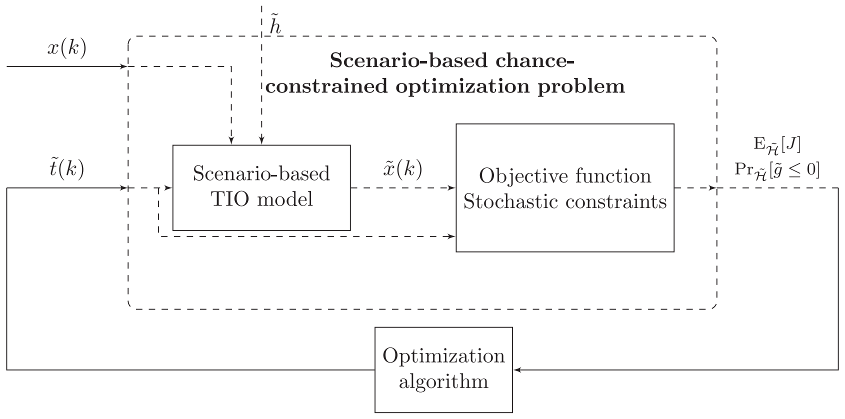

$\tilde{h}$
$x(k)$
$\tilde{t}(k)$
Scenario-based chance-constrained optimization problem
Scenario-based TIO model
$\tilde{x}(k)$
Objective function
Stochastic constraints
$\mathrm{E}_{\tilde{\mathcal{H}}}[J]$
$\mathrm{Pr}_{\tilde{\mathcal{H}}}[\tilde{g} \leq 0]$
Optimization algorithm

Fig. 6. Optimization scheme for scenario-based chance-constrained TIO-MPC. Unlike nominal MPC, chance-constrained MPC optimizes the expectation of the objective function, while guaranteeing a confidence level of probability of constraint satisfaction.

planned six months ahead, then we can simply set $t^{\min}_{j,q}$ to be equal to six months. The upper bound $t^{\max}_{j,q}$ is calculated in (10) to allow the situation of no intervention planned within the control period, where T_s is the sampling time (usually in months), and $\Delta t^{\min}_{j,q}$ is the minimal interval between two consecutive intervention of the same type, as specified in constraint (9).

The constraints (7)–(10) must be included in the optimization problem at each time step. Note that there is no stochasticity associated with these constraints, and they can be treated as normal linear constraints.

4.1.2. Scenario-based chance-constrained MPC

An scenario-based chance-constrained MPC controller is developed based on the TIO prediction model in Section 4.1.1. The optimization problem at each time step is illustrated by the schematic plot in Fig. 6.

In practice, the set Θ containing all possible realizations of the uncertainties for a railway infrastructure might be very large. The set of all possible realizations of uncertainties over the whole prediction period, $\tilde{\Theta} = \Theta^{N_\mathrm{P}}$, might be huge for a long prediction horizon. For tractability, a relatively small number of representative scenarios is selected from the set $\tilde{\Theta}$. Let $\tilde{\mathcal{H}} \subset \tilde{\Theta}$ denote the set of representative scenarios, the following scenario-based TIO prediction model can then be derived for any $\tilde{h} \in \tilde{\mathcal{H}}$:

$$\tilde{x}(k) = \tilde{f}_{\mathrm{TIO}}(x(k), \tilde{t}(k), \tilde{h}) \quad (11)$$

$$\tilde{g}_{\mathrm{TIO}}(x(k), \tilde{t}(k), \tilde{h}) \leqslant 0. \quad (12)$$

Define

$$J(k) = J_{\mathrm{Deg}}(k) + \lambda\mu J_{\mathrm{Maint}}(k) \tag{13}$$

as the objective function that needs to be minimized at each time step k. The parameter μ is a scaling factor, and the parameter λ captures the trade-off between the condition of the infrastructure and the maintenance cost.

The first part

$$J_{\mathrm{Deg}}(k) = \sum_{j=1}^{n}\sum_{l=1}^{N_{\mathrm{P}}} |x_j^{\mathrm{con}}(k+l) - \underline{x}_j^{\mathrm{con}}|_q \tag{14}$$

minimizes the magnitude of condition degradation, where the norm $|\cdot|_q$ can be the 1-norm, 2-norm, or infinity norm, for $q = 1, 2, \infty$, respectively.

The second part in the objective function is the accumulated maintenance cost, which can be formulated as:

$$J_{\mathrm{Maint}}(k) = \sum_{j=1}^{n}\sum_{l=1}^{N_{\mathrm{P}}}\sum_{q=1}^{p} \gamma_{j,q} I_{u_j(k+l-1)=a_q} \tag{15}$$

where the binary indicator function I_X takes value 1 if the statement X is true, otherwise it takes value 0, and the parameter $\gamma_{j,q}$ represents the required cost if intervention a_q is applied to component j.

As shown in (13)–(15), the objective function $J(k)$ is a function of $\tilde{x}(k)$ and $\tilde{u}(k)$. Using the TIO converting rule, the value of the objective function at time step k can be rewritten as:

$$J(k) = f_{\mathrm{opt}}(x(k), \tilde{t}(k), \tilde{h}) \tag{16}$$

Finally, the continuous, nonlinear optimization problem to be solved at each time step by the scenario-based chance-constrained TIO-MPC controller can be formulated as:

$$\min_{\tilde{t}(k)} \mathrm{E}_{\tilde{\mathcal{H}}}[f_{\mathrm{opt}}(x(k), \tilde{t}(k), \tilde{h})] \tag{17}$$

$$\text{Subject to}: \ \Pr_{\tilde{\mathcal{H}}}[\tilde{g}_{\mathrm{TIO}}(x(k), \tilde{t}(k), \tilde{h}) \leqslant 0] \geqslant \eta \tag{18}$$

$$g_{\tilde{t}}(\tilde{t}(k)) \leqslant 0 \tag{19}$$

where constraint (19) is the compact expression of constraints (7)–(10), and $\eta \in (0, 1)$ is the confidence level of the chance constraints.

The goal is to minimize the expected value of the objective function f_{opt} over the set of all representative scenarios $\tilde{\mathcal{H}}$. Note that the expected value of the objective function f_{opt} can be computed as

$$\mathrm{E}_{\tilde{\mathcal{H}}}[f_{\mathrm{opt}}(x(k), \tilde{t}(k), \tilde{h})] = \sum_{\tilde{h}\in\tilde{\mathcal{H}}} p(\tilde{h}) f_{\mathrm{opt}}(x(k), \tilde{t}(k), \tilde{h}), \tag{20}$$

where $p(\tilde{h})$ is the probability of scenario $\tilde{h}$. In a similar way, the chance constraint (18) can be reformulated as

$$\sum_{\tilde{h}\in\tilde{\mathcal{H}}} p(\tilde{h}) I_{\tilde{g}_{\mathrm{TIO}}(x(k), \tilde{t}(k), \tilde{h}) \leqslant 0} \geqslant \eta. \tag{21}$$

Because of the rounding procedure in the TIO prediction model, (17)–(19) is a non-smooth optimization problem. Hence, derivative-free or direct search algorithms Lewis et al. (2000) like a genetic algorithm or pattern search should be considered. Pattern search with multi-start is used in the case study.

4.2. Middle-level slot allocation problem

The middle-level slot allocation problem is triggered at any time step at which an intervention is recommended at the high level. This problem is solved to determine the maintenance time slots for the corresponding intervention, optimizing the trade-off between traffic disruption and the total setup costs to complete the corresponding intervention, as stated in Section 1.2. The planning horizon of the slot allocation problem is from the current time step to the next time step (e.g. from October to November), as each intervention suggested at the high level should be completed within the sampling time (e.g. one month).

Let N_{sl} denote the number of time slots available at the current time step of the high-level controller. Based on the corresponding train schedule, we can evaluate the cost of traffic disruption as a piecewise-constant function of track possession time. Traffic-free time interval is assigned a zero-cost, while other non-traffic-free intervals are associated with different costs of disruption. Let N_τ denote the number of intervals of this piecewise-constant function, and let c_j denote the cost of disruption of the j-th interval. An illustration of this piecewise-constant function is given in Fig. 7.

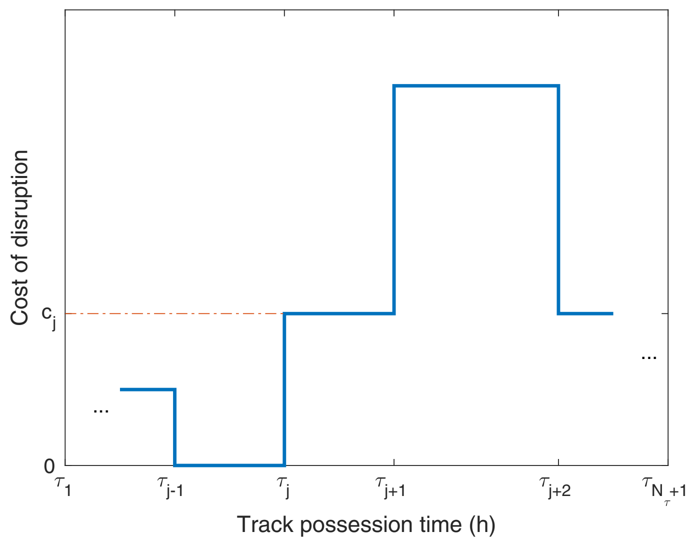


Fig. 7. An example of the piecewise-constant function of disruption cost, where τ_1 and $\tau_{N_\tau+1}$ coincide with the start of the current and next time steps, respectively. The cost of disruption is c_j if the j-th interval $[\tau_j, \tau_{j+1}]$ is occupied for maintenance.

Similar to the TIO technique used in the high-level MPC controller, we define t_i^{start} and t_i^{end} as the start and end time instants of the i-th maintenance slot at the current time step, respectively. Let the positive parameter $\Delta\tau_{\min}$ denote the minimal length of a time slot. Recall that $\widehat{T}_{\text{Maint}}$ is the estimated maintenance time, and let c_{sl} denote the setup cost associated with a maintenance slot. Moreover, a fixed setup time, denoted by T_{set}, is also associated with a maintenance operation. This setup time includes the time to prepare and finish a maintenance operation in a time slot. The middle-level optimization problem can then be formulated as:

$$\min_{\{(t_i^{\text{start}}, t_i^{\text{end}})\}_{i=1}^{N_{\text{sl}}}} \sum_{i=1}^{N_{\text{sl}}}\sum_{j=1}^{N_\tau} c_j\big(\max(\min(\tau_{j+1}, t_i^{\text{end}}), \tau_j) - \min(\max(\tau_j, t_i^{\text{start}}), \tau_{j+1})\big) + \lambda_{\text{sl}}\sum_{i=1}^{N_{\text{sl}}} c_{\text{sl}} I_{t_i^{\text{end}} \leqslant \tau_{N_\tau+1}} \tag{22}$$

subject to

$$t_1^{\text{start}} \geqslant \tau_1 \tag{23}$$

$$t_{N_{\text{sl}}}^{\text{end}} \leqslant \tau_{\max} \tag{24}$$

$$t_i^{\text{end}} - t_i^{\text{start}} \geqslant \Delta\tau_{\min} \quad \forall i \in \{1, \ldots, N_{\text{sl}}\} \tag{25}$$

$$t_i^{\text{end}} + \epsilon \leqslant t_{i+1}^{\text{start}} \quad \forall i \in \{1, \ldots, N_\tau - 1\} \tag{26}$$

$$\tau_{\max} = \tau_{N_\tau+1} + 2N_{\text{sl}}\Delta\tau_{\min} \tag{27}$$

$$t_i^{\text{start}} \leqslant \tau_{N_\tau+1} \Rightarrow t_i^{\text{end}} \leqslant \tau_{N_\tau+1} \tag{28}$$

$$\sum_{i=1}^{N_{\text{sl}}} I_{t_i^{\text{end}} \leqslant \tau_{N_\tau+1}} \cdot \left(t_i^{\text{end}} - t_i^{\text{start}} - T_{\text{set}}\right) \geqslant \widehat{T}_{\text{Maint}}. \tag{29}$$

The first term in the objective function (22) is the total cost of disruption, while the second term represents the number of *active* time slots, i.e. those located within the planning period. Constraints (23) and (24) are the lower and upper bounds of the time slots, respectively. Similar to TIO, the upper bound $\tau_{\max}$ is given in (27) to allow for the situation with no active time slots. Constraint (25) guarantees that each time slot is larger than the minimum length $\Delta\tau_{\min}$, while constraint (26) ensures that there is no overlap between the time slots. Constraint (28) excludes the situation of fractional time slots, where the starting instant is inside the planning period while the end instant is outside the planning period. Finally, constraint (29) guarantees that the resulting time slots are sufficient to perform all the maintenance interventions suggested at the high level.

The optimization problem (22)–(29) is a nonsmooth nonlinear programming problem. However, it can be converted into to an MILP problem by introducing new binary and auxiliary variables and linear constraints, following the procedure described in Bemporad and Morari (1999) for MLD systems. The detailed transformation procedure is described in Appendix C.1. The number of binary variables in the transformed MILP problem is $2N_{\text{sl}}(N_\tau + 1)$. As the maximum number of time slots is in general small (say, less than 5) in practice, the resulting MILP problem can be solved exactly when the number of intervals in the piecewise-constant function of disruption cost is not too large (say, less than 500).

4.3. Low-level clustering problem

The low-level problem is triggered whenever an intervention is suggested for any component by the high-level controller. A nonsmooth nonlinear programming problem is solved to determine the optimal execution plan for each active time slots determined at the middle level. The resulting execution plan groups various basic units into different clusters depending on their location and condition. Only the basic units located inside a cluster will be treated. The low-level problem determines the optimal start and end position of each cluster, trying to cover as many severe basic units as possible inside a cluster, subject to the maintenance time slot. Similar to the middle-level problem, different low-level problems must be solved if different interventions are suggested at the high level. Moreover, for each intervention, we only consider the basic units inside the components where the corresponding intervention is prescribed. This further reduces the size of the optimization problem.

Let N^{bu} denote the total number of basic units where the corresponding intervention is suggested by the high-level controller, while the vectors ξ and w contain the positions and conditions of these basic units respectively. Furthermore, let ξ_k and w_j denote the position and condition of the j-th basic unit. The basic units are all located inside the planning range $[\underline{\xi}, \overline{\xi}]$. Let T_s^{sl} denote the duration of the s-th resulting active time slot from the middle-level problem. Let $\mathcal{I}_s$ denote the set of indices of the basic units that need to be treated at time slot s. In particular, we have $\mathcal{I}_1 = \{1, \ldots, N^{\mathrm{bu}}\}$, and $\mathcal{I}_s$ contains all the basic units not covered by any cluster in time slots 1 to $s-1$. At each time slot s, the basic units are grouped into N^{cl} clusters to be treated by the given considered maintenance intervention, where $\phi_{i,s}^{\mathrm{start}}$ and $\phi_{i,s}^{\mathrm{end}}$ denote the start and end position of the i-th cluster. Let $\Delta\phi_{\min}$ and $\Delta\phi_{\max}$ denote the minimum and maximum size of a cluster, respectively. Only the basic units inside a cluster are processed by a specific machine. Let v_{on} and v_{off} represent the speed of the machine in working mode (e.g. tamping or grinding) and non-working mode (driving), respectively. Let T_{on} and T_{off} denote the switch-on and switch-off time of the machine. The following nonlinear optimization problem is formulated to determine the clusters within the s-th time slot:

$$\max_{\{\phi_{i,s}^{\mathrm{start}}, \phi_{i,s}^{\mathrm{end}}\}_{i=1}^{N^{\mathrm{cl}}}} \sum_{i=1}^{N^{\mathrm{cl}}} \sum_{j \in \mathcal{I}_s} w_j I_{\phi_{i,s}^{\mathrm{start}} \leqslant \xi_j \leqslant \phi_{i,s}^{\mathrm{end}}} + \lambda \sum_{i=1}^{N^{\mathrm{cl}}} I_{\phi_{i,s}^{\mathrm{end}} > \overline{\xi}} \tag{30}$$

subject to

$$\phi_{1,s}^{\mathrm{start}} \geqslant \underline{\xi} \tag{31}$$

$$\phi_{N_{\mathrm{cl}},s}^{\mathrm{end}} \leqslant \xi_{\max} \tag{32}$$

$$\Delta\phi_{\min} \leqslant \phi_{i,s}^{\mathrm{end}} - \phi_{i,s}^{\mathrm{start}} \leqslant \Delta\phi_{\max} \quad \forall i \in \{1, \ldots, N_{\mathrm{cl}}\} \tag{33}$$

$$\phi_{i+1,s}^{\mathrm{start}} - \phi_{i,s}^{\mathrm{end}} \geqslant \epsilon \quad \forall i \in \{1, \ldots, N_{\mathrm{cl}} - 1\} \tag{34}$$

$$\xi_{\max} = \overline{\xi} + 2N_{\mathrm{cl}}(\Delta\phi_{\min} + \epsilon) \tag{35}$$

$$\phi_{i,s}^{\mathrm{start}} \leqslant \overline{\xi} \Longleftrightarrow \phi_{i,s}^{\mathrm{end}} \leqslant \overline{\xi} \quad \forall i \in \{1, \ldots, N_{\mathrm{cl}}\} \tag{36}$$

$$\sum_{i=1}^{N^{\mathrm{cl}}} I_{\phi_{i,s}^{\mathrm{end}} \leqslant \overline{\xi}} \cdot \left(\frac{\phi_{i,s}^{\mathrm{end}} - \phi_{i,s}^{\mathrm{start}}}{v_{\mathrm{on}}} + T_{\mathrm{on}} + T_{\mathrm{off}} \right) + \sum_{i=1}^{N^{\mathrm{cl}}-1} I_{\phi_{i,s}^{\mathrm{end}} \leqslant \overline{\xi}} \cdot \frac{\phi_{i+1,s}^{\mathrm{start}} - \phi_{i,s}^{\mathrm{end}}}{v_{\mathrm{off}}} + T_{\mathrm{set}} \leqslant T_s^{\mathrm{sl}} \tag{37}$$

The first term in the objective function (30) strives to assign the most severely-deteriorated basic units to a cluster, while the second term minimizes the total number of *active* clusters that are located within the planning range. Similar to constraints (23)–(28) of the middle-level problem, a TIO approach is used again to constraints (31)–(36). The first term in constraint (37) calculates the time needed for the machine to treat the basic units inside all active clusters, including the switching on/off time. The second time computes the time need for the machine to drive between clusters. The summation of the two parts gives the total maintenance time, which should be less than the duration T_s^{sl}. The low-level problem (30)–(37) can be transformed into an MILP problem following the procedure described in Bemporad and Morari (1999). The detailed transformation procedure is provided in Appendix C.2. The number of binary variables of the transformed MILP can be as large as $2N_{\mathrm{cl}}N^{\mathrm{bu}} + 2N_{\mathrm{cl}}$, which can be huge for a long track line with a large number of basic units. However, for a short track line (e.g. 25 km) with a moderate number of basic units (e.g. less than 500 squats) and a small number of available clusters (e.g. less than 5), the resulting MILP is still tractable.

The resulting optimal clusters might not cover all the required basic units due to lack of maintenance time. Let $\mathcal{I}_{\mathrm{rem}}$ denote the set of the indices of all the remaining squats not covered by any cluster in any active time slot. Define

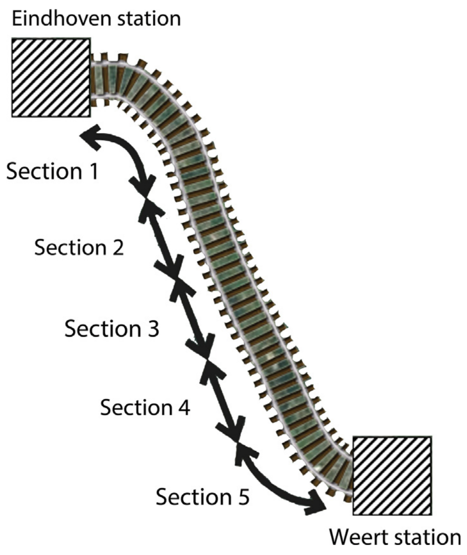


Fig. 8. Eindhoven-Weert divided into five sections.

$$v_{\mathrm{rem}} = \frac{\sum_{j \in \mathcal{I}_{\mathrm{rem}}} w_j}{\sum_{j=1}^{N^{\mathrm{bu}}} w_j} \tag{38}$$

$$\Delta\widehat{T} = \frac{|\mathcal{I}_{\mathrm{rem}}|}{N^{\mathrm{bu}}} \cdot \frac{\overline{\xi} - \underline{\xi}}{v_{\mathrm{on}}}, \tag{39}$$

where v_{rem} measures the ratio of the accumulated severities of the remaining basic units over the total severities of all the basic units, while $\Delta\widehat{T}$ is an estimate of the additional maintenance time to cover the remaining basic units. The values of v_{rem} and $\Delta\widehat{T}$ determine whether to leave the remaining basic units uncovered, or treat them with additional maintenance time, as stated at the beginning of Section 4.

5. Case study

5.1. Settings

A case study on the optimal treatment of squats is performed for the Eindhoven-Weert line in the Dutch railway network. This line is approximately 25 km long, which is divided into five sections of equal length, as shown in Fig. 8. The rail of the Eindhoven-Weert line is considered as the infrastructure in this case study, and the five sections of rail are treated as components with independent deterioration dynamics. The basic units are the 454 individual squats located on the entire rail. A squat is a typical rail contact fatigue, and its evolution depends on the dynamic contact between wheels and rails. Squats are classified into different categories depending on their visual length,[7] which can be detected automatically using techniques like axle box acceleration (ABA) systems (Li et al., 2015; Molodova et al., 2014), eddy current testing (Song et al., 2011), and ultrasonic surface wave (Fan et al., 2007). In this paper, squats with a visual length below 30 mm are considered as light squats, in which cracks have not appeared yet. Squats with a visual length ranging from 30 to 50 mm are considered to be at the medium stage of growth. The medium squats evolve into severe squats when the network of cracks spreads further. Squats with a visual length over 50 mm are considered as severe squats. They should be treated as early as possible because they might lead to hazards like derailment (see Fig. 9).

The deterioration model of one individual squat is given in Appendix A. We call this model the *simulation model*, which is a piecewise-affine function fitted with historical data in the Eindhoven-Weert line. The length of each individual squat is updated by the simulation model at each time step. The condition of one section is defined as the average length of all the squats within the section. The condition of each section is updated by aggregating the simulated squat lengths using the simulation model for each individual squat. The dynamics of the condition of one section is described by the *prediction model*, which can be obtained by piecewise-affine identification using simulated data obtained from the simulation model.

[7] The length noticeable at the surface of the rail.

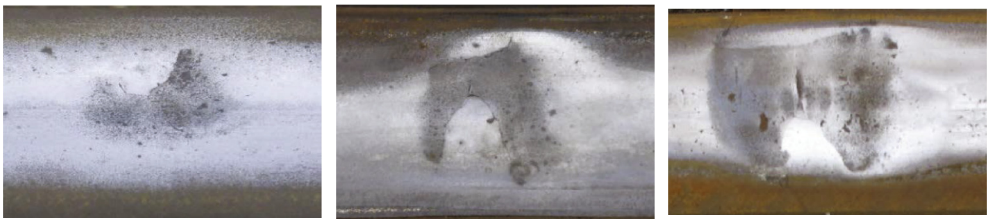

(a) Light squat. (b) Medium squat. (c) Severe squat.

Fig. 9. Squats with different severities.

The prediction model is used for long-term maintenance planning at the high level. The set $\mathcal{U} = \{0, 1, 2\}$ contains all possible maintenance actions that can be applied to one section, with 0, 1, 2 representing "performing no maintenance", "grinding", and "replacing", respectively. Note that these maintenance actions are applied to each squat in the corresponding section. The condition $x_{\text{con},j}$ is defined as the average length of squats in section j, while the auxiliary variable $x_{\text{aux},j}$ records the number of previous grinding operations on section j since the last replacement. Grinding cannot be applied an unlimited number of times, as it tries to remove a squat by reducing the thickness of the rail. Three realizations of uncertainties are considered, which are collected in the set $\Theta_j = \{1, 2, 3\}$, with 1, 2, 3 representing fast, average, and slow deterioration. Following the notation of the generic deterioration model described in Section 3, the dynamics of the condition of section j can be described by the following scenario-based model:

$$\begin{aligned} x_j^{\text{con}}(k+1) &= f^{\text{con}}(x_j^{\text{con}}(k), u_j(k), \theta_j(k)) \\ &= \begin{cases} f_{\text{Deg}}(x_j^{\text{con}}(k), \theta_j(k)) & \text{if } u_j(k) = 0 \text{ (no maintenance)} \\ f_{\text{Gr}}(x_j^{\text{con}}(k), \theta_j(k)) & \text{if } u_j(k) = 1 \text{ (grinding)} \\ 0 & \text{if } u_j(k) = 2 \text{ (replacing)} \end{cases} \\ &\forall j \in \{1, \ldots, n\}. \end{aligned} \tag{40}$$

The natural degradation of one section is described by function f_{Deg}, which is a piecewise-affine function in the form

$$f_{\text{Deg}}(x_j^{\text{con}}, \theta_j) = a_{q,\theta_j} x_j^{\text{con}} + b_{q,\theta_j} \quad \text{if} x_j^{\text{con}} \in \mathcal{X}_{j,q}^{\text{con}} \subset \mathcal{X}_j^{\text{con}}, \tag{41}$$

where the condition space of section j is partitioned $\{\mathcal{X}_{j,q}\}_{q=1}^{3}$. As discussed in Jamshidi et al. (2017), light (early-stage), medium (middle-stage), and severe (late-stage) squats exhibit different deterioration dynamics, and a piecewise-affine function is able to capture the deterioration dynamics of the squats.

Function f_{Gr} captures the effect of grinding, which becomes less effective when the condition deteriorates more severely. It is also a piecewise-affine function of the form

$$f_{\text{Gr}}(x_j, \theta_j) = \begin{cases} 0 & \text{if } x_j \leqslant x_{\theta_j}^{\text{eff}} \\ \psi_{\theta_j}(x_j - x_{\theta_j}^{\text{eff}}) & \text{if } x_j > x_{\theta_j}^{\text{eff}} \end{cases} \tag{42}$$

where $x_{\theta_j}^{\text{eff}}$ represents the effective condition for grinding if θ_j is realized.

The sampling time T in this case study is one month. The prediction horizon N_{P} and control horizon N_{C} are both 6 months. The predicted condition of each section must be kept below a maintenance limit within the prediction window at every time step, which can be explained by the following constraint:

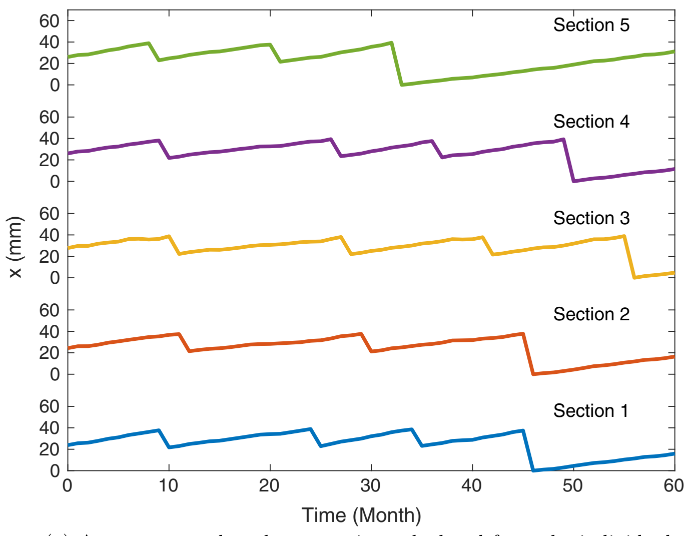
$$x_j^{\text{con}}(k+l) \leqslant x_{\max} \quad \forall j \in \{1, \ldots, n\}, \forall l \in \{1, \ldots, N_{\text{P}}\}. \tag{43}$$

The dynamics of the auxiliary variable x_j^{aux}, which is a counter for grinding, can be formally expressed as:

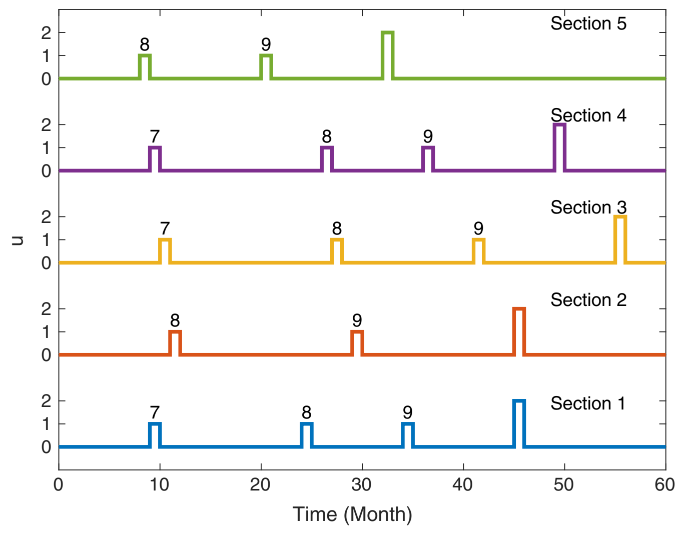
$$\begin{aligned} x_j^{\text{aux}}(k+1) &= f^{\text{aux}}(x_j^{\text{aux}}(k), u_j(k)) \\ &= \begin{cases} x_j^{\text{aux}} & \text{if } u_j(k) = 0 \text{(no maintenance)} \\ x_j^{\text{aux}} + 1 & \text{if } u_j(k) = 1 \text{(grinding)} \\ 0 & \text{if } u_j(k) = 2 \text{ (replacing)} \end{cases} \\ &\forall j \in \{1, \ldots, n\}. \end{aligned} \tag{44}$$

The number of grinding operations cannot exceed a maximum number $N_{\max}^{\text{Gr}}$ within the prediction window, thus we have:

$$x_j^{\text{aux}}(k+l) \leqslant N_{\max}^{\text{Gr}} \quad \forall j \in \{1, \ldots, n\}, \forall l \in \{1, \ldots, N_{\text{P}}\}. \tag{45}$$



(a) Average squat length per section calculated from the individual squat dynamics.



(b) Interventions suggested by the high-level MPC controller. The number above each grinding intervention indicates the number of previous grinding operations applied to the section since the last replacement.

Fig. 10. Simulated average squat length and interventions suggested by the high-level MPC controller for each of the five sections of the entire track.

In summary, Eqs. (40) and (44), together with constraints (43) and (45) form the stochastic deterioration model for this case study. A scenario-based approach is applied to this stochastic model. Three representative scenarios are selected from the set $\tilde{\Theta}$ containing all possible realizations of uncertainties within the prediction period. They are fast, average, slow deterioration for all sections at every time step within the prediction period.

The initial conditions and parameters for the prediction model (40) and the high-level objective function (17) are provided in Appendix B. The parameters for the middle-level and low-level optimization problem are provided in Appendices B.2 and B.3, respectively. For illustration purposes, both the middle-level and low-level problems are only triggered for grinding.

The multi-level approach is implemented in Matlab R2016b, on a desktop computer with an Intel Xeon E5-1620 eight-core CPU and 64 GB of RAM, running a 64-bit version of SUSE Linux Enterprise Desktop 12. The nonlinear optimization problem at each time step of the high-level MPC controller is solved using the function *patternsearch* of the Matlab Global

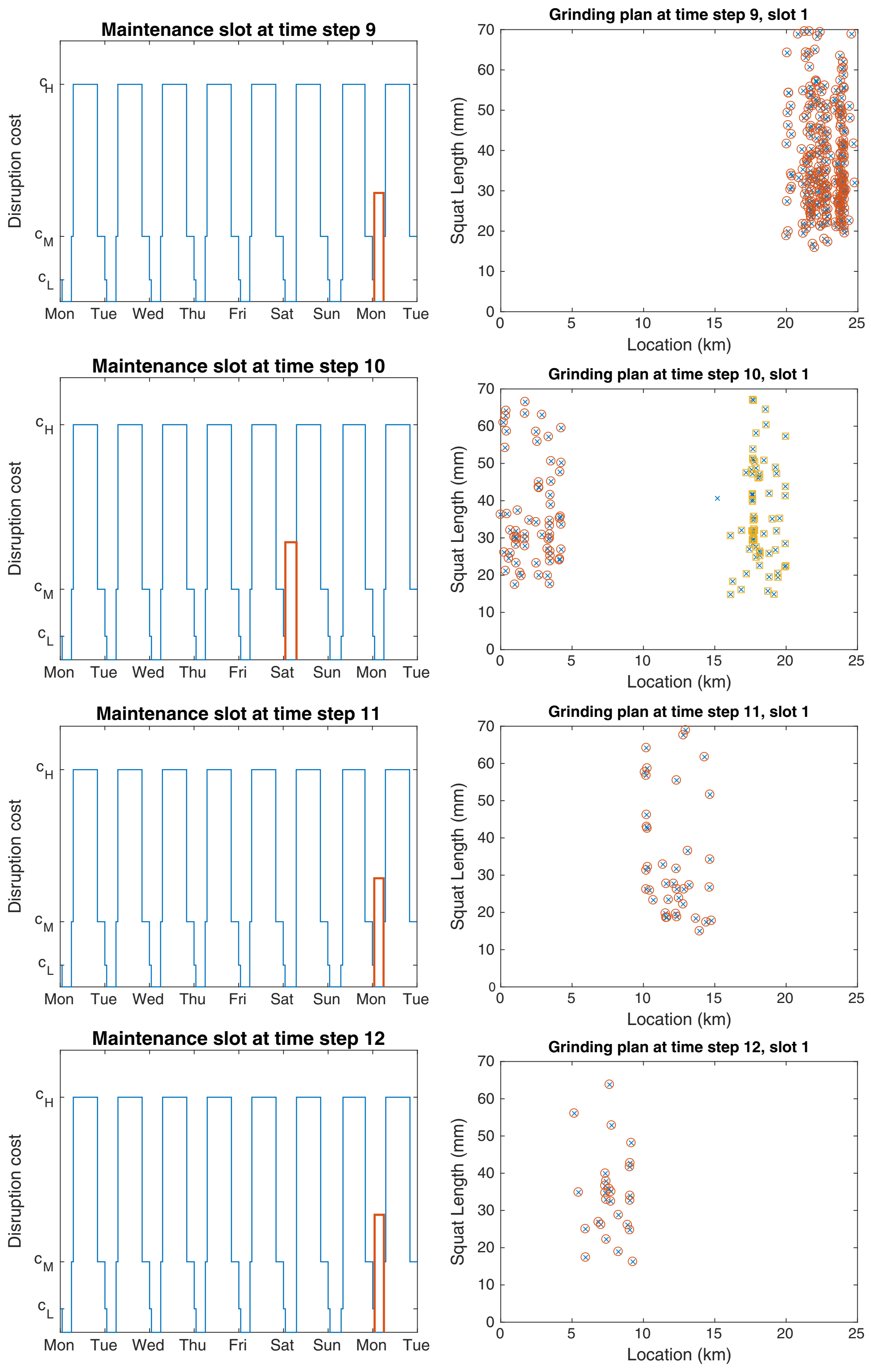


Fig. 11a. Results of the middle-level problem and the low-level problem at time step 9, 10, 11, and 12 of the high level. A squat (represented by a dot) is in a cluster if it is covered by a circle (first cluster) or square (second cluster).

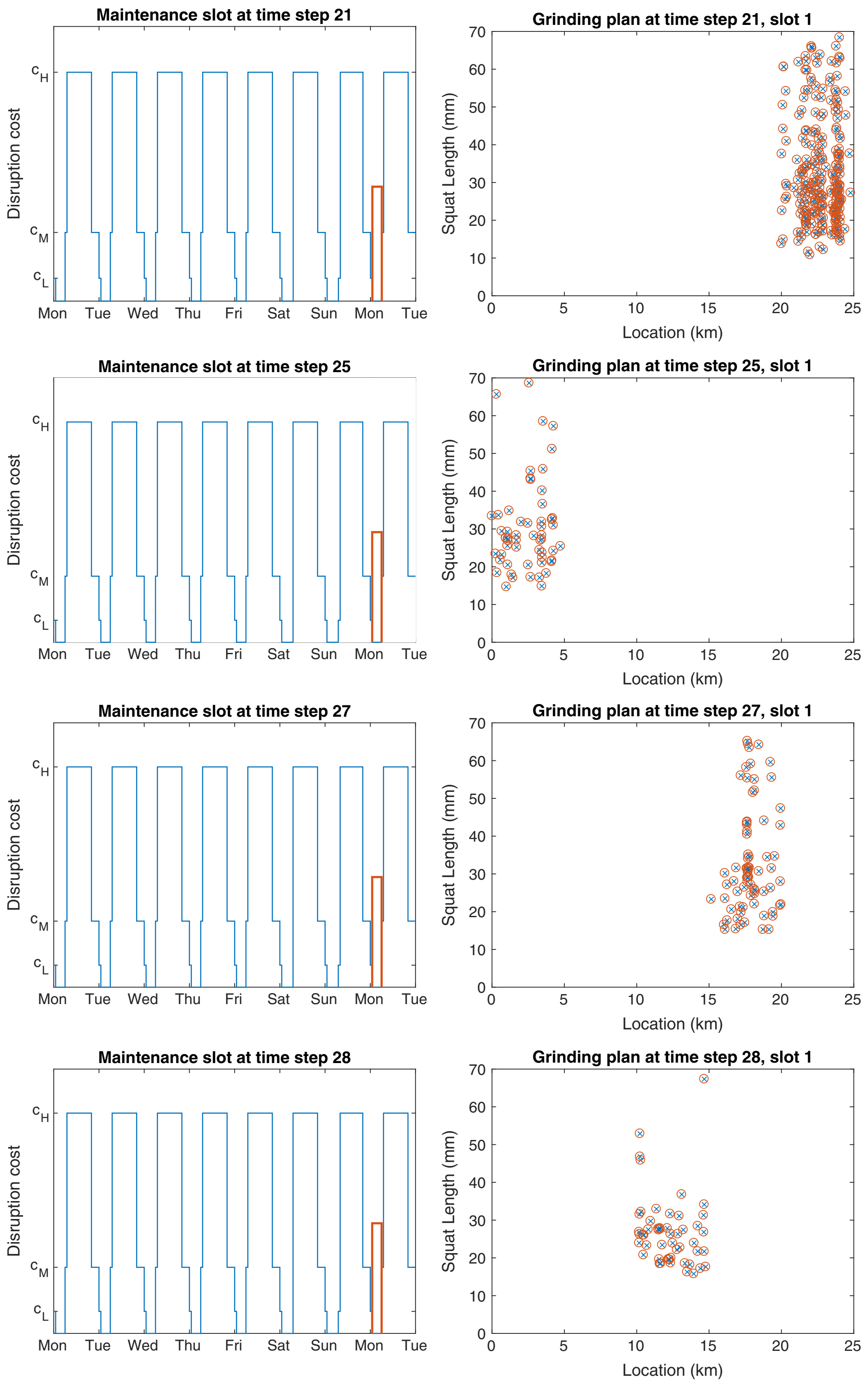


Fig. 11b. Results of the middle-level problem and the low-level problem at time step 21, 25, 27, and 28 of the high level. A squat (represented by a dot) is in a cluster if it is covered by a circle (first cluster) or square (second cluster).

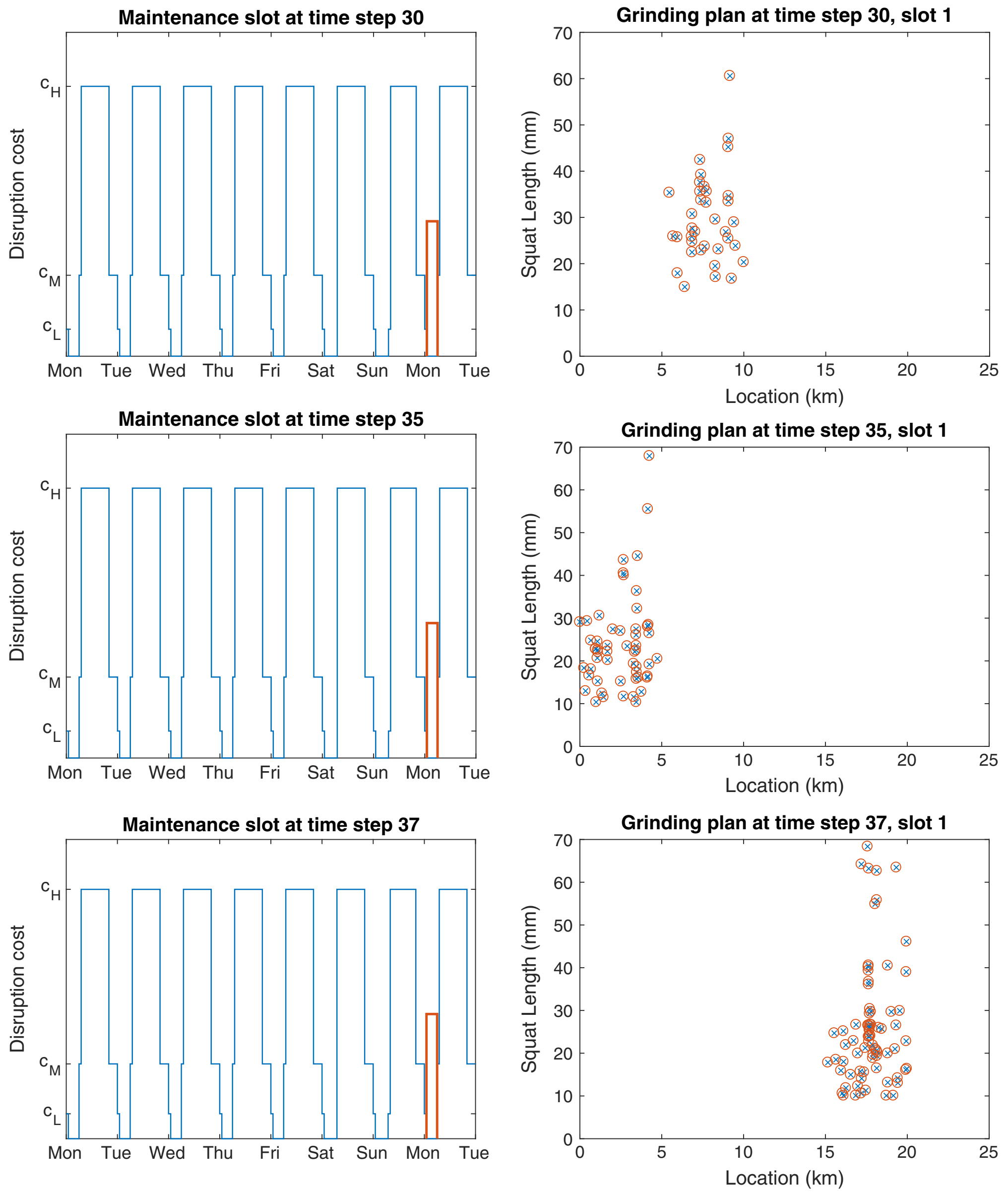


Fig. 11c. Results of the middle-level problem and the low-level problem at time step 30, 35, and 37 of the high level. A squat (represented by a dot) is in a cluster if it is covered by a circle (first cluster) and square (second cluster).

Optimization Toolbox, with 100 random starting points. CPLEX 12.5 (called via Tomlab 8.0) is used as the MILP solver for the middle-level and low-level problems.

5.2. Discussions of results

The multi-level, scenario-based, chance-constrained approach is demonstrated by a representative run with a five-year planning horizon. The optimal maintenance interventions suggested by the high-level MPC controller are shown in Fig. 10b, and the simulated condition of each section is shown in Fig. 10a. Note that the state at each time step is computed from the individual squat lengths simulated by the simulation model, and only the cluster-wise grinding plan from the low level is applied to the simulation model. Grinding is suggested when the condition is near its maintenance limit (40 mm), and the interval between two consecutive grinding turns out to be between 10 to 18 months for a section. It is interesting to notice that the interval between two consecutive grindings becomes shorter over time, as shown in the resulting

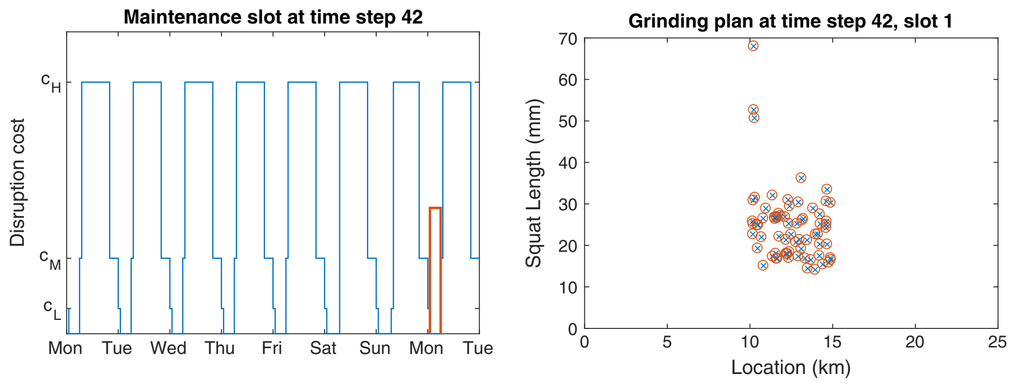


Fig. 11d. Results of the middle-level problem and the low-level problem at time step 42 of the high level. A squat (represented by a dot) is in a cluster if it is covered by a circle (first cluster) and square (second cluster).

intervention plan of Section 1, 3, and 4 in Fig. 10b. When the maximum number of grinding (10 in the case study) is reached for any section, replacing is suggested. The mean and maximum CPU time to solve the MPC optimization problem at each time step is 47s and 68s, respectively, which is much faster than the sampling time (one month). Moreover, both the middle-level and the low-level problem can be solved to global optimality within 10 s. Thus we can claim that the proposed multi-level MPC approach is implementable in real-time.

The results of the middle-level and low-level problem are shown in Fig. 11. The middle-level and low-level problems are triggered whenever grinding is suggested for any of the five sections. As shown in Table 5, a 5-h traffic free time slot (1:00–6:00) is available for workdays, and a 6-h traffic free time slot (1:00–7:00) is available for weekends. As shown in Fig. 11, only one section is to be ground at time step 9, 11, 12, 21, 25, 27, 28, 30, 35, 37, 42, according to the high-level MPC controller, and all the squats inside the single section can be covered in one cluster, using a 5-h short time slot on a weekday (e.g. Monday). Two non-consecutive sections are suggested at time step 10 at the high level, and even with a 6-h long time slot on weekends (e.g. Sunday), there is still one squat not covered by the resulting two clusters. This is because the benefit of covering this single medium-stage squat is less than the cost associated with the additional maintenance time to cover it.

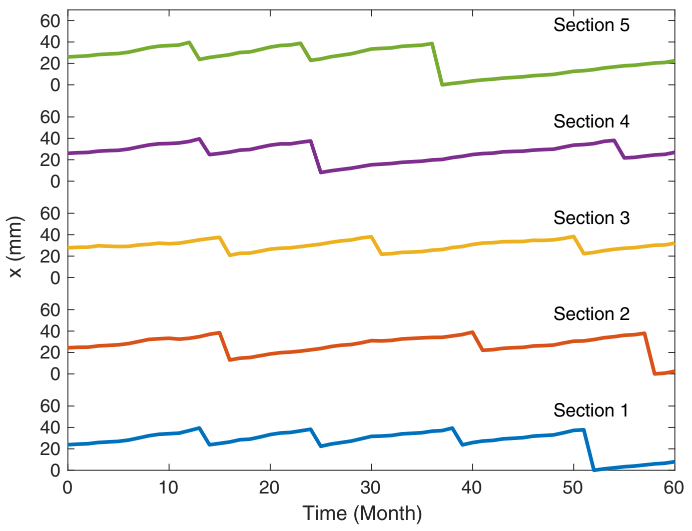
5.3. Supplementary run with modified parameters

In addition to the representative run with realistic parameters, another supplementary run with a lower setup cost of one maintenance time slot and a lower grinding speed is performed to demonstrate the how the iterative procedure of the middle-level and low-level problem works in more “difficult” situations. The results of the high-level MPC controller is given in Fig. 12, and the resulting time slots and clustering plans are provided in Figs. 13 and 14, respectively.

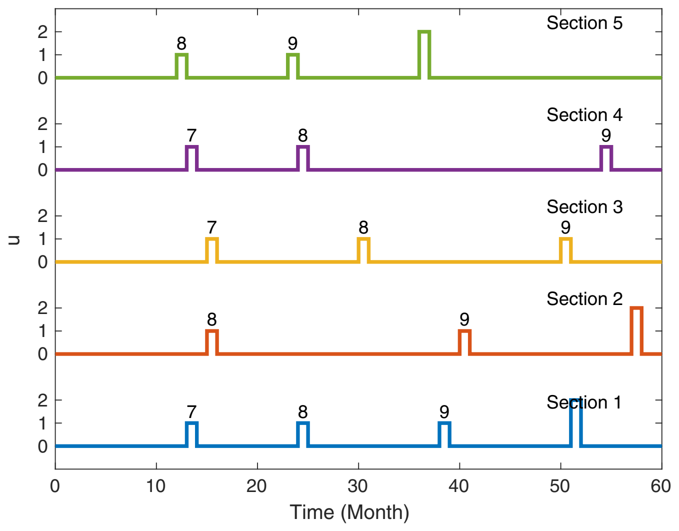
As shown in Fig. 12, at time step 13, 31, 51, 55, only one section are suggested to be ground by the high-level controller. In this case, the short traffic-free time slot on a weekday (Monday) is sufficient to cover all the squats in one cluster, as shown in Figs. 13 and 14. Two non-consecutive sections are suggested at time step 14, as shown in Fig. 12, resulting in a 6.8-h maintenance time slot (see Fig. Fig. 13), which incurs cost of disruption to train traffic. There are still 6 uncovered light and medium squats, as shown in the clustering plan in Fig. 14. Section 2 and 3 are to be ground at time step 16, as shown in Fig. 12. Because the setup cost of one maintenance time slot is much lower in the supplementary run (10% of the value in the representative run), two different time slots (on Saturday and Modnay) are suggested at the middle level, without incurring any disruption cost. Two clusters are used to cover the most severe squats for the first long time slot on Saturday, while another cluster is used in the second short time slot on Monday to cover all the remaining squats. Similarly, at time step 25, two non-consecutive sections are to be ground, according to the high-level intervention plan in Fig. 12. The two long traffic-free time slots on weekends are used to treat the squats in the two sections. Two clusters are used in the Saturday maintenance slot to cover the most severe squats in Section 1 and 4, and three clusters are suggested in the Sunday maintenance slot to cover the remaining squats.

5.4. Comparison with alternative approaches

Three alternative approaches for maintenance planning: the nominal MPC, the cyclic approach, and the approach currently used in practice, are implemented for comparison with the proposed approach. The nominal MPC can be viewed as the deterministic counterpart of the scenario-based chance-constrained MPC, as it considers only the average deterioration dynamics in the optimization problem at each time step. The cyclic approach is a preventive maintenance strategy that performs grinding and replacement at regular intervals. The formulation of the offline optimization problem to obtain the



(a) Average squat length per section calculated from the individual squat dynamics.



(b) Interventions suggested by the high-level MPC controller. The number above each grinding intervention indicates the number of previous grinding operations applied to the section since the last replacement.

Fig. 12. Results of the high-level controller with reduced setup cost at the middle level and reduced grinding speed at the low level.

optimal intervals for grinding and replacement for the cyclic approach is presented in Appendix D. We also implement the approach currently used in the Netherlands. We refer to this approach as "current approach", which is a cyclic preventive maintenance approach that grinds the entire line every six months.

The four maintenance approaches are applied to ten test runs of a five-year planning horizon with different pre-defined sequences of realizations of uncertainties provided in Appendix B.1. We compare the constraint violation, the value of the closed-loop objective function for the three approaches, and CPU time of the two MPC controllers. The constraint violation is measured by:

$$v = \max_{\substack{j=1,\ldots,n \\ k=1,\ldots,k_{\text{end}}}} \left\{ \frac{x_j^{\text{con}}(k) - x_{\max}^{\text{con}}}{\bar{x}^{\text{con}}} \right\} \tag{46}$$

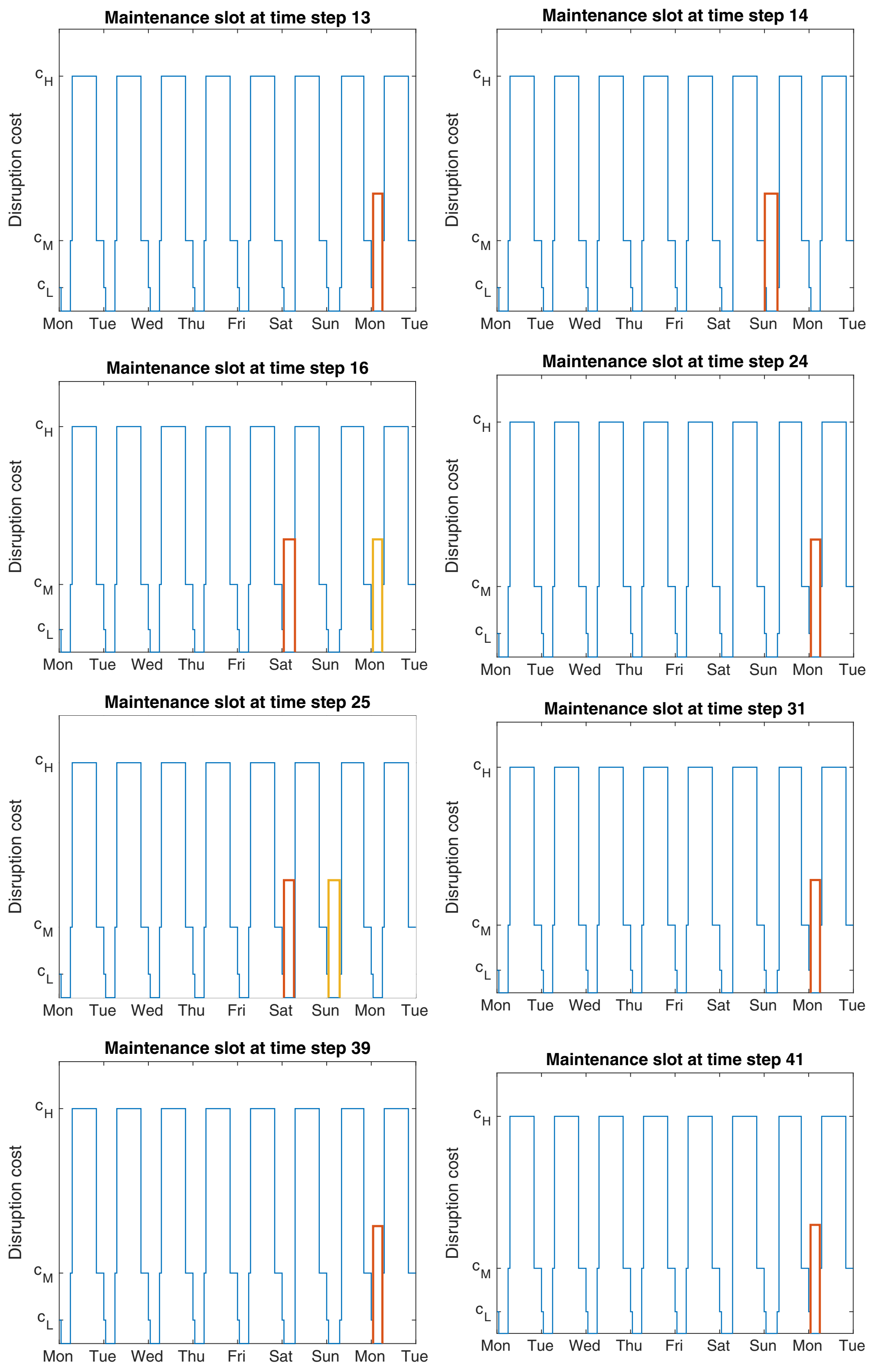


Fig. 13a. Results of the middle-level problem, when the setup cost of one maintenance slot is reduced.

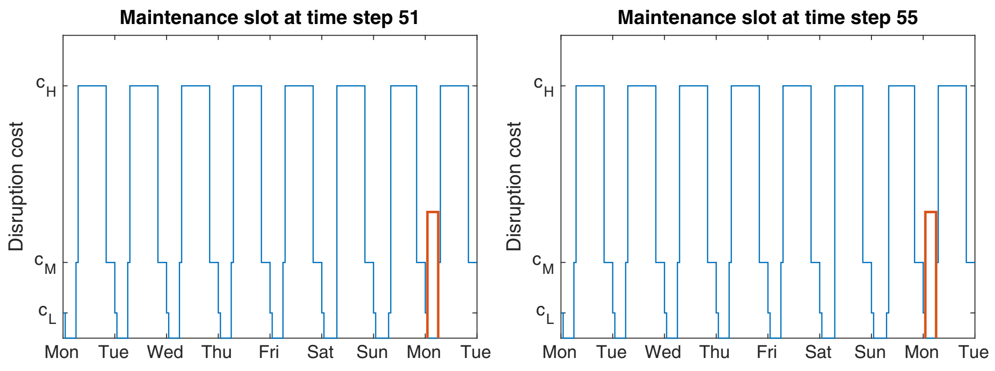


Fig. 13b. Results of the middle-level slot allocation problem, when the setup cost of one maintenance slot is reduced.

where $x^{\text{con}}_{\max} = 40$ mm is the maintenance limit, and $\bar{x}^{\text{con}} = 70$ mm is the range of the condition. Let $v_{\text{Nom}}, v_{\text{CC}}, v_{\text{Cyc}}$ and v_{cur} denote the maximum constraint violation of the nominal MPC controller, the scenario-based chance-constrained MPC controller, the cyclic approach, and the current approach, respectively. The maximum constraint violation measures the robustness of each approach. Similarly, let $J_{\text{Nom}}, J_{\text{CC}}, J_{\text{Cyc}}$, and J_{cur} denote the closed-loop objective function value[8] for the nominal MPC controller, the scenario-based chance-constrained MPC controller, the cyclic approach, and the current approach, respectively. The closed-loop objective function value measures the cost-efficiency of each approach. A lower value indicates higher cost-efficiency. Moreover, let T_{Nom} and T_{CC} denote the CPU time[9] for the nominal and chance-constrained MPC controller, respectively. As our goal is a safe but non-conservative maintenance strategy that is tractable, robustness and cost-efficiency are more important evaluation criteria than CPU time. The performance and computational effort of the four maintenance approaches are compared in Table 1.

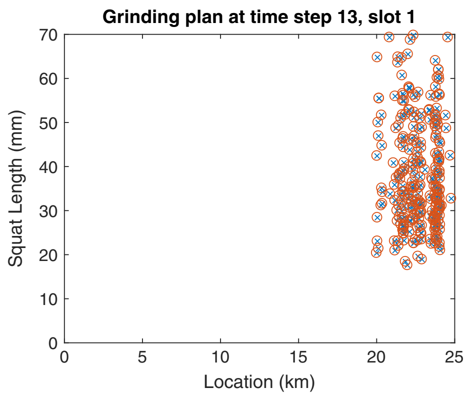
The current approach, i.e. grinding every six months, is the most conservative approach. Although it has no constraint violation for the ten test runs, the closed-loop objective function value is almost three time as much as that of the nominal MPC approach. The cyclic approach, which can be viewed as an improvement on the over-conservative current approach by optimizing the intervals for grinding and replacing, is more cost-efficient than the current approach, as its closed-loop objective function value is only slightly higher than that of the two MPC approaches, except in Run 4. The cyclic approach is not robust, as it has constraint violations in seven out of the ten test runs. The advantage of cyclic approach is that it is less computational demanding than MPC approaches, as only one optimization problem needs to be solved offline.

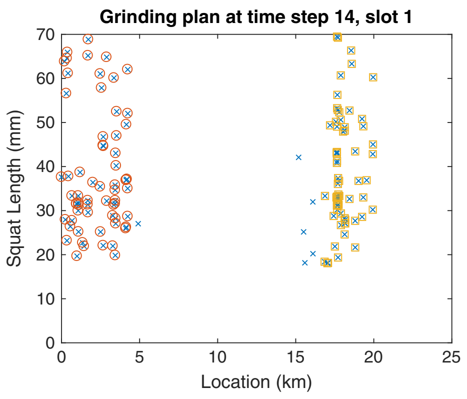
Theoretically, both MPC controllers can have constraint violations in the whole planning period. Although more robust than the nominal MPC, scenario-based chance-constrained MPC only guarantees that the constraints are satisfied with a possibility higher than a given confidence level (90% in this case study). But there is no constraint violation for the scenario-based chance-constrained MPC in the ten runs of simulation, as shown in Table 1, while the constraints are violated for nominal MPC in seven out of the ten runs. The largest constraint violation for the nominal MPC approach is 0.78% for Run 3, which might lead to hazards like derailment. The closed-loop objective function values of the two MPC approaches are almost equal in every test run, and the two MPC approaches are the most cost-efficient among the four approaches. It is interesting to notice that the closed-loop objective function values of the two MPC approaches in Run 4 are only half of those of the other runs, showing a significant advantage over the cyclic approach and the current approach in terms of cost-efficiency. This is because MPC is a flexible real-time decision making scheme, that can adapt the intervention plan to the actual (or estimated) condition of the infrastructure, while the cyclic and the current approaches are both offline schemes that do not take the actual (or estimated) condition into consideration.

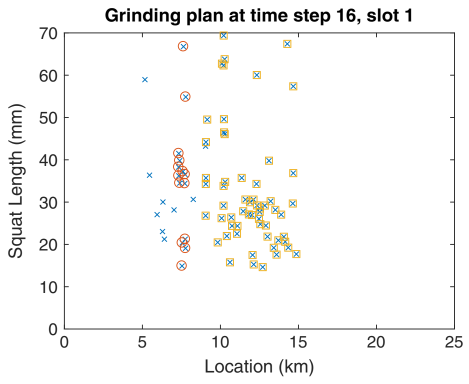
The greatest advantage of the nominal MPC approach over the chance-constrained MPC approach is that it requires less than half of the CPU time of the chance-constrained approach. Despite being the most computationally demanding approach, from Table 1 we can still conclude that the chance-constrained MPC approach is the most promising one among the four alternative approaches, as robustness and cost-efficiency are more important evaluation criteria than computational efforts, since the time to compute an optimal maintenance plan is abundant for infrastructures with a slow deterioration process. Indeed, despite being the slowest one among the four approaches, the chance-constrained MPC approach is not only tractable but also real-time implementable, as the sampling time in the case study is one month, and the optimization problem at each time step of the chance-constrained MPC approach takes approximately one minute to solve.

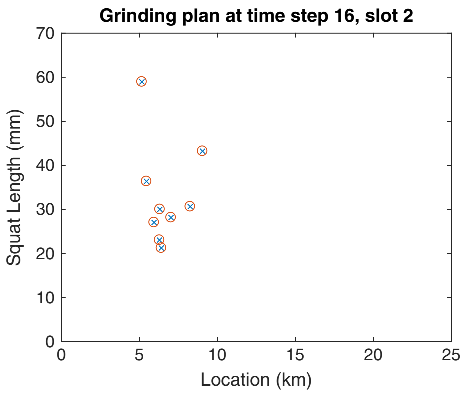
[8] This is obtained by evaluating (17) over the entire planning horizon.

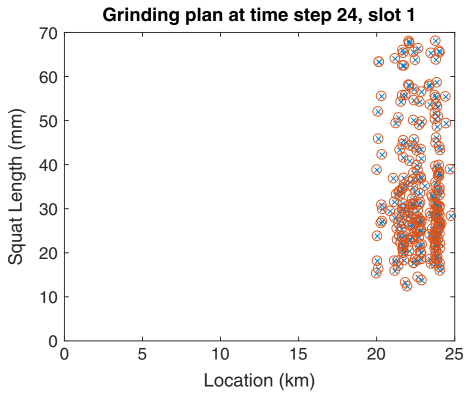
[9] We measure only the CPU time to solve all the optimization problems within the planning horizon.

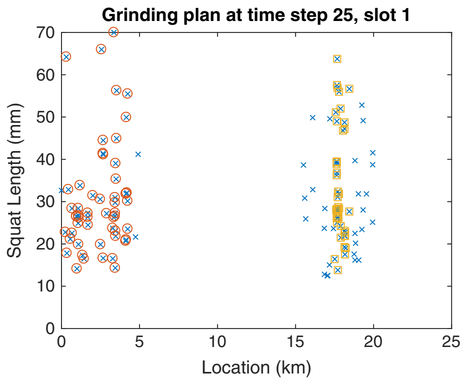


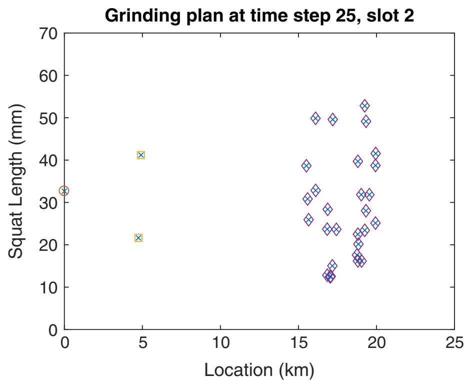












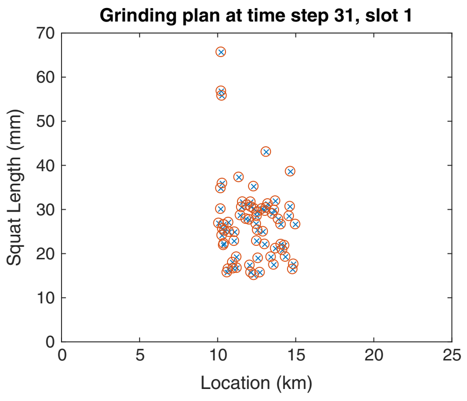


Fig. 14a. Results of the low-level clustering problem, in which the grinding speed has been reduced.

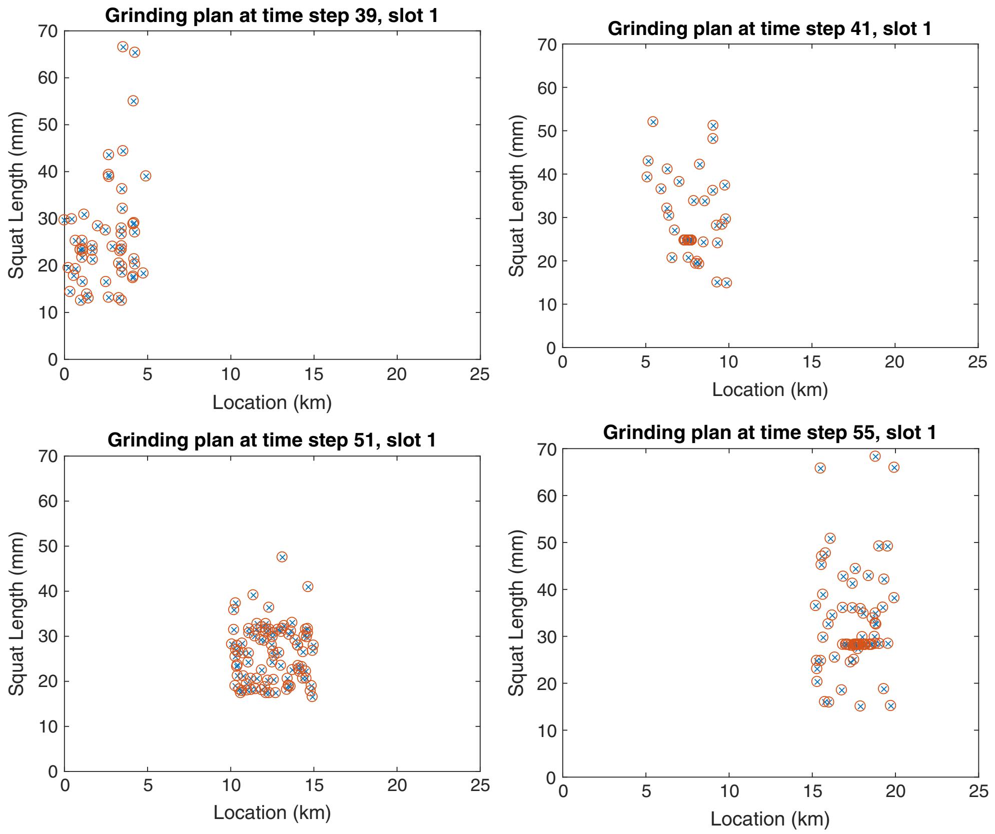


Fig. 14b. Results of the low-level clustering problem, in which the grinding speed has been reduced.

Table 1
A comparision between the nominal MPC controller, scenario-based chance-constrained MPC controller, the cyclic approach, and current strategy for ten runs with a pre-defined sequence of uncertainties. The CPU time to solve the offline optimization problem for the cyclic approach is 7.5 mininutes.

Run	Constraint violation				Closed-loop performance				CPU time (h)	
	v_{Nom} (%)	v_{CC}	v_{Cyc} (%)	v_{cur}	$\frac{J_{\text{Nom}}}{J_{\text{cur}}}$ (%)	$\frac{J_{\text{CC}}}{J_{\text{cur}}}$ (%)	$\frac{J_{\text{Cyc}}}{J_{\text{cur}}}$ (%)	J_{cur}	T_{Nom}	T_{CC}
1	0.62	0	0.26	0	35.37	35.33	36.12	344,400	0.39	0.87
2	0	0	0	0	35.53	35.45	36.00	344,320	0.40	0.89
3	0.78	0	0.08	0	35.39	35.10	36.12	344,420	0.38	0.89
4	0.27	0	0	0	17.46	23.17	35.86	344,260	0.37	0.85
5	0.51	0	0	0	35.48	35.39	36.05	344,380	0.38	0.88
6	0	0	0.61	0	35.38	35.33	36.11	344,370	0.40	0.90
7	0.69	0	0.69	0	35.37	35.33	36.107	344,370	0.40	0.88
8	0	0	0.55	0	35.12	35.08	36.10	344,390	0.38	0.89
9	0.68	0	0.68	0	35.17	35.33	36.11	344,360	0.39	0.88
10	0.51	0	1.47	0	35.34	35.07	36.19	344,390	0.42	0.89

Table 2
Evolution of squats of different categories and with different uncertain growth rate. The length that a squat has evolved into after one month is calculated from its current length L.

Realization of uncertainties	Light squat ($L < 30$)	Medium squat ($30 \leqslant L \leqslant 50$)	Severe squat ($L > 50$)
Fast growth	$1.016 \cdot L + 1.2809$	$1.1029 \cdot L - 1.6725$	$L + 4.5$
Average growth	$1.0017 \cdot L + 0.9959$	$1.0699 \cdot L - 1.2165$	$1.0008 \cdot L + 2.6694$
Slow growth	$0.9915 \cdot L + 0.681$	$1.016 \cdot L - 0.0801$	$0.9949 \cdot L + 1.0127$

Table 3
Effect of grinding for squats of different categories and with different uncertain growth rate. The length of a squat after grindng can be calculted from its current length L.

Realization of uncertainties	$L \leqslant 16$	$L > 16$
Fast	0	$1.0028 \cdot (L - 16)$
Average	0	$1.0009 \cdot (L - 16)$
Slow	0	$0.9985 \cdot (L - 16)$

6. Conclusions and future work

A multi-level approach for the optimal planning of maintenance interventions for railway infrastructures has been developed in this paper. A scenario-based, chance-constrained TIO-MPC controller is implemented at the high level for long-term, component-wise, condition-based planning of maintenance interventions. Both the middle-level and low-level problems are triggered whenever an intervention is suggested for any component by the high-level controller. When triggered, the middle-level problem allocates the time slots for the corresponding interventions by optimizing the trade-off between total setup costs of maintenance operations and the disruption to the train traffic. The low-level problem then groups the basic units into clusters to execute the maintenance interventions suggested at the high level. This cluster-wise work plan must be performed within the time slot determined at the middle level. A case study on the optimal treatment of squats in the Eindhoven-Weert line in the Dutch railway network is performed. The simulation results of a representative run with a five-year planning horizon show that the proposed multi-level approach is real-time implementable and provides a suitable maintenance plan. A comparison with the nominal approach further demonstrates the advantage of the proposed chance-constrained approach in keeping the condition below the maintenance limit.

In the future, distributed MPC schemes can be considered for maintenance planning of railway infrastructures of larger scales. Moreover, a more optimal solution can be obtained by combining the middle-level slot allocation and the low-level clustering problem into one single optimization problem. This is tractable for small problems like the case study in Section 5. However, real-world problems might involve a much longer track line (e.g. over 250 km) with a large number of basic units (e.g. more than 4500 squats), and in this case the combined approach might become intractable. In order to address such cases, decomposition method seems to be a viable solution approach; this will be a topic of future research. Other interesting practical issues, like speed limitations caused by maintenance, can also be considered. Furthermore, maintenance planning for railway infrastructures containing heterogeneous components, e.g. rail and switches, could also be considered.

Acknowledgement

Research sponsored by the NWO/ProRail project "Multi-party risk management and key performance indicator design at the whole system level (PYRAMIDS)", project 438-12-300, which is partly financed by the Netherlands Organisation for Scientific Research (NWO).

Appendix A. Simulation model

The simulation model describes the dynamics of one individual squat. The model has the same form as the prediction model (40), but with different parameters. Three realizations of the uncertainties are considered. They are fast, average, and slow growth, with a probability of occurrence of 0.3, 0.4, and 0.3, respectively. Let L denote the length of an individual squat; the natural evolution (without any maintenance interventions) is given in Table 2, which calculates the length of the squat after one month.

A squat can be treated effectively by grinding only when its length is less than the effective grinding length (16 mm). The length a squat can be reduced to by grinding is calculated using the formulas of Table 3.

We also consider the occurrence of new squats. At each time step (month), the number of new squats in the entire track is the rounded value of a random variable normally distributed with mean 3 and variance 1. The probability distribution of the location of a new squat is also a normal distribution with mean 12.5 km and standard deviation 4 km. New squats should always be early-stage squats when they first appear on the rail, and the initial length of a new squat follows a normal distribution with mean 15 mm and standard deviation 5 mm.

Appendix B. Parameters for case study

B.1. High-level prediction model

The prediction model describes the dynamics of each of the five section, which deteriorate independently from each other. The same prediction model with the same parameters are used for each section. The condition space $\mathcal{X}_j^{\text{con}} = [0, 70]$ for section j is partitioned into the following three intervals:

Table 4
Sequences of the realizations of uncertainties, where 1, 2, 3 stands for fast, average, slow growth for every squat, respectively.

Run	Sequence of the realizations of uncertainties
1	1, 1, 1, 2, 2, 2, 2, 3, 3, 3
2	1, 2, 2, 2, 2, 2, 2, 3, 3, 3
3	1, 1, 1, 1, 2, 2, 3, 3, 3, 3
4	1, 1, 2, 2, 3, 3, 3, 3, 3, 3
5	1, 1, 1, 2, 2, 2, 3, 3, 3, 3
6	3, 1, 1, 3, 2, 3, 2, 1, 2, 2
7	2, 1, 3, 3, 2, 2, 1, 3, 2, 1
8	1, 2, 3, 1, 3, 2, 2, 1, 3, 2
9	1, 3, 2, 3, 2, 1, 2, 2, 3, 1
10	1, 3, 2, 1, 3, 1, 2, 2, 1, 3

$$\mathcal{X}^{\text{con}}_{j,1} = [0, 30), \quad \mathcal{X}^{\text{con}}_{j,2} = [30, 50), \quad \mathcal{X}^{\text{con}}_{j,3} = [50, 70).$$

The parameters for the piecewise-affine degradation function (41) are collected in the following two matrices:

$$A_j = (a_{q,\theta_j}) = \begin{bmatrix} 1.0037 & 1.0073 & 1.0120 \\ 1.0017 & 1.0053 & 1.0075 \\ 0.9992 & 1.0007 & 1.0008 \end{bmatrix}, \quad B_j = (b_{q,\theta_j}) = \begin{bmatrix} 0.1954 & 0.1484 & 0 \\ 0.1438 & 0.0732 & 0 \\ 0.1041 & 0.0701 & 0.0743 \end{bmatrix} \quad \forall j \in \{1, \ldots, n\}.$$

The parameters for the grinding model (42) are collected in the two matrices

$$\Psi_j = (\psi_{\theta_j}) = \begin{bmatrix} 0.9996 \\ 0.9996 \\ 0.9995 \end{bmatrix}, \quad X^{\text{eff}}_j = (x^{\text{eff}}_{\theta_j}) = \begin{bmatrix} 11.8275 \\ 11.9586 \\ 11.9336 \end{bmatrix} \quad \forall j \in \{1, \ldots, n\}.$$

The initial condition $x(0) = [(x^{\text{con}}(0))^{\text{T}}\,(x^{\text{aux}}(0))^{\text{T}}]^{\text{T}}$ is given by

$$x^{\text{con}}(0) = \begin{bmatrix} 23.8757 \\ 24.356 \\ 27.7457 \\ 26.0526 \\ 26.0487 \end{bmatrix}, \quad x^{\text{aux}}(0) = \begin{bmatrix} 7 \\ 8 \\ 7 \\ 7 \\ 8 \end{bmatrix}.$$

The operational limit $x_{\max}$ is 40 mm, and a maximum of 10 grinding operations since the last replacement is allowed for each section, i.e. $N^{\text{Gr}}_{\max} = 10$.

The trade-off between the condition and the cost is $\lambda = 10$ in the high-level objective function (17), and the scaling factor μ is 70. The 1-norm is taken for J_{Deg} in (14), and replacement is 30 times as expensive as grinding, i.e. $\gamma_{j,1} = 1$ and $\gamma_{j,2} = 30$ in (15). The maintenance limit is $x^{\text{con}}_{\max} = 40$ mm.

Ten sequences of the realization of uncertainties in the five-year planning period are given here. Each sequence specifies the realization of uncertainties for each individual squat at every time step in the planning period for one particular run. For easy reproduction of our results, we assign a uniform realization of uncertainties for all the squats at every time step, i.e. the realization of uncertainty is the same for all the squat at one time step. Moreover, we make each of the ten 60-step sequences a six-times repetition of the 10-step sequences summarized in Table 4.

B.2. Middle-level problem

The hourly cost of traffic disruption is determined by the number of passenger trains[10] from Eindhoven to Weert every hour. Only regular schedules (excluding special schedules for holidays) are considered. The train schedule is periodic with a period of one week. Moreover, all workdays (Monday-Fridy) have the same schedule, while Saturdays and Sundays have different schedules.

Three levels of hourly disruption cost, high cost ($c_{\text{H}} = 10$), medium cost ($c_{\text{M}} = 3$), and low cost ($c_{\text{L}} = 1$) are considered, while a zero cost is assigned to traffic-free intervals. The disruption function, which specifies the hourly disruption cost for any intervals in the planning horizon, can then be described by a lookup-table presented in Table 5. The planning horizon is the sampling time of the high-level controller (one month), and the parameters for the middle-level problem (22)–(29) are given in Table 6. The estimated time for maintenance $\widehat{T}_{\text{Maint}}$ is calculated from the interventions suggested by the high-level controller, i.e.

[10] Freight trains are not considered, as there is at most one freight train from Eindhoven to Weert every workday.

Table 5
Hourly disruption cost for a week. Interval with no trains passing are assigned a zero cost. Intervals with less than 2 trains per hour are assigned a low hourly cost c_L. Intervals with 2–3 trains per hour are assigned a medium hourly cost c_M. Intervals with 4–6 trains per hour are assigned a high hourly cost c_H.

Interval	Workday		Saturday		Sunday	
	Number of trains	disruption cost	Number of trains	Disruption cost	Number of trains	Disruption cost
0:00–1:00	1	c_L	2	c_L	2	c_L
1:00–6:00	0	0	0	0	0	0
6:00–7:00	3	c_M	0	0	0	0
7:00–8:00	6	c_H	4	c_H	1	c_L
8:00–13:00	6	c_H	6	c_H	5	c_H
13:00–14:00	5	c_H	6	c_H	5	c_H
14:00–19:00	6	c_H	6	c_H	5	c_H
19:00–20:00	5	c_H	5	c_H	4	c_H
20:00–24:00	3	c_M	3	c_M	3	c_M

Table 6
Paramters for the middle-level optimization problem.

Parameter	Explanation	Value
N_{sl}	Maximum number of time slots	2
c_{sl}	Setup cost of one time slot	1
λ_{sl}	Trade-off between disruption and setup cost	10 (representative run) 1 (supplementary run)
T_{set}	Setup time for one time slot	1 h
$\Delta\tau_{min}$	Minimum size of a time slot	5 h

Table 7
Paramters for the low-level optimization problem.

Parameter	Explanation	Value
N_{cl}	Maximum number of clusters	3
λ_{cl}	Penalty on the number of active clusters	1
$\Delta\phi_{min}$	Minimum cluster size	1 km
$\Delta\phi_{max}$	Maximum cluster size	25 km
T_{on}	Switch-on time of grinding machine	15 min
T_{off}	Switch-off time of grinding machine	15 min
v_{on}	Grinding speed	2.2 km/h (representative run) 1.6 km/h (supplementary run)
v_{off}	Driving speed of grinding machine	80 km/h

$$\widehat{T}_{\text{Maint}} = T_{\text{Sec}} \sum_{j=1}^{n} I_{u_j=1} \tag{47}$$

where $T_{\text{Sec}} = 2.5$ is the estimated time (in hours) to grind one section.

B.3. Low-level problem

As only squats located in the section where grinding is suggested by the high-level controller are considered in the low-level optimization problem, the number (N^{bu}) and the locations (ξ) of the squats are not fixed. However, we have $N_{\text{bu}} \leqslant 454$ as there are 454 existing squats on the whole track. The vector w containing the lengths of the considered squats also changes at each time step. The parameters for the low-level problem (30)–(37) is given in Table 7. Only one slot is used in the middle-level problem, thus $s = 1$. The setup time T_{set} in constraint (37) is the same as in the middle-level problem, and the actual length of the time slot T_s^{sl} is the one obtained from the results of the middle-level problem (22)–(29).

Appendix C. Transformation into an MILP problem

C.1. MILP formulation for the slot allocation problem

In this section we explain how to transform the nonsmooth optimization problem (22)–(29) into a standard MILP problem following the procedure described in Bemporad and Morari (1999).

First we introduce the following binary variables:

$$\overline{\delta}_{i,j} = 1 \iff t_i^{\text{end}} - \tau_j \leqslant 0 \tag{48}$$
$$\underline{\delta}_{i,j} = 1 \iff t_i^{\text{start}} - \tau_j \leqslant 0 \tag{49}$$
$$\forall i \in \{1,\ldots,N_{\text{sl}}\}, \forall j \in \{1,\ldots,N_\tau + 1\}.$$

Then we introduce the following continuous variables:

$$\overline{z}_{i,j} = \overline{\delta}_{i,j} t_i^{\text{start}} \tag{50}$$
$$\underline{z}_{i,j} = \underline{\delta}_{i,j} t_i^{\text{end}} \tag{51}$$
$$\forall i \in \{1,\ldots,N_{\text{sl}}\}, \forall j \in \{1,\ldots,N_\tau + 1\}.$$

The procedure to transform the new variables (48)–(51) to equivalent linear constraints can be found in Section 2 of Bemporad and Morari (1999).

Define

$$\delta = [\underbrace{\overline{\delta}_{1,1} \ldots \overline{\delta}_{N_{\text{sl}},N_\tau+1}}_{\overline{\delta}^{\text{T}}} \ \underbrace{\underline{\delta}_{1,1} \ldots \underline{\delta}_{N_{\text{sl}},N_\tau+1}}_{\underline{\delta}^{\text{T}}}]^{\text{T}}$$
$$z = [\underbrace{\overline{z}_{1,1} \ldots \overline{z}_{N_{\text{sl}},N_\tau+1}}_{\overline{z}^{\text{T}}} \ \underbrace{\underline{z}_{1,1} \ldots \underline{z}_{N_{\text{sl}},N_\tau+1}}_{\underline{z}^{\text{T}}}]^{\text{T}}.$$

The equivalent MILP problem for (22)–(29) can then be expressed as:

$$\min_{t,\delta,z} \sum_{i=1}^{N_{\text{sl}}} \sum_{j=1}^{N_\tau} c_j (\tau_j \overline{\delta}_{i,j} - \tau_{j+1} \overline{\delta}_{i,j+1} + \overline{z}_{i,j+1} - \overline{z}_{i,j} - \tau_j \underline{\delta}_{i,j} + \tau_{j+1} \underline{\delta}_{i,j+1} - \underline{z}_{i,j+1} + \underline{z}_{i,j}) + \lambda_{\text{sl}} \sum_{i=1}^{N_{\text{sl}}} c_{\text{sl}} \overline{\delta}_{i,N_\tau+1} \tag{52}$$

subject to constraints (23)–(27) and

$$\underline{\delta}_{i,N_\tau+1} - \overline{\delta}_{i,N_\tau+1} = 0 \quad \forall i \in \{1,\ldots,N_{\text{sl}}\} \tag{53}$$
$$\sum_{i=1}^{N_{\text{sl}}} \overline{z}_{i,N_\tau+1} - \underline{z}_{i,N_\tau+1} - T_{\text{set}} \overline{\delta}_{i,N_\tau+1} \geqslant \widehat{T}_{\text{Maint}} \tag{54}$$

and the equivalent linear constraints (see Bemporad and Morari (1999)) for the new variables δ and z.

C.2. MILP formulation for the clustering problem

In this section we explain how to transform the nonsmooth optimization problem (30)–(37) into a standard MILP problem following the procedure described in Bemporad and Morari (1999).

First we introduce the following binary variables:

$$\delta_{i,j}^{\text{end}} = 1 \iff \phi_i^{\text{end}} - \xi_j \leqslant 0 \tag{55}$$
$$\delta_{i,j}^{\text{start}} = 1 \iff \phi_i^{\text{start}} - \xi_j \leqslant 0 \tag{56}$$
$$\overline{\delta}_i = 1 \iff \phi_i^{\text{end}} - \overline{\xi} \leqslant 0 \tag{57}$$
$$\underline{\delta}_i = 1 \iff \phi_i^{\text{start}} - \overline{\xi} \leqslant 0 \tag{58}$$
$$\forall i \in \{1,\ldots,N^{\text{cl}}\}, \forall j \in \{1,\ldots,|\mathcal{I}_s|\}.$$

Then we introduce the following continuous variables:

$$z_{1,i} = \underline{\delta}_i \phi_i^{\text{start}} \tag{59}$$
$$z_{2,i} = \overline{\delta}_i \phi_i^{\text{end}} \tag{60}$$
$$z_{3,i} = \underline{\delta}_i \phi_{i+1}^{\text{start}} \tag{61}$$
$$\forall i \in \{1,\ldots,N^{\text{cl}}\}.$$

Define

$$\delta = [\underbrace{\delta_{1,1}^{\text{start}} \ldots \delta_{N^{\text{cl}},|\mathcal{I}_s|}^{\text{start}}}_{(\delta^{\text{start}})^{\text{T}}} \ \underbrace{\delta_{1,1}^{\text{end}} \ldots \delta_{N^{\text{cl}},|\mathcal{I}_s|}^{\text{end}}}_{(\delta^{\text{end}})^{\text{T}}} \ \underbrace{\underline{\delta}_1 \ldots \underline{\delta}_{N^{\text{cl}}}}_{\underline{\delta}^{\text{T}}} \ \underbrace{\overline{\delta}_1 \ldots \overline{\delta}_{N^{\text{cl}}}}_{\overline{\delta}^{\text{T}}}]^{\text{T}}$$
$$z = [\underbrace{z_{1,1} \ldots z_{1,N^{\text{cl}}}}_{z_1^{\text{T}}} \ \underbrace{z_{2,1} \ldots z_{2,N^{\text{cl}}}}_{z_2^{\text{T}}} \ \underbrace{z_{3,1} \ldots z_{3,N^{\text{cl}}}}_{z_3^{\text{T}}}]^{\text{T}}$$

The equivalent MILP of (30)–(37) can be expressed as:

$$\min_{\phi,\delta,z}\sum_{i=1}^{N^{\mathrm{cl}}}\left(\sum_{j=1}^{N^{\mathrm{bu}}}w_j(\delta_{i,j}^{\mathrm{end}}-\delta_{i,j}^{\mathrm{start}})+\lambda\overline{\delta}_i\right) \tag{62}$$

subject to constraint (31)–(35) and

$$\underline{\delta}_i-\overline{\delta}_i=0 \quad \forall i\in\{1,\ldots,N^{\mathrm{cl}}\} \tag{63}$$

$$\sum_{i=1}^{N^{\mathrm{cl}}-1}\left(\left(\frac{1}{v_{\mathrm{on}}}-\frac{1}{v_{\mathrm{off}}}\right)z_{2,i}-\frac{1}{v_{\mathrm{on}}}z_{1,i}+\frac{1}{v_{\mathrm{off}}}z_{3,i}+(T_{\mathrm{on}}+T_{\mathrm{off}})\overline{\delta}_i\right) \tag{64}$$
$$+\frac{1}{v_{\mathrm{on}}}(z_{2,N^{\mathrm{cl}}}-z_{1,N^{\mathrm{cl}}})+(T_{\mathrm{on}}+T_{\mathrm{off}})\overline{\delta}_{N^{\mathrm{cl}}}\leqslant T^{\mathrm{sl}}-T_{\mathrm{set}}$$

and the equivalent linear constraints for the new variables δ and z.

Appendix D. Cyclic approach

In this section we briefly introduce the cyclic approach mentioned in Section 5.4. Let $t_{0,j}$ denote the time instants at which the first grinding is applied to the j-th sections. Let $T_{\mathrm{Gr},j}$ denote the period of grinding for section j. Replacing is usually performed after a multiple of consecutive grindings, e.g. a section of rail is replaced after 5 grindings. We denote this multiple by a scalar r, which is the same for all sections. Define $t_0=[t_{0,1}\ldots t_{1,n}]^{\mathrm{T}}$ and $T_{\mathrm{Gr}}=[T_{\mathrm{Gr},1}\ldots T_{\mathrm{Gr},n}]^{\mathrm{T}}$, the cyclic maintenance optimization problem can then be formulated as:

$$\min_{t_0,T_{\mathrm{Gr}},r}\sum_{k=1}^{k_{\mathrm{end}}}\sum_{j=1}^{n}x_j^{\mathrm{con}}(k)+\lambda(\gamma_{j,1}I_{u_j(k)=1}+\gamma_{j,2}I_{u_j(k)=2}) \tag{65}$$

subject to

$$x_j^{\mathrm{con}}(k+1)=f^{\mathrm{con}}(x_j^{\mathrm{con}}(k),u_j(k),2) \tag{66}$$

$$x_j^{\mathrm{con}}(k)\leqslant x_{\max} \tag{67}$$

$$u_j(k)=\begin{cases}1, & \text{if } k=t_{0,j} \text{ or } (k-t_{0,j}) \bmod \mathrm{round}\,(T_{\mathrm{Gr},j})=0\\ 2, & \text{if } (k-t_{0,j}) \bmod \mathrm{round}\,(rT_{\mathrm{Gr},j})=0\\ 0, & \text{otherwise}\end{cases} \tag{68}$$

$$t_0^{\min}\leqslant t_0\leqslant t_0^{\max} \tag{69}$$

$$T_{\mathrm{Gr}}^{\min}\leqslant T_{\mathrm{Gr}}\leqslant T_{\mathrm{Gr}}^{\max} \tag{70}$$

$$2\leqslant r\leqslant r_{\max} \tag{71}$$

for all $j\in\{1,\ldots,n\}$ and $k\in\{1,\ldots,k_{\mathrm{end}}\}$.

The objective (65) corresponds to minimizing the accumulated condition degradation and intervention cost for the entire track over the five-year planning horizon. Only the average (nominal) deterioration rate ($\theta(k)=2$ for all k) is considered, as shown in (66). Constraint (67) guarantees that the nominal condition will not exceed the maintenance limit for the whole planning horizon. Eq. (68) converts the grinding and replacing periods into interventions. Constraints (69) and (71) are upper and lower bounds for the decision variables. The offline cyclic maintenance optimization problem (65)–(71) is a nonsmooth optimization problem, which is solved using multi-start pattern search in the case study.

References

Al-Douri, Y., Tretten, P., Karim, R., 2016. Improvement of railway performance: a study of Swedish railway infrastructure. J. Modern Transport. 24, 22–37.
Bemporad, A., Morari, M., 1999. Control of systems integrating logic, dynamics, and constraints. Automatica 35, 407–427.
Ben-Daya, M., Kumar, U., Murthy, D., 2016. Condition-based maintenance. Introd. Mainten. Eng.: Model. Optim. Manage., 355–387
Bešinović, N., Goverde, R., Quaglietta, E., Roberti, R., 2016. An integrated micro–macro approach to robust railway timetabling. Transport. Res. Part B: Methodol. 87, 14–32.
Budai, G., Huisman, D., Dekker, R., 2006. Scheduling preventive railway maintenance activities. J. Oper. Res. Soc. 57, 1035–1044.
Caetano, L., Teixeira, P., 2016. Strategic model to optimize railway-track renewal operations at a network level. J. Infrastruct. Syst. 1076–0342.
Camacho, E., Alba, C., 2013. Model Predictive Control. Springer Science & Business Media.
Campo, P., Morari, M., 1987. Robust model predictive control, In: American Control Conference, pp. 1021–1026.
Dekker, R., 1996. Applications of maintenance optimization models: a review and analysis. Reliab. Eng. Syst. Safety 51, 229–240.
De Schutter, B., De Moor, B., 1998. Optimal traffic light control for a single intersection. Eur. J. Control 4, 260–276.
Durango-Cohen, P., Madanat, S., 2008. Optimization of inspection and maintenance decisions for infrastructure facilities under performance model uncertainty: a quasi-Bayes approach. Transport. Res. Part A: Policy Pract. 42, 1074–1085.
Esveld, C., 2001. Modern Railway Track. MRT-Productions Zaltbommel.
Famurewa, S., Xin, T., Rantatalo, M., Kumar, U., 2015. Optimisation of maintenance track possession time: a tamping case study. Proc. Inst. Mech. Eng., Part F: J. Rail Rapid Transit 229, 12–22.
Fan, Y., Dixon, S., Edwards, R., Jian, X., 2007. Ultrasonic surface wave propagation and interaction with surface defects on rail track head. NDT & E Int. 40, 471–477.

Fararooy, S., Allan, J., 1995. Condition-based maintenance of railway signalling equipment. In: International Conference on Electric Railways in a United Europe. IET, pp. 33–37.
Frangopol, D., Kallen, M., Noortwijk, J.V., 2004. Probabilistic models for life-cycle performance of deteriorating structures: review and future directions. Prog. Struct. Eng. Mater. 6, 197–212.
Goverde, R., Bešinović, N., Binder, A., Cacchiani, V., Quaglietta, E., Roberti, R., Toth, P., 2016. A three-level framework for performance-based railway timetabling. Transport. Res. Part C: Emerg. Technol. 67, 62–83.
Grosso, J., Ocampo-Martínez, C., Puig, V., Joseph, B., 2014. Chance-constrained model predictive control for drinking water networks. J. Process Control 24, 504–516.
Gruber, J., Ramirez, D., Limon, D., Alamo, T., 2013. Computationally efficient nonlinear min-max model predictive control based on Volterra series models – application to a pilot plant. J. Process Control 23, 543–560.
Guler, H., 2012. Decision support system for railway track maintenance and renewal management. J. Comput. Civil Eng. 27, 292–306.
He, Q., Li, H., Bhattacharjya, D., Parikh, D., Hampapur, A., 2015. Track geometry defect rectification based on track deterioration modelling and derailment risk assessment. J. Oper. Res. Soc. 66, 392–404.
Higgins, A., 1998. Scheduling of railway track maintenance activities and crews. J. Oper. Res. Soc. 49, 1026–1033.
Jamshidi, A., Núñez, A., Dollevoet, R., Li, Z., 2017. Robust and predictive fuzzy key performance indicators for condition-based treatment of squats in railway infrastructures. Journal of Infrastructure Systems 23 (3), 04017006.
Jardine, A., Lin, D., Banjevic, D., 2006. A review on machinery diagnostics and prognostics implementing condition-based maintenance. Mech. Syst. Signal Process. 20, 1483–1510.
Jurado, I., Maestre, J., Velarde, P., Ocampo-Martinez, C., Fernández, I., Tejera, B.I., del Prado, J., 2016. Stock management in hospital pharmacy using chance-constrained model predictive control. Comput. Biol. Med. 72, 248–255.
Kobbacy, K., Murthy, D., 2008. Complex System Maintenance Handbook. Springer Science & Business Media.
Lewis, R., Torczon, V., Trosset, M., 2000. Direct search methods: then and now. J. Comput. Appl. Math. 124, 191–207.
Li, H., Parikh, D., He, Q., Qian, B., Li, Z., Fang, D., Hampapur, A., 2014. Improving rail network velocity: a machine learning approach to predictive maintenance. Transport. Res. Part C: Emerg. Technol. 45, 17–26.
Li, Z., Molodova, M., Núñez, A., Dollevoet, R., 2015. Improvements in axle box acceleration measurements for the detection of light squats in railway infrastructure. IEEE Trans. Indust. Electron. 62, 4385–4397.
Ling, L., Xiao, X., Jin, X., 2014. Development of a simulation model for dynamic derailment analysis of high-speed trains. Acta Mech. Sinica 30, 860–875.
Madanat, S., 1993. Optimal infrastructure management decisions under uncertainty. Transport. Res. Part C: Emerg. Technol. 1, 77–88.
Mercier, S., Meier-Hirmer, C., Roussignol, M., 2012. Bivariate Gamma wear processes for track geometry modelling, with application to intervention scheduling. Struct. Infrastruct. Eng. 8, 357–366.
Molodova, M., Li, Z., Núñez, A., Dollevoet, R., 2014. Automatic detection of squats in railway infrastructure. IEEE Trans. Intell. Transport. Syst. 15, 1980–1990.
Morari, M., Zafiriou, E., 1989. Robust Process Control, vol. 488. Prentice Hall, Englewood Cliffs, NJ.
Nandola, N., Rivera, D., 2013. An improved formulation of hybrid model predictive control with application to production-inventory systems. IEEE Trans. Control Syst. Technol. 21, 121–135.
Nicolai, R., Dekker, R., 2008. Optimal Maintenance of Multi-Component Systems: A Review. Springer.
Peng, F., Ouyang, Y., 2012. Track maintenance production team scheduling in railroad networks. Transport. Res. Part B: Methodol. 46, 1474–1488.
Peng, F., Ouyang, Y., 2014. Optimal clustering of railroad track maintenance jobs. Comput.-Aid. Civil Infrastruct. Eng. 29, 235–247.
Peng, F., Kang, S., Li, X., Ouyang, Y., Somani, K., Acharya, D., 2011. A heuristic approach to the railroad track maintenance scheduling problem. Comput.-Aid. Civil Infrastruct. Eng. 26, 129–145.
Prekopa, A., 1970. On probabilistic constrained programming. In: Proceedings of the Princeton Symposium on Mathematical Programming. Princeton University Press, Princeton, NJ, pp. 113–138.
Quaglietta, E., Pellegrini, P., Goverde, R., Albrecht, T., Jaekel, B., Marlière, G., Rodriguez, J., Dollevoet, T., Ambrogio, B., Carcasole, D., Giaroli, M., Nicholson, G., 2016. The ON-TIME real-time railway traffic management framework: a proof-of-concept using a scalable standardised data communication architecture. Transport. Res. Part C: Emerg. Technol. 63, 23–50.
Quiroga, L., Schnieder, E., 2012. Monte Carlo simulation of railway track geometry deterioration and restoration. Proc. Inst. Mech. Eng. Part O: J. Risk Reliab. 226, 274–282.
Rawlings, J., Mayne, D., 2009. Model Predictive Control: Theory and Design. Nob Hill Publishing, Madison, Wisconsin.
Sadowska, A., van Overloop, P., Burt, C., De Schutter, B., 2014. Hierarchical operation of water level controllers: formal analysis and application on a large scale irrigation canal. Water Resour. Manage. 28, 4999–5019.
Sandström, J., Ekberg, A., 2009. Predicting crack growth and risks of rail breaks due to wheel flat impacts in heavy haul operations. Proc. Inst. Mech. Eng. Part F: J. Rail Rapid Transit 223, 153–161.
Santos, R., Teixeira, P., 2012. Heuristic analysis of the effective range of a track tamping machine. J. Infrastruct. Syst. 18, 314–322.
Scarf, P., 1997. On the application of mathematical models in maintenance. Eur. J. Oper. Res. 99, 493–506.
Smilowitz, K., Madanat, S., 2000. Optimal inspection and maintenance policies for infrastructure networks. Comput.-Aid. Civil Infrastruct. Eng. 15, 5–13.
Song, Z., Yamada, T., Shitara, H., Takemura, Y., 2011. Detection of damage and crack in railhead by using eddy current testing. J. Electromagnet. Anal. Appl. 3, 546.
Su, Z., Núñez, A., Jamshidi, A., Baldi, S., Li, Z., Dollevoet, R., De Schutter, B., 2015. Model predictive control for maintenance operations planning of railway infrastructures. In: Computational Logistics (Proceedings of the 6th International Conference on Computational Logistics (ICCL'15), Delft, The Netherlands, Sept. 2015), pp. 673–688.
van Ekeren, H., Negenborn, R., van Overloop, P., De Schutter, B., 2013. Time-instant optimization for hybrid model predictive control of the Rhine–Meuse delta. J. Hydroinform. 15, 271–292.
Wen, M., Li, R., Salling, K., 2016. Optimization of preventive condition-based tamping for railway tracks. Eur. J. Oper. Res. 252, 455–465.
Zafra-Cabeza, A., Ridao, M., Camacho, E., 2008. Using a risk-based approach to project scheduling: a case illustration from semiconductor manufacturing. Eur. J. Oper. Res. 190, 708–723.
Zafra-Cabeza, A., Maestre, J., Ridao, M., Camacho, E., Sánchez, L., 2011. Hierarchical distributed model predictive control for risk mitigation: an irrigation canal case study. J. Process Control, 787–799.
Zhang, T., Andrews, J., Wang, R., 2013. Optimal scheduling of track maintenance on a railway network. Qual. Reliab. Eng. Int. 29, 285–297.
Zilko, A., Kurowicka, D., Goverde, R., 2016. Modeling railway disruption lengths with Copula Bayesian Networks. Transport. Res. Part C: Emerg. Technol. 68, 350–368.
Zoeteman, A., 2001. Life cycle cost analysis for managing rail infrastructure. Eur. J. Transport Infrastruct. Res. EJTIR 1 (4).