Two-stage stochastic mixed-integer programming with chance constraints for extended aircraft arrival management

Ahmed Khassiba, Fabian Bastin, Sonia Cafieri, Bernard Gendron, Marcel Mongeau

Two-stage stochastic mixed-integer programming with chance constraints for extended aircraft arrival management

Ahmed Khassiba [*†‡] Fabian Bastin[‡] Sonia Cafieri[†]
Bernard Gendron[‡] Marcel Mongeau[†]

Abstract

The extended aircraft arrival management problem, as an extension of the classic Aircraft Landing Problem, seeks to pre-schedule aircraft on a destination airport a few hours before their planned landing times. A two-stage stochastic mixed-integer programming model enriched by chance constraints is proposed in this paper. The first-stage optimization problem determines an aircraft sequence and target times over a reference point in the terminal area, called initial approach fix (IAF), so as to minimize the landing sequence length. Actual times over the IAF are assumed to deviate randomly from target times following known probability distributions. In the second stage, actual times over the IAF are assumed to be revealed, and landing times are to be determined in view of minimizing a time-deviation impact cost function. A Benders reformulation is proposed and acceleration techniques to Benders decomposition are sketched. Extensive results on realistic instances from Paris Charles-de-Gaulle airport show the benefit of two-stage stochastic and chance-constrained programming over a deterministic policy.

1 Introduction

Predicted growth in air traffic, capacity limitations of the overall air transportation system, environmental and human-factor challenges have been the main motivations for air transportation experts to formulate and tackle problems arising in Air Traffic Management (ATM). At the airport level, landings are considered to be among the most critical, bottleneck operations, where safety and efficiency are of great importance. Accordingly, the Aircraft Landing Problem (ALP) was introduced more than four decades ago (see Dear 12 and, later, Bennell et al. 4). The ALP deals with *sequencing* and *scheduling* aircraft landings optimally on the available runways at a given airport. Sequencing consists in finding an order among the considered aircraft, while scheduling is related to the timing of aircraft landings. Optimality criteria usually include maximizing airport throughput or minimizing aircraft delay, while satisfying operational and safety constraints, mainly

[*]Ph.D. candidate, corresponding author, ahmed.khassiba@outlook.com
[†]Ecole Nationale de l'Aviation Civile, Toulouse, France
[‡]Université de Montréal, DIRO, CIRRELT, Montréal, QC, Canada

separation constraints between operating aircraft near the runway threshold, called *final-approach separations*. Final-approach separations are based on aircraft wake-turbulence categories, presented in Table 1, and are expressed as inter-aircraft distances in nautical miles (NM; 1 NM = 1.852 m) as in Table 2. The difficulty of the ALP is due to the non-symmetry of the final approach separations, unlike in other flight phases. For example, in the near-to-airport airspace, called the *terminal area*, before the final approach phase, aircraft are horizontally separated by 5 NM.

Table 1: Wake-turbulence categories (WTC) according to the International Civil Aviation Organization (ICAO).

WTC	Max certificated take-off mass (kg)	Aircraft-type examples
Heavy (H)	above $136,000$	A350, A340, B747, B777
Medium (M)	between $7,000$ and $136,000$	A320, B737
Light (L)	below $7,000$	General aviation and executive jets

source: `https://www.skybrary.aero/index.php/ICAO_Wake_Turbulence_Category`

Table 2: Minimal final-approach separations (NM) according to ICAO's wake-turbulence categories.

		Following aircraft		
		H	M	L
Leading aircraft	H	4	5	6
	M	2.5	2.5	4
	L	2.5	2.5	2.5

source: de Neufville et al. 11

On the operational side, since the early 90's in the USA and Europe, air traffic controllers (ATCs), responsible for air traffic flows' safety and efficiency around major airports, have been using decision-support tools that attempt to sequencing and scheduling landings optimally at available runways according to ATCs' input criteria [20, 37, 44, 21]. Nowadays, the main such tool in the USA is known as the Traffic Management Advisor, while in Europe it is called Arrival Manager (AMAN). Without loss of generality, we will retain the European naming in the sequel. AMAN typically captures inbound aircraft at distances under 200 NM from their destination airport *i.e.*, around 40 minutes before landing [13, 43]. Then, using predicted landing times and aircraft characteristics such as wake-turbulence categories, AMAN determines an "optimal" landing sequence and target landing times according to the ATCs' input criteria. Afterwards, the controllers have to communicate control actions to pilots in order to enforce this optimal sequence, and to satisfy as far as possible the target landing times. Apart from recoursing to *holding stacks*, where aircraft keep flying in a circle at low altitudes close to the *terminal area* (formally called *Terminal Control Area* (TCA) in the USA, and *Terminal Maneuvering Area* (TMA) in Europe), controllers are allowed to change the aircraft speeds and trajectories in order to avoid terminal area congestion. The latter two control actions are more likely to achieve the so-called *linear holding*, which is preferred to holding stacks in terms of safety, ATC workload and eco-efficiency. However, recoursing to linear holding is most effective when flights are still relatively far from the destination airport, *e.g.* while still in their cruise phase.

These facts motivate the extension of AMAN's horizon in order to reduce the need for holding stacks and to rely more on linear holding techniques. Accordingly, important ATM research and development programs, NextGen in the USA and SESAR in Europe, foresee their decision support tools' operational horizons to be extended up to 500 NM, *i.e.,* about 2 hours before landing [43]. The new European decision-support tool is called *Extended-AMAN* (E-AMAN). However, with extended horizons come greater uncertainties on the predicted times, such as those used by AMAN, when optimizing the landing sequence [34, 43]. A recent attempt to quantify the uncertainty on predicted landing times at a horizon of three hours, using actual flight data, is presented in Tielrooij et al. 43. To deal with predicted-time errors, current AMANs rely on regularly re-optimizing the schedule (for example every time aircraft data are updated). Although it may appear satisfying in practice, re-optimizing addresses uncertainty by brute force instead of embedding it within the optimization problem. One of the obvious drawbacks of frequent re-optimization is the instability of the optimal sequence it produces. With an extended operational horizon, addressing uncertainty through frequent re-optimizations will, very likely, result in highly-instable sequences, increasing the workload of ATCs which cannot easily build and maintain the continuously-changing sequence of aircraft.

To the best of our knowledge, the ALP has been most-commonly studied when considering the *deterministic* case [12, 3, 2, 4, 19], while uncertainty has less often been taken into account. Pioneer studies of the ALP considering uncertainty were conducted by [34, 9], and [32] who basically added probabilistic considerations to the deterministic ALP. Stochastic optimization models, including two-stage and multi-stage models, were applied by [42, 40, 41], and [7] to address a variant of the ALP under uncertainty that considers departures and surface operations on the airport. Recently, [23, 26], and [22] proposed various robust optimization models to address the *runway scheduling problem* under uncertainty. The aforementioned studies focus on the ALP under uncertainty with an operational horizon under one hour, whereas [26] investigates the *pre-tactical ALP* which starts several hours before the planned landing times. In Kapolke et al. 26, a simplified one-stage stochastic optimization model is compared with several robust optimization models. Their study tends to show that robust optimization is more promising than stochastic optimization for solving the pre-tactical ALP. Remark that the proposed one-stage stochastic optimization model only addresses the "expected-scenario" problem, *i.e.,* the variant in which uncertain data are replaced by their expected values. However, as stated by [6]: "planning for the expected case is in fact 'forgetting' uncertainty". We believe there is room for considering more practical aspects and algorithmic enhancements in order to get the most from stochastic optimization applied to the ALP under uncertainty.

In this paper, we consider an *extended Aircraft Landing Problem*, the ALP variant in which the operational horizon is extended, as for E-AMAN. Moreover, we aim at embedding the uncertainty *within* the optimization model. In view of simplifying the presentation, the scope of this preliminary study is limited to the case involving a single reference point in the terminal area, called the *initial approach fix* (IAF), and a single landing runway. We propose a two-stage stochastic optimization model with recourse, that is enhanced by probability constraints in the first stage, to mitigate the risk of separation violations over the IAF. Our two-stage stochastic model seeks to find a schedule that minimizes both the runway sequence length and the expected time-deviation impact costs. In a first stage, aircraft are sequenced and scheduled at the IAF so as to minimize the runway sequence length. In this stage, while *IAF target times* are decision variables,

IAF actual times are considered to be random variables. In a hypothetical second stage, uncertainty is assumed to be revealed and aircraft are scheduled to the runway threshold so as to minimize the time-deviation impact costs. The first-stage problem boils down to the classical Asymmetric Traveling Salesman Problem with Time Windows (ATSP-TW), an NP-hard problem which can be modeled as a mixed-integer linear program (MILP). Assuming a piecewise linear time-deviation impact cost function, the second-stage problem reduces to a simple linear program (LP). Hence, in this paper, we propose a two-stage stochastic *mixed-integer* programming model, where some first-stage variables are binary and the remaining first- and second-stage variables are continuous. Such two-stage stochastic programming models (with integer variables only in the first stage) have already been considered in the literature (see e.g., Wollmer 45, Laporte and Louveaux 31). In this context, the convexity of the second-stage problem is conserved and much of the theory and algorithms of two-stage stochastic linear programming are still applicable, as observed e.g., by Ahmed [1] and Birge and Louveaux [6, Section 3.3]. For our proposed model, a partially-aggregated Benders reformulation is proposed. The implementation of Benders decomposition is discussed and some acceleration techniques are sketched. We present computational results using the state-of-the-art MILP solver CPLEX, which allows to simplify the development of a Benders decomposition algorithm.

The paper is organized as follows. The problem statement along with the operational context are introduced in Section 2. In Section 3, we propose a two-stage stochastic model with recourse. Solution methods are proposed in Section 4. Results of numerical experiments are discussed in Section 5. Section 6 presents some conclusions and future research tracks.

2 Problem statement

We consider a set of aircraft planning to land at a given destination airport in two to three hour look-ahead time. For the sake of simplifying the exposition, in this preliminary study we make the following two operational assumptions. Firstly, all considered aircraft pass over the same IAF to prepare for landing. Secondly, all aircraft land on the same runway of the considered airport. Paris Charles-de-Gaulle airport (CDG) is an illustration of this simplified setting when arrival flows from North and South are disaggregated, the subset of aircraft coming from a same corner (north-west, for example) passes over a same IAF and lands on a same runway.

Our problem involves two types of separations: final-approach separations and separation over the IAF. For modeling and optimization purposes, separations expressed in terms of nautical miles are converted to seconds. Remark that this is a common practice in the literature; Table 3 shows final-approach separations converted to seconds, as used in CDG. Detail of such a conversion may be found in de Neufville et al. 11. For the sake of exposition simplification, we assume that all aircraft ground speeds over the IAF are equal to 250 knots (1 knots = 1 NM per hour), which is typically the maximal allowed on-board indicated air speed over the IAF. Hence, the usual 5 NM minimal separation over the IAF may be converted into 72 seconds.

Given a set of aircraft, we seek to find a *target aircraft sequence over the IAF*, and a *target time over the IAF* for each aircraft. We assume that the target sequence over the IAF is the same as the target landing sequence. In the sequel, the *target sequence* will equivalently designate any of these two target sequences. We aim at finding a target sequence so as to maximize the runway throughput. Target times over the IAF have

Table 3: Final-approach separations (seconds) at CDG according to ICAO's wake-turbulence categories

		Following aircraft		
		H	M	L
Leading aircraft	H	96	157	207
	M	60	69	123
	L	60	69	82

to satisfy the separation requirements over the IAF. *Actual times over the IAF* correspond to the times at which aircraft effectively pass over the IAF. The order in which aircraft effectively pass over the IAF is called the *actual sequence over the IAF*. We assume that actual times over the IAF randomly deviate from the target times following known distributions. These deviations are unknown when the target sequence and the target times over the IAF are decided. Because of these deviations, actual times may violate the separation constraints over the IAF, even though aircraft were safely separated in terms of target times. Also, the actual sequence over the IAF may differ from the target sequence. In practice, ATCs have to make control decisions to prevent such violations over the IAF, and to build the target sequence for landing. To limit subsequent delay impact costs (such as ATC workload), we consider probability constraints to express the acceptable rate of separation violations (in terms of actual times) over the IAF (for instance, one may expect these probability constraints to prevent excessive subsequent re-sequencing). Furthermore, target times over the IAF have to respect predefined time-window constraints. These constraints, when correctly defined, will prevent aircraft from being either excessively delayed or excessively expedited, with respect to their planned times. As suggested in [4], this may also help fulfill fairness requirements among aircraft, similarly to the more classical *constraint position shifting (CPS)* approach (see Balakrishnan and Chandran 2 for a comprehensive study in the deterministic case), where each aircraft position cannot be shifted by more than a predefined number of positions in the first-come first-serve sequence. In the case of tactical aircraft scheduling (typically 30 to 45 minutes before landing), the CPS constraints, in addition to being realistic, are known to reduce significantly the solution space, and thereby the problem complexity. However, in our operational context of an extended horizon of 2 to 3 hours before landing, we expect any sequence to be potentially feasible (as long as time-window constraints are satisfied), which hinders setting of any appropriate "maximum position shifting" parameter. Also, according to [4], "maximum *time* shifting" induced by time-window constraints would be preferable to CPS. For the two reasons mentioned above, we choose not to consider CPS constraints in our problem statement. Because of the uncertainty on actual times, decisions over the IAF (target sequence and target times) may result in different air traffic situations over the IAF (actual sequence and actual times) that are likely to deteriorate runway throughput and to incur further delay costs.

We define the hypothetical *second stage* in view of taking into account (ideally all) the different outcomes of the decisions over the IAF, subsequently called the *first-stage* decisions. In this second stage, deviations from target times over the IAF are assumed to be revealed. The second-stage problem consists in finding a target landing time for each aircraft in order to minimize a second-stage cost, while satisfying realistic flight times through the terminal area and, more importantly, without violating final-approach separations. These new scheduling decisions represent the ATCs recourse to handle the

air traffic situation from the IAF to the runway threshold once uncertainties are revealed. Therefore, we seek to minimize the expected cost of this recourse through considering different (eventually all) *scenarios*, *i.e.*, realizations of the uncertainties. We assume that second-stage decisions are required only when all uncertainties are revealed. For this assumption to hold, we have to ensure that the set of considered aircraft will arrive at the IAF during a reasonably short time frame, so that when the uncertainty of the last aircraft is revealed, second-stage decisions can still be implemented. Moreover, in more operational terms, as suggested in [29], the beginning of the second stage can be set to the entry time of the last considered aircraft to the en-route sector neighboring the terminal area.

3 A two-stage stochastic optimization model with recourse

Let $\mathcal{A} = \{1, 2, \ldots, n\}$ be the set of aircraft indices to be sequenced and scheduled over the IAF. The minimal time separation required over the IAF is given and noted $\underline{S}^I$. The minimal time separation during the final approach between a leading aircraft $i \in \mathcal{A}$ and a following aircraft $j \in \mathcal{A}$ is also given; it is noted S_{ij} and called *final-approach separation* between aircraft i and j. In the first stage, the n aircraft need to be sequenced and scheduled over the IAF. Let δ_{ij} be the binary decision variable that takes the value 1 if and only if aircraft $i \in \mathcal{A}$ **directly precedes** aircraft $j \in \mathcal{A}$ in the sequence, and 0 otherwise. These variables are called the *sequencing variables*. We seek to find the aircraft sequence with the minimum length in terms of final-approach separations. Such a sequence can be obtained by solving an (open) Asymmetric Traveling Salesman Problem (ATSP) instance where the city set corresponds to the aircraft set $\mathcal{A}$ and distances between cities correspond to final-approach separations. Equivalently, we can consider a classical ATSP instance involving the set $\mathcal{A}^+ = \{1, 2, \ldots, n+1\}$, where index $n+1$ corresponds to a fictitious extra aircraft to close the Hamiltonian circuit. Then, $2n$ more sequencing binary decision variables, $\delta_{i,n+1}, \delta_{n+1,i}, i \in \mathcal{A}$, are introduced to take into account the $(n+1)^{st}$ aircraft. This spurious aircraft has null minimal time separation with the n original aircraft. To summarize, the (first-stage) sequencing variables are:

$$\delta_{ij} = \begin{cases} 1 & \text{if aircraft } i \textbf{ directly precedes} \text{ aircraft } j \\ 0 & \text{otherwise} \end{cases} \qquad (i,j) \in \mathcal{A}^+ \times \mathcal{A}^+, \quad i \neq j.$$

The sequence length can then be expressed easily using final-approach separations S_{ij} and the ATSP-like sequencing variables, as follows: $\sum_{\substack{(i,j)\in\mathcal{A}^+\times\mathcal{A}^+ \\ i\neq j}} \delta_{ij} S_{ij}$. Remark that with sequencing variables that express general relative (not necessarily direct) precedence between aircraft, as used in the literature (Beasley et al. 3), the sequence length would not be straightforward to express.

At a look-ahead time of two to three hours before landing, every aircraft $i \in \mathcal{A}$ has a fixed (given) time window $\left[E_i^I, L_i^I\right]$ to pass over the IAF, where E_i^I and L_i^I are respectively the given earliest and latest times. Let x_i be a first-stage decision variable representing the target time over the IAF of aircraft $i \in \mathcal{A}$; it must satisfy the bound constraints:

$$x_i \in \left[E_i^I, L_i^I\right], \qquad i \in \mathcal{A}.$$

Let $\boldsymbol{\omega_i}$ be the random variable representing the deviation of the actual time over the IAF of aircraft $i \in \mathcal{A}$ with respect to its target time x_i. Let ω_i be a realization of the random variable $\boldsymbol{\omega_i}$. Then, the actual time over the IAF of aircraft $i \in \mathcal{A}$ is simply $x_i + \omega_i$. Let $\alpha \in [0,\ 1]$ be the lowest acceptable probability that separation over the IAF is satisfied between the pair of aircraft $(i, j) \in \mathcal{A} \times \mathcal{A}$, $i \neq j$, once uncertainties are revealed.

In the hypothetical second stage, actual times over the IAF are assumed to be known with certainty. Recall that the actual sequence over the IAF might not correspond to the target sequence. As mentioned in Section 2, we choose to enforce the target landing sequence to be the same as the target sequence over the IAF, since the latter was computed so as to minimize the runway sequence length. Hence, no (re-)sequencing variables are needed in the second stage. However, the n aircraft need to be scheduled at the runway threshold. Let y_i be the decision variable representing the target landing time of aircraft $i \in \mathcal{A}$. These variables are called *second-stage scheduling variables* and have to satisfy the time separation constraints during the final approach. In order to keep these target times realistic, we introduce a landing time window $[E_i,\ L_i]$ for every aircraft $i \in \mathcal{A}$ so that second-stage variables must satisfy the bound constraints:

$$y_i \in [E_i, L_i], \qquad i \in \mathcal{A}.$$

For an aircraft i, recalling that the actual IAF time is $x_i + \omega_i$, the earliest and the latest landing times can be expressed using (given) minimal and maximal flight times from the IAF to the runway threshold, $\underline{V_i}$ and $\overline{V_i}$ respectively, where $0 < \underline{V_i} \leq \overline{V_i}$, as follows: $E_i = (x_i + \omega_i) + \underline{V_i}$ and $L_i = (x_i + \omega_i) + \overline{V_i}$.

Following [5], we define the *unconstrained* (or *uncongested*) *landing time* of aircraft $i \in \mathcal{A}$, noted U_i, to be the landing time of aircraft i as if it were alone in the terminal area, and the unconstrained flight time $\hat{V}_i$ such that $\underline{V_i} \leq \hat{V}_i \leq \overline{V_i}$ and $U_i = (x_i + \omega_i) + \hat{V}_i$.

To stress the difference between unconstrained and minimal flight times, remark that unconstrained flight time is achieved when an aircraft flies its preferred trajectory at its nominal speed. On the other hand, minimal flight time can be achieved, for example, if the aircraft slows down later than expected in the standard procedure (hence, keeping a high speed for a longer time), or if the approach controller gives a bit earlier the landing clearance to the pilot. These changes are both undesirable from a fuel-consumption perspective and in terms of controller workload.

Let f be a function that estimates time-deviation costs incurred in the second stage. In this study, we propose to model such costs with a convex piecewise linear function.

We introduce the following vector notation: $x = (x_1, x_2, \ldots, x_n)^T$. Vectors $\boldsymbol{\omega}$, ω, and y are defined likewise. We also introduce the matrix notation: $\delta = (\delta_{ij})_{\substack{(i,j) \in \mathcal{A}^+ \times \mathcal{A}^+ \\ i \neq j}}$. Given the expectation operator $\mathbb{E}_{\boldsymbol{\omega}}[.]$ over the random vector $\boldsymbol{\omega}$, and a weighting parameter λ, we propose the following two-stage stochastic optimization model with recourse, also called the *true model*:

$$\min_{\delta,\, x} \quad \sum_{\substack{(i,j)\in\mathcal{A}^+\times\mathcal{A}^+ \\ i\neq j}} \delta_{ij} S_{ij} + \lambda\, \mathbb{E}_{\boldsymbol{\omega}}[Q(\delta, x, \omega)] \tag{1}$$

$$\text{s.t.} \quad \sum_{\substack{j\in\mathcal{A}^+ \\ j\neq i}} \delta_{ji} = 1 \qquad i \in \mathcal{A}^+ \tag{2}$$

$$\sum_{\substack{j\in\mathcal{A}^+ \\ j\neq i}} \delta_{ij} = 1 \qquad i \in \mathcal{A}^+ \tag{3}$$

$$x_j \geq x_i + \underline{S}^I - M^I_{ij}(1-\delta_{ij}) \qquad (i,j) \in \mathcal{A}\times\mathcal{A},\quad i \neq j \tag{4}$$

$$\mathbb{P}(x_j + \omega_j \geq x_i + \omega_i + \underline{S}^I - M^{I\alpha}_{ij}(1-\delta_{ij})) \geq \alpha \qquad (i,j) \in \mathcal{A}\times\mathcal{A},\ i \neq j \tag{5}$$

$$E^I_i \leq x_i \leq L^I_i \qquad i \in \mathcal{A} \tag{6}$$

$$\delta_{ij} \in \{0,1\} \qquad (i,j) \in \mathcal{A}^+\times\mathcal{A}^+,\quad i \neq j \tag{7}$$

where:

$$Q(\delta, x, \omega) = \min_{y} \quad f(x, \omega, y) \tag{8}$$

$$\text{s.t.} \quad y_j \geq y_i + S_{ij} - M_{ij}(1-\delta_{ij}) \qquad (i,j) \in \mathcal{A}\times\mathcal{A},\quad i \neq j \tag{9}$$

$$\underline{V_i} \leq y_i - (x_i + \omega_i) \leq \overline{V_i} \qquad i \in \mathcal{A} \tag{10}$$

The objective function (1) is the weighted sum of: the first-stage objective function, $\sum_{\substack{(i,j)\in\mathcal{A}^+\times\mathcal{A}^+ \\ i\neq j}} \delta_{ij} S_{ij}$, and the expected cost of the second stage, $\mathbb{E}_{\boldsymbol{\omega}}[Q(\delta, x, \omega)]$. The first-stage problem minimizes the length of the sequence in terms of final-approach separations, subject to constraints (2) to (7). Given a scenario ω, the cost of the second-stage (so-called *recourse*) problem, $Q(\delta, x, \omega)$, is defined by (8) to (10). Big-M constants appearing in constraints (4), (5) and (9) will be further commented below.

First-stage model

Constraints (2), (3) and (4) are directly inspired from the classical ATSP formulation. Constraints (2) and (3) ensure that all aircraft in $\mathcal{A}^+$ are sequenced, which corresponds to visiting all cities in an ATSP. Constraints (4) express the minimal time separation requirement over the IAF between any two successive aircraft, where the big-M type constants M^I_{ij} are large enough so that the corresponding constraint is necessarily satisfied as soon as $\delta_{ij} = 0$. Constraints (5) are *individual* probability constraints that ensure separation based on actual times over the IAF between two given different aircraft with a probability higher than some given threshold value α. Under the assumption of independent and identically distributed (i.i.d.) random variables $\boldsymbol{\omega_i}$ for all $i \in \mathcal{A}$, probability constraints (5) can be expressed in a deterministic form analogous to the big-M separation constraints (4). This will be detailed in Subsection 3.2. Remark that α expresses a protection level against separation loss over the IAF between two given aircraft i and j. To express a protection level against any separation loss over the IAF (that is, there is no separation loss over the IAF α% of the time), we should recourse to the so-called *joint* chance constraints (Miller and Wagner 35). Constraints (6) are time-window constraints on target times over the IAF. Constraints (7) stipulate the binary nature of the δ_{ij} variables.

Without the probability constraints (5), the first-stage problem defined by the first-stage objective function and constraints (2) to (4), (6) and (7), reduces to an instance of

the ATSP with time windows (ATSP-TW). The reduction goes as follows: cities correspond to aircraft, the traveling salesperson corresponds to the IAF, costs of travel between cities is represented by final-approach separations (S_{ij}), and times of travel between cities correspond to the IAF separation ($\underline{S}^I$). Remark, however, that the scheduling part of the problem is a special simple case, since the IAF separation is not aircraft-dependent, unlike typical ATSP-TW travel time between cities. Finally, subtour elimination constraints are not required since the IAF separation constraints (4) play the role of MTZ constraints [36]. As mentioned above, big-M constants must be large enough for the formulation to be correct. However, very large big-M values are known to lead to numerical instabilities during resolution. The best expression (smallest while sufficiently large) for the big-M constants M_{ij}^I in (4) can easily be shown to be: $M_{ij}^I = L_i^I - E_j^I + \underline{S}^I$.

Second-stage model

With regard to the second-stage model, the objective function (8) minimizes a cost function f that represents the impact of time-deviation with respect to unconstrained landing times. This time-deviation impact can be interpreted as the additional workload of an *approach controller* applying AMAN recommendations (in terms of time deviations per aircraft) to handle the inbound traffic. To illustrate such an impact cost, consider an approach controller aided by AMAN. Typically, the responsibility of this controller is to ensure the arrival flow's safety and efficiency, from the time when inbound aircraft enter the terminal area until they align with the runway axis for landing. With a look-ahead time of 30 to 45 minutes (before landing), AMAN, as a decision support tool, computes a target landing sequence (according to predefined criteria) and provides the approach controller with recommendations in terms of time to lose or to gain for each aircraft (with respect to estimated landing times computed internally by AMAN) in order to build the target landing sequence. The approach controller's mission is to find adequate control instructions for each concerned aircraft (speed change, vectoring, holding patterns) in order to apply the time deviations recommended by AMAN. If AMAN computes no time to lose or to gain for a given aircraft, then the approach controller has only to supervise that flight and to give it the landing clearance at the right time, according to a standard procedure. On the other hand, applying a large time deviation induces a heavy workload for the approach controller. In more general terms, the shorter the time deviations (amounts of time to lose or to gain) displayed by AMAN, the lighter the workload of the approach controller.

A candidate expression of f is proposed in Subsection 3.1. Constraints (9) ensure final approach minimal time separation. Minimal and maximal flight times are enforced by constraints (10). Hence, the second-stage problem consists in finding a landing schedule for n aircraft that minimizes the cost function f, given a target sequence and landing time windows. Big-M constants M_{ij} in (9) can be computed as the lowest upper bound to $(y_i - y_j + S_{ij})$. Using constraints (9) and (10) and bound constraints (6) on x_i, the best expression for M_{ij} can be shown to be: $M_{ij} = \left(L_i^I + \omega_i + \overline{V_i}\right) - \left(E_j^I + \omega_j + \underline{V_j}\right) + S_{ij}$.

Note on the recourse type

We remark that for some first-stage solutions (x, δ), the second-stage problem may turn out to be infeasible: in our problem the recourse is *not relatively complete*. To give an example of a first-stage solution appearing to be infeasible for some second-stage scenario

problem, consider (p_1, p_2) a subsequence of two consecutive aircraft from a target sequence found in the first stage. Recall that this target sequence computed in the first stage for the IAF is intended to hold for landing (second-stage problem). Let x_{p_1} and x_{p_2} be target IAF times for the two considered aircraft. Consider a second-stage scenario s with IAF time deviations $\omega^s_{p_1}$ and $\omega^s_{p_2}$ for aircraft p_1 and p_2 respectively, such that $x_{p_1} + \omega^s_{p_1} \gg x_{p_2} + \omega^s_{p_2}$, i.e., aircraft p_1 arrives actually to the IAF much later than p_2. In such a scenario, the relative positions of aircraft p_1 and p_2 close to the IAF are inverted with respect to the target sequence. Assuming narrow landing time windows $\left[E^s_{p_1}, L^s_{p_1}\right]$ and $\left[E^s_{p_2}, L^s_{p_2}\right]$ such that $\left[E^s_{p_1}, L^s_{p_1}\right] \cap \left[E^s_{p_2}, L^s_{p_2}\right] = \emptyset$, there are no landing times $y^s_{p_1} \in \left[E^s_{p_1}, L^s_{p_1}\right]$ and $y^s_{p_2} \in \left[E^s_{p_2}, L^s_{p_2}\right]$ such that $y^s_{p_2} \geq y^s_{p_1} + S_{p_1,p_2}$. In other words, aircraft p_1 cannot land before aircraft p_2 in such a scenario, and the target subsequence (p_1, p_2) is infeasible for landing. In a real-life context, when the "real" uncertainties are revealed and give rise to an infeasible second-stage problem, re-sequencing is needed. This can be achieved by correcting the target sequence (returned by the stochastic program) by solving an additional deterministic scheduling problem.

3.1 Second-stage objective function: minimizing total time-deviation impact cost

A problem-specific second-stage objective is to minimize the total impact cost of time deviations with respect to unconstrained landing times. Considering a single aircraft $i \in \mathcal{A}$, we assume that a deviation, within predefined bounds, of this aircraft target time (y_i) with respect to its unconstrained landing time (U_i) has an impact cost proportional to the size of the deviation within these bounds. Larger time deviations are assumed to generate larger costs. Such a cost function, say f_i, can be described by a convex piecewise linear function of y_i that estimates the time-deviation impact cost of aircraft $i \in \mathcal{A}$. To compute the total cost over all the considered aircraft in set $\mathcal{A}$, we consider an additive total time-deviation impact cost function $f = \sum_{i \in \mathcal{A}} f_i$. The parameters defining the specific shape of each convex piecewise linear cost function, f_i, are to be defined by the user to match any stakeholder viewpoint (e.g., airlines, controllers, airports, etc).

To give an example, for a controller relying on AMAN to sequence and schedule aircraft for landing, this impact cost can be interpreted as a simplified estimation of the controller's additional workload. Indeed, a one-minute *advance* of an aircraft landing time is almost costless in terms of control workload, since he only has to give "a bit earlier" one instruction to the pilot, that is to follow the standard approach procedure. However, for a *delay* of one to four minutes, the approach controller has to communicate several instructions to modify the trajectory and/or the speed of the given aircraft. For delays larger than four minutes, the approach controller has to keep the aircraft in a holding stack, a predefined circular circuit in a confined space, often seen as an "airborne waiting room". Holding patterns are known to generate much more workload for controllers and for pilots than trajectory and speed changes. This progression of workload in terms of the delay (the time deviation to be implemented by the controller) can be captured by a convex piecewise linear cost function, made of three pieces, as formalized below. However, more sophisticated functions can be elaborated if controllers using AMAN are more involved in the modeling process.

Given the slopes $c_1, c_2, c_3 \in \mathbb{R}_+$ such that $c_2 \leq c_3$ and some intermediate landing times L^{med}_i, such that $U_i \leq L^{\text{med}}_i \leq L_i$, the following is an example of time-deviation impact

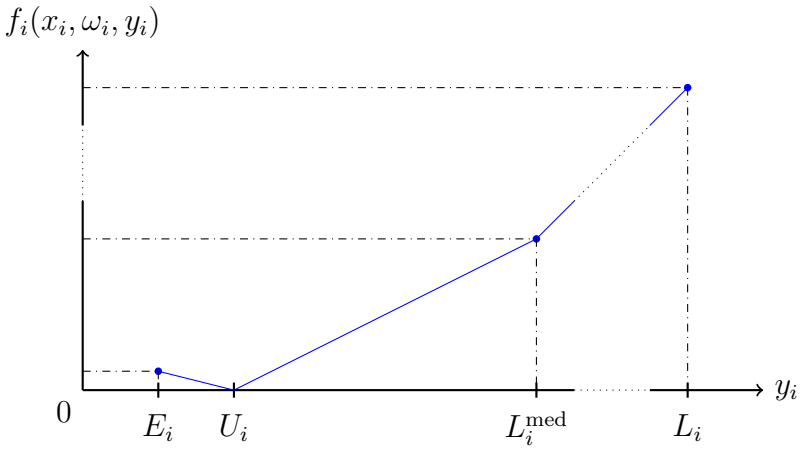


Figure 1: Time-deviation impact cost function f_i of aircraft $i \in \mathcal{A}$.

cost function f_i for an aircraft $i \in \mathcal{A}$ (Figure 1):

$$f_i(x_i, \omega_i, y_i) = \begin{cases} c_1 (U_i - y_i) & \text{if} \quad E_i \leq y_i \leq U_i \\ c_2 (y_i - U_i) & \text{if} \quad U_i \leq y_i \leq L_i^{\text{med}} \\ c_2 \left(L_i^{\text{med}} - U_i\right) + c_3 \left(y_i - L_i^{\text{med}}\right) & \text{if} \quad L_i^{\text{med}} \leq y_i \leq L_i \end{cases}$$

For an aircraft $i \in \mathcal{A}$, similarly to the definitions of E_i, U_i and L_i making use of appropriate flight times (minimal, unconstrained, and maximal), the intermediate landing time L_i^{med} can be defined using an intermediate flight time V_i^{med} such that $0 < \underline{V_i} \leq \hat{V}_i \leq V_i^{\text{med}} \leq \overline{V}_i$ and $L_i^{\text{med}} = (x_i + \omega_i) + V_i^{\text{med}}$.

Given such a separable convex piecewise-linear form, the objective function (8) can be linearized using, for the example above, three auxiliary variables z_i^-, z_i^+ and z_i^{++} per aircraft $i \in \mathcal{A}$ as follows:

$$\min_{\substack{y,\, z^-,\, z^+ \\ z^{++}}} \sum_{i \in \mathcal{A}} \left(c_1 z_i^- + c_2 z_i^+ + c_3 z_i^{++}\right) \tag{11}$$

$$y_i - U_i = z_i^+ + z_i^{++} - z_i^- \qquad i \in \mathcal{A} \tag{12}$$

$$z_i^+ \leq L_i^{\text{med}} \qquad i \in \mathcal{A} \tag{13}$$

$$z_i^-, z_i^+, z_i^{++} \geq 0 \qquad i \in \mathcal{A} \tag{14}$$

3.2 Reformulating probability constraints in the i.i.d. case

The aim of this Subsection is twofold. Firstly, we show that the chance constraints (5) can be reformulated as deterministic linear constraints analogous to the big-M separation constraints (4), under the assumption of independent and identically distributed random variables $\boldsymbol{\omega_i}$'s for all aircraft $i \in \mathcal{A}$ (Proposition 1). Secondly, we show that assuming normal random variables $\boldsymbol{\omega_i}$'s and for values of $\alpha \geq 0.5$, the linearized form of probability constraints (5) can substitute for the IAF separation constraints (4) in the true model (Proposition 2).

Lemma 1. *Consider a couple $(i, j) \in \mathcal{A} \times \mathcal{A}$ such that $i \neq j$ and the constraints:*

$$\mathbb{P}\left(x_j + \omega_j \geq x_i + \omega_i + \underline{S}^I - M_{ij}^{I\alpha}(1 - \delta_{ij})\right) \geq \alpha \tag{5_{ij}}$$

$$x_j \geq x_i + S^I(\alpha) - M_{ij}^{I\alpha}(1 - \delta_{ij}) \tag{17_{ij}}$$

where:

- $S^I(\alpha) \stackrel{def}{=} \underline{S}^I + F_{\boldsymbol{\gamma}}^{-1}(\alpha)$ *is a given time separation,*
- *and* $F_{\boldsymbol{\gamma}}^{-1}(\alpha)$ *is the* α *quantile of the random variable* $\boldsymbol{\gamma} \stackrel{def}{=} \boldsymbol{\omega_i} - \boldsymbol{\omega_j}$.

Assuming i.i.d. random variables $\boldsymbol{\omega_i}$ *and* $\boldsymbol{\omega_j}$*, the chance constraint* (5_{ij}) *is equivalent to the deterministic constraint* (17_{ij}).

Proof. Proof of Lemma 1

Consider a couple $(i,j) \in \mathcal{A} \times \mathcal{A}$ such that $i \neq j$. Then, the constraint (5_{ij}) can be re-written as:

$$\mathbb{P}\left(\omega_i - \omega_j \leq x_j - x_i - \underline{S}^I + M_{ij}^{I\alpha}(1-\delta_{ij})\right) \geq \alpha \tag{15}$$

Remark that, under this form, the i.i.d. random variables and the decisions are clearly decoupled. In this special case where the random variables only appear in the right-hand side of the expression inside the probability operator, the probability constraint can be re-written as a deterministic constraint using an inverse cumulative distribution function (Charnes and Cooper 10, Miller and Wagner 35).

As $\boldsymbol{\omega_i}$ and $\boldsymbol{\omega_j}$ are i.i.d., $(\boldsymbol{\omega_i} - \boldsymbol{\omega_j})$ is a random variable, that we denote $\boldsymbol{\gamma} \stackrel{def}{=} \boldsymbol{\omega_i} - \boldsymbol{\omega_j}$. Let $F_{\boldsymbol{\gamma}}$ be its distribution function.

Then (15) is equivalent to:

$$F_{\boldsymbol{\gamma}}\left(x_j - x_i - \underline{S}^I + M_{ij}^{I\alpha}(1-\delta_{ij})\right) \geq \alpha \tag{16}$$

Let us denote $F_{\boldsymbol{\gamma}}^{-1}(\alpha)$ the α quantile of the random variable $\boldsymbol{\gamma}$ and $S^I(\alpha) \stackrel{def}{=} \underline{S}^I + F_{\boldsymbol{\gamma}}^{-1}(\alpha)$, the buffered separation over the IAF.

Remark that, since α is a given parameter, then $F_{\boldsymbol{\gamma}}^{-1}(\alpha)$ and $S^I(\alpha)$ can be computed beforehand. Finally, (16) is equivalent to:

$$\begin{aligned} & x_j - x_i - \underline{S}^I + M_{ij}^{I\alpha}(1-\delta_{ij}) \geq F_{\boldsymbol{\gamma}}^{-1}(\alpha) \\ \Leftrightarrow \quad & x_j \geq x_i + S^I(\alpha) - M_{ij}^{I\alpha}(1-\delta_{ij}) \end{aligned}$$

□

The next Proposition directly follows from Lemma 1, by considering constraints (5_{ij}) and (17_{ij}) for all couples $(i,j) \in \mathcal{A} \times \mathcal{A}$ such that $i \neq j$.

Proposition 1. *The chance constraints* (5) *in the true model can be replaced by the following deterministic linear constraints:*

$$x_j \geq x_i + S^I(\alpha) - M_{ij}^{I\alpha}(1-\delta_{ij}) \qquad (i,j) \in \mathcal{A} \times \mathcal{A}, \quad i \neq j \tag{17}$$

Remark. *The best expression (smallest while sufficiently large) for the big-M constants* $M_{ij}^{I\alpha}$ *in* (17) *can easily be shown to be:* $M_{ij}^{I\alpha} = L_i^I - E_j^I + S^I(\alpha)$.

In the following, we show that, for large values of α, the linearized constraints (17) can replace the original IAF separation constraints (4) in the true model. First, we recall the definition of a *dominance relationship* between two linear constraints.

Definition 1. *Let* a, $a' \in \mathbb{R}^n$ *and* b, $b' \in \mathbb{R}$ *be given. Let* $x \in \mathcal{X} \subset \mathbb{R}^n$ *be a vector of decision variables, where* $\mathcal{X}$ *is some given subset of* $\mathbb{R}^n$*. Then we say that* $a'^T x \geq b'$ ***dominates*** $a^T x \geq b$ *with respect to* $\mathcal{X}$ *if:*

- $a'^T x \geq b' \Rightarrow a^T x \geq b$, $\forall x \in \mathcal{X}$
- *and* $\exists\, x' \in \mathcal{X}$ *such that* $a'^T x' \geq b'$ *and* $a^T x' > b$.

One can easily prove the following Lemma:

Lemma 2. *Consider a couple* $(i,j) \in \mathcal{A} \times \mathcal{A}$ *such that* $i \neq j$ *and the constraints:*

$$x_j \geq x_i + \underline{S}^I - M_{ij}^I(1-\delta_{ij}) \qquad (4_{ij})$$
$$x_j \geq x_i + S^I(\alpha) - M_{ij}^{I\alpha}(1-\delta_{ij}) \qquad (17_{ij})$$

where $M_{ij}^I = L_i^I - E_j^I + \underline{S}^I$ *and* $M_{ij}^{I\alpha} = L_i^I - E_j^I + S^I(\alpha)$.

1. *When* $\delta_{ij} = 0$, *the two big-M constraints* (4_{ij}) *and* (17_{ij}) *are redundant.*
2. *When* $\delta_{ij} = 1$, *we have the following relations between the constraints* (4_{ij}) *and* (17_{ij})*:*
   (a) *constraint* (17_{ij}) *dominates constraint* (4_{ij}) *if* $\alpha > \mathbb{P}(\gamma \leq 0)$
   (b) *constraint* (4_{ij}) *dominates constraint* (17_{ij}) *if* $\alpha < \mathbb{P}(\gamma \leq 0)$
   (c) *constraint* (4_{ij}) *is equivalent to constraint* (17_{ij}) *if* $\alpha = \mathbb{P}(\gamma \leq 0)$

   where $\boldsymbol{\gamma} \stackrel{def}{=} \boldsymbol{\omega_i} - \boldsymbol{\omega_j}$.

Proof. Proof of Lemma 2 When $\delta_{ij} = 0$, and the two big-M constants M_{ij}^I and $M_{ij}^{I\alpha}$ are respectively equal to $\left(L_i^I - E_j^I + \underline{S}^I\right)$ and $\left(L_i^I - E_j^I + S^I(\alpha)\right)$, then constraints (4_{ij}) and (17_{ij}) can both be written as: $x_j \geq x_i - L_i^I + E_j^I$. Note that this constraint is always satisfied, since $x_j \geq E_j^I$ and $x_i - L_i^I \leq 0$, as expected in this case.

When $\delta_{ij} = 1$, constraints (4_{ij}) and (17_{ij}) simplify as follows respectively:

$$x_j \geq x_i + \underline{S}^I$$
$$x_j \geq x_i + S^I(\alpha) \;=\; x_i + \underline{S}^I + F_{\boldsymbol{\gamma}}^{-1}(\alpha)$$

Remark that the relationship between the last two constraints is driven by the sign of $F_{\boldsymbol{\gamma}}^{-1}(\alpha)$. If $F_{\boldsymbol{\gamma}}^{-1}(\alpha) > 0$, which can be equivalently expressed as $\alpha > F_{\boldsymbol{\gamma}}(0)$ or $\alpha > \mathbb{P}(\gamma \leq 0)$ (using the fact that the cumulative distribution function $F_{\boldsymbol{\gamma}}$ is strictly increasing), then $x_j \geq x_i + S^I(\alpha) > x_i + \underline{S}^I$, which satisfies the definition of dominance, even in a stronger sense than introduced in Definition 1. This proves the case *2.(a)* of the Lemma. The remaining two cases *2.(b)* and *2.(c)* can be deduced easily. □

Lemma 2 is instrumental to prove the next Proposition.

Proposition 2. *Assume that* $\boldsymbol{\omega}$ *is a vector of* n *i.i.d. normal random variables. For any value of* $\alpha \geq 0.5$, *constraints* (17) *can substitute for constraints* (4) *and* (5) *in the true model.*

Proof. Proof of Proposition 2 Using Proposition 1, constraints (17) can substitute for constraints (5) in the true model. Now, consider $\boldsymbol{\omega}$ a vector of n i.i.d. random variables normally distributed with mean μ and standard deviation σ. Let us note this normal distribution $\mathcal{N}(\mu, \sigma^2)$. Then, $\boldsymbol{\gamma}$ follows the normal distribution $\mathcal{N}(0, 2\sigma^2)$ and $\mathbb{P}(\gamma \leq 0) = 0.5$. The result follows then from Lemma 2. □

Remark. *Proposition 2 may be extended to any probability distribution on an i.i.d. random vector $\boldsymbol{\omega}$ implying a symmetric distribution (with respect to zero) on the random variable $\boldsymbol{\gamma}$.*

In the remainder of this article, we make the following two assumptions under which Proposition 2 always holds:

Assumption 1. *$\boldsymbol{\omega}$ is a vector of i.i.d. normal random variables.*

Assumption 2. *The protection level α from IAF separation violations is always set to values greater than (or equal to)* 0.5.

By integrating the convex piecewise linear second-stage cost function (introduced in Subsection 3.1), and under Assumptions 1 and 2, due to Proposition 2, the true model (introduced in the beginning of Section 3) simplifies as follows:

$$\begin{aligned} \min_{\delta,\, x} \quad & \sum_{\substack{(i,j)\in\mathcal{A}^+\times\mathcal{A}^+ \\ i\neq j}} \delta_{ij} S_{ij} + \lambda\, \mathbb{E}_{\boldsymbol{\omega}}[Q(\delta, x, \omega)] \\ \text{s.t.} \quad & (2), (3), (17), (6), (7) \\ \text{where:} & \\ Q(\delta, x, \omega) = \quad & (11) \\ \text{s.t.} \quad & (12), (13), (14), (9), (10) \end{aligned} \tag{True model}$$

4 Solution methods

The two-stage stochastic program introduced in Section 3 presents two main challenges. The first challenge is to deal with the probability constraints in the first stage. In Subsection 3.2, we have shown that under the assumptions of i.i.d. normal random variables (Assumption 1) and large protection levels α (Assumption 2), the probability constraints can be equivalently written as linear constraints. The second challenge comes from the expectation term in the objective function of the first stage. Since we assume continuous random variables, the exact expression of the expectation term is a multivariate integral, often impracticable to compute. One widely-used method to approximate the expectation term in stochastic programs is to compute a sample average over a finite number of scenarios, in the context of the so-called *Sample Average Approximation* (SAA) (see e.g., Fu et al. 18) giving rise to the *SAA problem*. This problem can be seen as a one-stage mixed-integer linear problem (MILP), called the *deterministic equivalent problem*, that can be solved directly by a state-of-the-art MILP solver. Nevertheless, it is well known in the literature [6] that an efficient solution method to two-stage stochastic linear programs is the L-Shaped method that derives from Benders decomposition. In the following, we present the SAA model describing our problem and we focus on Benders reformulations of such a model.

4.1 Model with Sample Average Approximation

Let $\mathcal{S}$ denote the set of $n_{\mathcal{S}}$ equally-probable scenarios. We introduce the following scenario-specific notations for an aircraft $i \in \mathcal{A}$ and a scenario $s \in \mathcal{S}$: ω_i^s, y_i^s, z_i^{s-}, z_i^{s+} and z_i^{s++}. For

a given scenario $s \in \mathcal{S}$, the corresponding vector notations are naturally deduced: $\omega^s = (\omega_1^s, \omega_2^s \dots \omega_n^s)^T$, $y^s = (y_1^s, y_2^s \dots y_n^s)^T$, $z^{s-} = \left(z_1^{s-}, z_2^{s-} \dots z_n^{s-}\right)^T$, $z^{s+} = \left(z_1^{s+}, z_2^{s+} \dots z_n^{s+}\right)^T$ and $z^{s++} = \left(z_1^{s++}, z_2^{s++} \dots z_n^{s++}\right)^T$. According to the SAA method, for a sufficiently large number of scenarios, $n_{\mathcal{S}}$, the objective function (1) can be replaced by:

$$\min_{\delta,\, x} \quad \sum_{\substack{(i,j) \in \mathcal{A}^+ \times \mathcal{A}^+ \\ i \neq j}} \delta_{ij} S_{ij} + \lambda \sum_{s \in \mathcal{S}} \frac{1}{n_{\mathcal{S}}} Q\left(\delta, x, \omega^s\right) \tag{18}$$

Replacing (1) by (18) in the true model leads to the so-called *SAA model*. The SAA method relies on the uniform law of large numbers to prove that, as $n_{\mathcal{S}} \to \infty$, the SAA-model optimal objective value converges almost surely to the true-model optimal objective value [39]. Using the linearized second-stage objective function proposed in Subsection 3.1, we can express the optimal value of the second-stage problem corresponding to a given scenario $s \in \mathcal{S}$, $Q\left(\delta, x, \omega^s\right)$, (as appearing in (18)) as follows:

$$Q\left(\delta, x, \omega^s\right) = \min_{\substack{y^s,\, z^{s-} \\ z^{s+}, z^{s++}}} \sum_{i \in \mathcal{A}} \left(c_1 z_i^{s-} + c_2 z_i^{s+} + c_3 z_i^{s++}\right) \tag{19}$$

$$-y_i^s + z_i^{s++} + z_i^{s+} - z_i^{s-} = -x_i - \omega_i^s - \hat{V}_i \qquad i \in \mathcal{A} \quad (\beta_i^s) \tag{20}$$

$$-y_i^s \geq -x_i - \omega_i^s - \overline{V_i} \qquad i \in \mathcal{A} \quad (\sigma_i^s) \tag{21}$$

$$y_i^s \geq x_i + \omega_i^s + \underline{V_i} \qquad i \in \mathcal{A} \quad (\rho_i^s) \tag{22}$$

$$y_j^s - y_i^s \geq S_{ij} - M_{ij}^s(1 - \delta_{ij}) \qquad (i,j) \in \mathcal{A} \times \mathcal{A}, i \neq j \quad (\pi_{ij}^s) \tag{23}$$

$$-z_i^{s+} \geq -x_i - \omega_i^s - V_i^{\text{med}} \qquad i \in \mathcal{A} \quad (\mu_i^s) \tag{24}$$

$$z_i^{s++},\ z_i^{s+},\ z_i^{s-} \geq 0 \qquad i \in \mathcal{A} \tag{25}$$

where dual variables corresponding to constraints (20) to (24) are shown between parenthesis. Recall that the big-M constant M_{ij}^s can be set to $\left(L_i^I + \omega_i^s + \overline{V_i}\right) - \left(E_j^I + \omega_j^s + \underline{V_j}\right) + S_{ij}$.

The SAA model basically describes a deterministic (possibly large-scale) MILP: the deterministic equivalent problem. The extended formulation of the deterministic equivalent problem is:

$$\begin{aligned} \min_{\substack{\delta,\, x \\ y^s,\, z^{s-} \\ z^{s+},\, z^{s++}}} \quad & \sum_{\substack{(i,j) \in \mathcal{A}^+ \times \mathcal{A}^+ \\ i \neq j}} \delta_{ij} S_{ij} + \lambda \sum_{s \in \mathcal{S}} \frac{1}{n_{\mathcal{S}}} \sum_{i \in \mathcal{A}} \left(c_1 z_i^{s-} + c_2 z_i^{s+} + c_3 z_i^{s++}\right) \\ \text{s.t.} \quad & (2), (3), (17), (6), (7) \\ & (20), (21), (22), (23), (24), (25) \end{aligned} \tag{Determ. Eq.}$$

The deterministic equivalent problem can be directly solved using a state-of-the-art MILP solver. One weakness of the extended formulation is that the problem size can become very large as the number of scenarios increases. For example, for $n = 10$ aircraft and $n_{\mathcal{S}} = 500$ scenarios, there are $20,000$ second-stage variables.

We remark that if the first-stage variables, x and δ, are fixed, then the second stage turns to be $n_{\mathcal{S}}$ separate linear programs that are straightforward to solve. This property allows us to reformulate our SAA problem using Benders decomposition, as presented in Subsection 4.2.

4.2 Benders reformulations

Using Benders decomposition [6, 38], we can decompose our two-stage stochastic integer problem described by the SAA model into a master problem, called *Benders master problem*, and one or many separate subproblem(s), called *Benders subproblem(s)*, corresponding to the second-stage problems. According to the level of aggregation chosen for the Benders subproblem(s), we can propose different Benders reformulations of our SAA model.

When the second-stage problems are completely aggregated, we are left with one Benders subproblem. Then, only one cut can be generated at each iteration. We call this reformulation: *simple-cut* Benders reformulation. When the second-stage problems are completely disaggregated (not aggregated at all), we have one Benders subproblem for each scenario. Accordingly, at most one cut per scenario can be generated by iteration. Hence, one can add up to $n_{\mathcal{S}}$ cuts at each iteration. We call this reformulation *multi-cut* Benders reformulation. When the second-stage problems are aggregated into different subsets, where each subset corresponds to multiple scenarios, we say that the second-stage problems are *partially aggregated*. This yields one Benders subproblem for each subset of scenarios, called a cluster of scenarios. In this case, at most one cut per cluster can be generated at each iteration. We call this reformulation: *partially-aggregated-cut* Benders reformulation. In the following, we only present the partially-aggregated-cut Benders reformulation, since it encompasses the other two versions above, that represent the two extreme cases.

Let $\mathcal{C} = \{c_1, c_2, \dots c_K\}$ be a partition of $\mathcal{S}$, where each c_i $(i = 1, 2, \dots, K)$ is a (non-empty) subset of $\mathcal{S}$, referred to as a *cluster of scenarios*. Remark that the simple-cut version corresponds to $K = 1$, while the multi-cut version corresponds to $K = n_{\mathcal{S}}$. The second-stage problems corresponding to scenarios belonging to a same cluster are aggregated to form a single Benders subproblem. Hence, there are K Benders subproblems and, consequently, K additional optimization variables $\nu^c\,(c \in \mathcal{C})$, are introduced to approximate the expected second-stage cost. Following the standard Benders' decomposition methodology (*e.g.*, Rahmaniani et al. 38), the initial Benders master problem, in the partially-aggregated-cut version, is therefore:

$$\min_{\delta,\, x,\, \nu} \quad \sum_{\substack{(i,j)\in\mathcal{A}^+\times\mathcal{A}^+ \\ i\neq j}} \delta_{ij} S_{ij} + \lambda \sum_{c\in\mathcal{C}} \nu^c \tag{26}$$

$$\text{s.t.} \quad \text{first-stage constraints:} \quad (2), (3), (17), (6), (7)$$

$$\nu^c \geq 0 \qquad\qquad c \in \mathcal{C} \tag{27}$$

where $\nu = \left(\nu^1, \nu^2, \dots, \nu^K\right)^T$. Constraints (27) are obvious bound constraints on variables ν^c that strengthen the standard Benders reformulation and can be included directly in the initial Benders master problem.

Consider a (non-empty) cluster of scenarios $c \in \mathcal{C}$. The Benders subproblem corresponding to cluster c consists of n_c separate scenario subproblems that can be solved separately. The results of these n_c scenario subproblems are aggregated to compute the results of the Benders subproblem associated to cluster c (objective-function value, dual-variables values, etc). Let $\mathcal{R}^c$ and $\mathcal{T}^c$ be respectively the set of extreme rays and the set of extreme points of the Benders-dual-subproblem polyhedron corresponding to cluster c. Benders feasibility and optimality cuts for the partially-aggregated-cut version of our

SAA model are given by constraints (28) and (29) respectively:

$$0 \geq \sum_{s \in c} \left[\frac{1}{n_{\mathcal{S}}} \sum_{i \in \mathcal{A}} \left[\left(-x_i - \omega_i^s - \hat{V}_i \right) \beta_i^{sr} + \left(-x_i - \omega_i^s - \overline{V_i} \right) \sigma_i^{sr} + \left(x_i + \omega_i^s + \underline{V_i} \right) \rho_i^{sr} \right.\right.$$
$$\left.\left. + \left(-x_i - \omega_i^s - V_i^{\text{med}} \right) \mu_i^{sr} \right] + \frac{1}{n_{\mathcal{S}}} \sum_{\substack{(i,j) \in \mathcal{A} \times \mathcal{A} \\ i \neq j}} \left(S_{ij} - M_{ij}^s (1 - \delta_{ij}) \right) \pi_{ij}^{sr} \right]$$
$$r \in \mathcal{R}^c,\ c \in \mathcal{C} \tag{28}$$

$$\nu^c \geq \sum_{s \in c} \left[\frac{1}{n_{\mathcal{S}}} \sum_{i \in \mathcal{A}} \left[\left(-x_i - \omega_i^s - \hat{V}_i \right) \beta_i^{st} + \left(-x_i - \omega_i^s - \overline{V_i} \right) \sigma_i^{st} + \left(x_i + \omega_i^s + \underline{V_i} \right) \rho_i^{st} \right.\right.$$
$$\left.\left. + \left(-x_i - \omega_i^s - V_i^{\text{med}} \right) \mu_i^{st} \right] + \frac{1}{n_{\mathcal{S}}} \sum_{\substack{(i,j) \in \mathcal{A} \times \mathcal{A} \\ i \neq j}} \left(S_{ij} - M_{ij}^s (1 - \delta_{ij}) \right) \pi_{ij}^{st} \right]$$
$$t \in \mathcal{T}^c,\ c \in \mathcal{C} \tag{29}$$

where we use β_i^s, σ_i^s, ρ_i^s, π_{ij}^s, and μ_i^s to denote the dual variables associated to constraints (20) to (24) respectively, to which we add the index r or t depending upon whether we refer to an extreme ray $r \in \mathcal{R}^c$, or to an extreme point $t \in \mathcal{T}^c$.

Notes on the size of the models

The first-stage problem involves n continuous variables, $n\,(n+1)$ binary variables, and $n\,(n+1)+2$ constraints (apart from the $2n$ bound constraints on x). Regarding the second stage, one scenario subproblem involves $4n$ continuous variables, and $n\,(n+3)$ constraints (apart from the $3n$ bound constraints on z^-,z^+, and z^{++}). For $n_{\mathcal{S}}$ scenarios, the model of the deterministic equivalent problem, called the *extended formulation*, requires $n\,(4n_{\mathcal{S}}+1)$ continuous variables, $n\,(n+1)$ binary variables, and $n\,(n+3)\,n_{\mathcal{S}}+n\,(n+1)+2$ constraints.

Regardless of the degree of aggregation, Benders reformulations comprise the same number of binary variables ($n\,(n+1)$) as the extended formulation, since these variables only appear in the first stage. In terms of continuous variables, the three Benders reformulations differ. The general partially-aggregated cut version has $n+K$ continuous variables, where $1 \leq K \leq n_{\mathcal{S}}$. The simple-cut version has $n+1$ continuous variables. The multi-cut version involves $n+n_{\mathcal{S}}$ continuous variables.

Table 4 summarizes the model sizes according to the different formulations. Table 5 gives numerical examples of model sizes for a 10-aircraft instance and two numbers of scenarios, $n_{\mathcal{S}} = 100$ and 500. The partially-aggregated-cut Benders reformulation version in Table 5, denoted "5-aggregated-cut Benders", involves $K = \dfrac{n_{\mathcal{S}}}{5}$ clusters.

4.3 Implementing Benders decomposition

In the context of two-stage stochastic programming, Benders reformulation has the merit of reducing effectively the number of continuous variables, as shown in the previous subsection. However, standard implementations of Benders decomposition may fail to reach

Table 4: Model sizes for different formulations.

Formulation	# bin. var.	# cont. var.	# constraints
Determ. Eq.	$n\,(n+1)$	$n\,(4n_{\mathcal{S}}+1)$	$n\,(n+3)\,n_{\mathcal{S}}+n\,(n+1)+2$
Multi-cut Benders	$n\,(n+1)$	$n+n_{\mathcal{S}}$	$n\,(n+1)+2^*$
Partially-aggregated-cut Benders	$n\,(n+1)$	$n+K$	$n\,(n+1)+2^*$

* without Benders cuts. In fact, the initial Benders master problem starts with no Benders cuts. Then, dynamically, such cuts are generated and added to the Benders master problem.

Table 5: Model sizes for $n = 10$ and different numbers of scenarios $n_{\mathcal{S}}$.

	Formulation	# bin. var.	# cont. var.	# constraints
$n_{\mathcal{S}} = 100$	Determ. Eq.	110	4,010	13,112
	Multi-cut Benders	110	110	112
	5-aggregated-cut Benders	110	30	112
$n_{\mathcal{S}} = 500$	Determ. Eq.	110	20,010	65,112
	Multi-cut Benders	110	510	112
	5-aggregated-cut Benders	110	110	112

the expected performance (in terms of computation time) or even to outperform state-of-the-art MILP solvers solving the deterministic equivalent problem by Branch-and-Cut. For that reason, numerous accelerating techniques have been proposed in the literature to boost the performance of Benders decomposition. For an extensive literature review on Benders decomposition and accelerating techniques, please refer to [38].

Modern implementations of Benders decomposition rely on a single search tree, where Benders subproblems are solved at every integer-solution node in the Branch-and-Cut tree (this can be done through the cut callback functionality of MILP solvers), unlike traditional implementations where the relaxed Benders master problem is solved to optimality at each iteration before solving Benders subproblem. This variant of Benders decomposition is often called Branch-and-Benders-Cut. Since version 12.7, IBM ILOG CPLEX implements an automatic Benders decomposition as a Branch-and-Benders-Cut following a two-phase solution scheme and involving an in-out cut loop strategy (24). In the following, we present these two acceleration techniques.

Two-phase solution scheme

[33] propose to apply Benders decomposition in two phases. In the first phase, the linear relaxation of the Benders master problem is solved using Benders decomposition. The authors show that Benders cuts generated during this first phase are valid for the original MILP problem. In the second phase, integrality constraints are reintroduced and Benders decomposition is relaunched with, hopefully, a tighter Benders master problem. Such an approach can be easily adapted in a Branch-and-Benders-Cut variant by implementing a traditional Benders decomposition only for the first phase, where Benders cuts are added explicitly as constraints to the linear relaxation of the Benders master problem, as in a traditional fractional cutting-plane method.

In-out cut loop strategy

According to [17], slow convergence of Benders decomposition is very likely caused by the cut loop's standard strategy at the root node, also known as Kelley's loop (28), which consists of separating the current solution by adding, to the master problem, Benders cuts violated by the current solution, then re-optimizing the updated Benders master problem. A more recent strategy, called "in-out search" is introduced in [15] and applied to Benders decomposition to solve efficiently facility location problems in [16, 17]. In [16], the "in-out" strategy is applied to generate initial cuts before the Branch-and-Benders-Cut. Similarly, in [17], this strategy is applied to stabilize the cut loop in the root node. Mainly, the "in-out" strategy requires two ingredients: an "*out point*" and an "*in point.*" The "out point" corresponds to the current solution (to the relaxed Benders master problem), while the "in point", also called the *stabilizing* point, lies in the interior of the feasible domain of the linear relaxation of the original MILP (in our case, the linear relaxation of the deterministic equivalent problem). Instead of separating the "out point" as in a standard Kelley's strategy, an "*intermediate point*" is built as a convex combination of the "out point" and the "in point", and then separated. The generated Benders cut is added, and the updated Benders master problem is re-solved. Iteratively, the "in point" is updated and a new "intermediate point" is built and separated. After a fixed number of trials, if the current solution ("out point") is not cut, then the "in-out search" is aborted, and the standard Kelley's cut loop strategy is applied to effectively cut the current solution. The "in-out search" is then resumed with a new "out point" and the trial counter is set to zero.

CPLEX automatic Benders decomposition implements the above acceleration techniques (among others, see e.g., [30]). Furthermore, this solver proposes a *Benders strategy* parameter that guides the MILP partitioning into a master Benders problem and one or many Benders subproblems. In the scope of this study, we use two such strategies: *FULL* and *USER*. According to CPLEX documentation [25], when the Benders strategy parameter is set to FULL, CPLEX automatically decomposes a given MILP by putting all integer variables in the Benders master problem and all continuous variables in a Benders subproblem. Then, CPLEX tries to refine the decomposition of the Benders subproblem. When the Benders strategy parameter is set to USER, CPLEX decomposes the given MILP according to the partition specified by the user by means of variable and constraint *annotations*. Remark that this user-specified partitioning feature is, in fact, crucial for use cases where the most interesting partitioning does not correspond to the one CPLEX automatically applies under the strategy FULL.

Given that CPLEX proposes an efficient automatic Benders decomposition, and more importantly, that it allows user-specified partitioning, we exploit this solver to test the Benders reformulation with multiple levels of subproblem aggregation, proposed in Subsection 4.2.

5 Computational study

This section aims firstly at showing the viability of our proposed model. The benefit of taking uncertainty into account is highlighted through the *value of stochastic solution* metric [6]. Secondly, we compare the different solution methods presented in Section 4, in terms of computational time. The remaining of this section is organized as follows.

Instances and parameter values are presented in Subsection 5.1. The methodology used to determine an appropriate number of scenarios, as well as our main computational results are presented in Subsection 5.2. In Subsection 5.3, we discuss the viability of our approach through an analysis of the value of stochastic solution under different test settings. A comparison of the performance of four solution methods is presented in Subsection 5.4. All results are obtained on a Linux platform with 8 x 2.66 GHz Xeon processors, 32 GB of RAM, and using CPLEX version 12.7.1.

5.1 Instances and parameter values

We construct ten instances from real arrival data corresponding to Paris CDG airport, May 15, 2015, covering a one-hour time frame (from 5:59 AM to 6:59 AM). During this time frame, 30 aircraft planned to cross three different IAFs (named MOPAR, LORNI, and OKIPA) and landed on runway 27 R afterwards. In order to match with our problem statement where we consider a single IAF, the three IAFs are merged, but no changes are made on IAF planned times. We construct five planned schedules as follows. The first planned schedule, named 10_559_618, corresponds to the first ten aircraft, and spans from 5:59 AM to 6:18 AM. We construct the second planned schedule, named 10_607_623, by shifting the first five aircraft and then considering the next 10 aircraft. Accordingly, the planned schedule 10_607_623 is made of the 6^{th} through the 15^{th} aircraft (in the 30-aircraft raw schedule), and spans from 6:07 AM to 6:23 AM. The following planned schedules, named 10_619_634, 10_624_640, and 10_634_659 respectively, are constructed using the same shifting procedure, while the number of aircraft per instance is kept fixed ($n = 10$). The five planned schedules can be visualized in Figure 2. Two problem instances are constructed from each planned schedule according to the IAF time window width (narrow or wide). The main characteristics of the ten instances that constitute our test bed are shown in Table 6. In the following, we present in more details the parameter values that are related to: IAF time windows, uncertainty, protection level against IAF separation violation (α), IAF and final-approach separations, and landing time windows.

Table 6: Test bed summary description.

Instance Id	time span (min)	n by original IAF			WTC mix		TW^I
		MOPAR	LORNI	OKIPA	H%	M%	
10_559_618_N	19′	6	2	2	60	40	Narrow
10_559_618_W	19′	6	2	2	60	40	Wide
10_607_623_N	16′	5	4	1	40	60	Narrow
10_607_623_W	16′	5	4	1	40	60	Wide
10_619_634_N	15′	5	5	0	30	70	Narrow
10_619_634_W	15′	5	5	0	30	70	Wide
10_624_640_N	16′	4	6	0	30	70	Narrow
10_624_640_W	16′	4	6	0	30	70	Wide
10_634_659_N	25′	3	7	0	30	70	Narrow
10_634_659_W	25′	3	7	0	30	70	Wide

WTC: Wake-turbulence category; H: Heavy; M: Medium; TW^I: IAF time-window width.

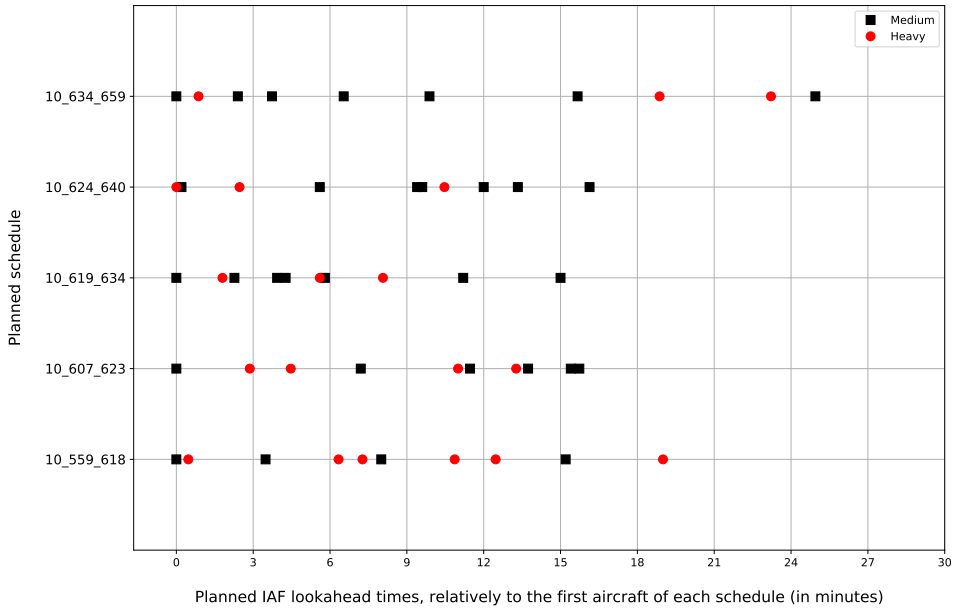


Figure 2: Visualization of the five planned schedules.

IAF time windows: We consider two possibilities for the IAF time-window width: *narrow* and *wide* time windows yielding two different types of instances. In the narrow instances, the aircraft IAF time window is given by $E_i^I = P_i^I - 1$ min and $L_i^I = P_i^I + 5$ min, where P_i^I is the planned IAF time for aircraft $i \in \mathcal{A}$. In the wide instances, the IAF time window is given by $E_i^I = P_i^I - 1$ min and $L_i^I = P_i^I + 15$ min. These IAF time window widths are motivated as in [29]. A one-minute advance can be achieved by speeding up the aircraft, while 5 minutes of delay can be absorbed by speed reduction, both over 300 nautical miles, according to the concept of E-AMAN. Note that speed reduction is known as a "linear holding" technique, that is considered as a fuel-consumption-friendly control technique. Larger delays require different air traffic control techniques, like path stretching for example.

Uncertainty: The random variables $\boldsymbol{\omega_i}$'s are i.i.d. following the normal distribution $\mathcal{N}(0, \sigma^2)$, with mean zero and standard deviation $\sigma = 30$ seconds. According to this value of standard deviation, most of the time (with probability greater than 0.99), the actual arrival of an aircraft to the IAF will not deviate more than $\pm 3\sigma = \pm 90$ seconds from its target time. The assumption of independent and identically distributed random variables can be supported by the fact that we consider a relatively small number of aircraft ($n = 10$) coming from different directions.

Protection level against IAF separation violation: In compliance with Assumption 2 (see Subsection 3.2), we only study values of α greater than or equal to 50%. Three values for the protection level α are considered: 50%, 90%, and 95%. The lowest value of α corresponds to the situation where an airborne conflict near the IAF between two given aircraft i and j is likely to happen at most 50% of the time (and consequently air traffic controllers must intervene to solve this conflict). The largest value of α (95%) corresponds

Table 7: Rounded buffered separation $S^I(\alpha)$ (in seconds) for uncertainty $\sigma = 30$ sec

α	50%	90%	95%
$S^I(\alpha)$	72	126	142

to a rare IAF separation violation between two given aircraft i and j (at most 5% of the time).

Separations: Final-approach time separations (S_{ij}), and minimum separation over the IAF ($\underline{S}^I$) are as indicated in Section 2. The buffered IAF separation $S^I(\alpha)$ (defined in Subsection 3.2) depends on the value of the protection level α, as shown in Table 7.

Landing time windows: Each landing time window is piecewise defined over three time intervals related to the unconstrained landing time of each aircraft according to the form of the second-stage objective function introduced in Subsection 3.1. In our tests, deviation costs incurred within the first time segment: [−1 min ; 0 min] are proportional to the weight $-c_1$. Delays within [0 min ; +4 min] yield costs proportional to the weight c_2. Finally, delays within [+4 min ; +19 min] are proportional to the weight c_3.

Second-stage time-deviation weights: Values of second-stage time-deviation weights, c_1, c_2, and c_3, should reflect the amount of workload required from air traffic controllers to implement each category of time deviations displayed by AMAN (time to gain up to 1 minute, time to lose up to 4 minutes, and time to lose greater than 4 minutes). Achieving a delay smaller than 4 minutes in the terminal area is common, and its workload impact cost per second can be set to the normalized value $c_2 = 1.0$. Since time advance is almost costless, c_1 should be set to a smaller value. We choose $c_1 = 0.5$. Finally, delaying an aircraft by more than 4 minutes should be avoided as much as possible, since it requires resorting to holding patterns. The corresponding workload cost per second, c_3, must be relatively high compared with c_1 and c_2. We set $c_3 = 4.0$. Note that the studied values ($c_1 = 0.5, c_2 = 1.0, c_3 = 4.0$) satisfy $0 < c_1 < c_2 < c_3$, so that the resulting functions, f_i's, for $i \in \mathcal{A}$, are convex piecewise linear.

Weighting parameter λ: We keep the weighting parameter λ fixed to the value 1 in our computational study, except in Subsection 5.3 where we study the trade-off between the first-stage and the expected second-stage objectives, and its effect on the benefit of two-stage stochastic programming to tackle our problem.

Common features across the ten instances are summarized in Table 8.

5.2 Determining an appropriate number of scenarios

In order to verify whether a given number of scenarios is sufficiently large, we use the *out-of-sample validation* technique that consists in computing a *validation score* and a *validation gap* of an SAA problem's solution using a large sample of scenarios called the *validation set*. By definition, the validation set should not contain any of the scenarios used during optimization. In our computational study, the validation set is sampled using a seed different from all seeds used to generate scenario sets for optimization. Let $\mathcal{S}$ and $\mathcal{S}^v$ be two sets of scenarios, where $\mathcal{S}$ is used for optimization and $\mathcal{S}^v$ is used for validation. By definition, the validation set $\mathcal{S}^v$ should contain many more scenarios than

Table 8: Instances common features.

Total number of aircraft	$n = 10$
IAF time windows* (TW^I)	Narrow: $[-1$ min ; $+5$ min$]$ Wide : $[-1$ min ; $+15$ min$]$
Uncertainty standard deviation	$\sigma = 30$ sec
Landing time window	$[-1$ min ; $+4$ min ; $+19$ min$]$
Second-stage unitary impact costs	$c_1 = 0.5$ $c_2 = 1.0$ $c_3 = 4.0$

* Narrow/Wide yields two different instances

the optimization set $\mathcal{S}$, i.e., $n_{\mathcal{S}^v} \gg n_{\mathcal{S}}$. Let $(\delta^\star_{\mathcal{S}}, x^\star_{\mathcal{S}})$ be a (first-stage) optimal solution for the two-stage stochastic program obtained with the set of scenarios $\mathcal{S}$, noted SP($\mathcal{S}$). Let $v^\star_{\mathcal{S}}$ be the optimal objective-function value of SP($\mathcal{S}$). Let SP($\mathcal{S}^v$) be the two-stage stochastic program obtained with the set of scenarios $\mathcal{S}^v$. The validation score of $(\delta^\star_{\mathcal{S}}, x^\star_{\mathcal{S}})$, noted $v_{\mathcal{S}^v}$, is defined as the objective-function value of SP($\mathcal{S}^v$) corresponding to the solution $(\delta^\star_{\mathcal{S}}, x^\star_{\mathcal{S}})$. Remark, here, that we implicitly assume that $(\delta^\star_{\mathcal{S}}, x^\star_{\mathcal{S}})$ is feasible for SP($\mathcal{S}^v$), although theoretically infeasibility may arise for some scenarios since the recourse is not relatively complete. The *validation gap* corresponding to the solution $(\delta^\star_{\mathcal{S}}, x^\star_{\mathcal{S}})$ is computed as the relative difference between the SAA-problem objective-function optimal value $v^\star_{\mathcal{S}}$, and the validation score $v_{\mathcal{S}^v}$:

$$\text{Validation gap} \quad \stackrel{\text{def}}{=} \quad \frac{v^\star_{\mathcal{S}} - v_{\mathcal{S}^v}}{v_{\mathcal{S}^v}} \times 100$$

The validation gap is a normalized quantity that helps to estimate whether a number of scenarios $n_{\mathcal{S}}$ "approximates well enough" the reference set of scenarios $\mathcal{S}^v$, also called the "reference tree" as mentioned in [27]. In order to find an appropriate number of scenarios, $n_{\mathcal{S}}$, we propose to solve the deterministic equivalent problem using CPLEX for increasing values of $n_{\mathcal{S}}$ ranging from 10 to 500. For each number of scenarios, 10 replications of the SAA problem are constructed and solved. For each replication, a validation score is computed using a validation set of $10,000$ scenarios, and a validation gap is deduced. For a given number of scenarios, an *average validation gap* is computed (over the validation gaps of the 10 replications). As computation time increases rapidly with the number of scenarios, we are content with an appropriate number of scenarios chosen to be the smallest number of scenarios that yields an absolute average validation gap smaller than 0.15%, and a corresponding 95%-confidence interval radius less than 0.15%.

Extensive results are given in the appendix. Table 9 summarizes the appropriate number of scenarios $n^*_{\mathcal{S}}$ for each instance, and recalls the number of different sequences "# Seq." corresponding to each $n^*_{\mathcal{S}}$ over the 10 replications.

Effect of α on the appropriate number of scenarios

In Table 9, we observe that the appropriate number of scenarios, $n^*_{\mathcal{S}}$, generally decreases, if not maintained, when increasing α. For $\alpha = 50\%$, the appropriate numbers of scenarios range from 100 to 500 scenarios, while for $\alpha = 95\%$, 50 to 200 scenarios are sufficient to approximate satisfactorily the reference problem (i.e., the problem with the validation set). We conclude that buffering the IAF separation (by increasing α) not only simplifies the problem from a combinatorial point of view (by decreasing the number of feasible

solutions, and thereby the computational time) but also helps to limit the number of scenarios needed to estimate appropriately the expected second-stage cost. However, for dense planned schedules (like 10_607_623 and 10_619_634), enforcing large IAF separations may cause some instances to become infeasible.

Effect of IAF time window width on the appropriate number of scenarios

We remark that the appropriate number of scenarios for any given instance with wide IAF time windows is always smaller than or equal to the appropriate number of scenarios for its counterpart with narrow IAF time windows.

Effect of IAF time window width on the optimal objective-function value

In line with Khassiba et al. 29, extensive results (see the appendix) show that optimization problems with narrow IAF time windows are easier to solve as more time windows are likely to be disjoint, reducing the number of feasible sequences. This may, in turn, reduce the solution space for the (NP-hard) first-stage problem. On the other hand, optimal objective-function values are slightly smaller with wide IAF time windows as the problem features more feasible candidate solutions of a potentially better quality. As a drawback, the problem is more combinatorial and therefore harder to solve. From an operational viewpoint, if more degrees of freedom are available to schedule flights on the IAF, then uncertainty can be better absorbed (i.e., less last-minute control workload to sequence landings is needed), and better sequences can be built (i.e., sequences that yield higher landing rate).

Note on the stability of the problem

Based on the extensive results reported in the appendix, when increasing the number of scenarios $n_{\mathcal{S}}$ for a given instance, the average objective-function optimal value first increases, then stagnates more or less early depending on the instance. Also, the variance of the average objective-function optimal value decreases sharply. On the contrary, the average validation score first decreases then stagnates more or less early depending on the instance, while its variance decreases rapidly. These remarks hold for the average validation gap. This illustrates the fact that an SAA-problem optimal value (estimated by the average objective-function optimal value) is negatively biased, and that the bias decreases when increasing the number of scenarios [39].

As for the solution stability of a given instance, we remark that the number of different sequences across the optimal solutions of the 10 replications (recall that only one solution is retained for each replication) decreases when increasing the number of scenarios $n_{\mathcal{S}}$. Nevertheless, in many test cases, even for $n_{\mathcal{S}} = 500$, there is still many different optimal sequences across the replications. This may be due to the fact that there is not a unique optimal sequence.

With respect to the IAF target times, that represent the continuous part of our mixed-integer problem's solution, instability across the replications remains, even when $n_{\mathcal{S}}$ increases. This may be explained partially by the fact that the sequences are not stable themselves. Another explanation may come from the fact that the convergence rate of the continuous components of the solution is governed by the "rather-slow" $\mathcal{O}\left(n_{\mathcal{S}}^{-1/2}\right)$ SAA-convergence rate [39].

Table 9: Appropriate number of scenarios, $n^*_{\mathcal{S}}$, for each instance.

	α					
	50%		90%		95%	
Instance Id	$n^*_{\mathcal{S}}$	# Seq.	$n^*_{\mathcal{S}}$	# Seq.	$n^*_{\mathcal{S}}$	# Seq.
10_559_618_N	200	2	100	2	50	1
10_559_618_W	100	7	100	10	50	10
10_607_623_N	500	7	500	2	NA	NA
10_607_623_W	100	2	100	9	100	9
10_619_634_N	200	1	NA	NA	NA	NA
10_619_634_W	100	5	100	6	100	4
10_624_640_N	200	6	100	6	200	5
10_624_640_W	100	9	100	10	100	10
10_634_659_N	100	2	100	2	50	2
10_634_659_W	100	10	100	3	50	2

In the remainder of this computational study, we keep the number of scenarios fixed to $n^*_{\mathcal{S}}$ for each instance, as shown in Table 9.

In the next Subsection, we quantify the benefit from solving a two-stage stochastic program over a deterministic-optimization approach. We also study the characteristics of some stochastic solutions, and we compare them with their deterministic counterparts.

5.3 Value of the stochastic solution

The *value of the stochastic solution* (VSS) expresses the benefit of solving a two-stage stochastic problem, where uncertain data are assumed to follow known random distributions, over solving a deterministic problem, where uncertain data are reduced to their average values.

Let $(\delta_{\text{SP}}, x_{\text{SP}})$ be a retained solution of the two-stage stochastic problem (SP), and $v^\star_{\text{SP}}$ its validation score over a given validation set of scenarios $\mathcal{S}^v$, assuming that $(\delta_{\text{SP}}, x_{\text{SP}})$ is feasible for all scenarios in $\mathcal{S}^v$. We shall refer to $(\delta_{\text{SP}}, x_{\text{SP}})$ as the stochastic solution. Let us define the *expected-value problem* (EP) as the two-stage "stochastic" problem, where the only second-stage scenario considered is the average (or expected-value) scenario. Remark that, reducing uncertain problem data to their average values corresponds to a full deterministic approach that completely overlooks uncertainty. Let $(\delta^\star_{\text{EP}}, x^\star_{\text{EP}})$ be an optimal solution of (EP), and $v^\star_{\text{EP}}$ be its validation score over the validation set $\mathcal{S}^v$, assuming that $(\delta^\star_{\text{EP}}, x^\star_{\text{EP}})$ is feasible for all scenarios in $\mathcal{S}^v$. We shall refer to $(\delta^\star_{\text{EP}}, x^\star_{\text{EP}})$ as the deterministic solution.

To quantify the advantages of the stochastic solution over its deterministic counterpart, we may rely on the validation score as an expected quality metric. In our context, the validation score $v^\star_{\text{SP}}$ expresses the expected quality of the stochastic solution, while the validation score $v^\star_{\text{EP}}$ estimates the expected quality of the deterministic solution over the validation set $\mathcal{S}^v$. Accordingly, we define the *relative VSS* as the relative difference between the validation scores of the stochastic and the deterministic solutions as follows:

$$\text{VSS}(\%) \quad \stackrel{\text{def}}{=} \quad 100 \times \frac{v^\star_{\text{EP}} - v^\star_{\text{SP}}}{v^\star_{\text{EP}}}$$

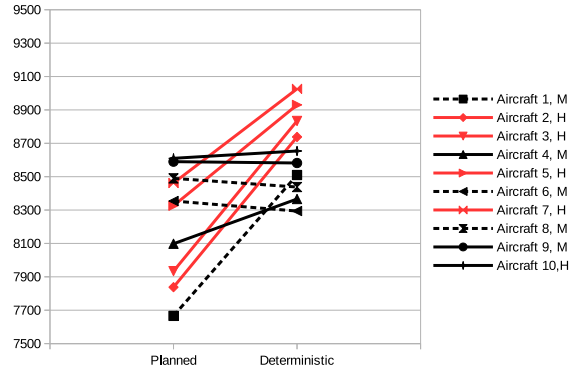
Note that, since we solve a minimization problem, and since the stochastic solution is expected to be better than its deterministic counterpart (i.e., $v^\star_{\text{SP}} \leq v^\star_{\text{EP}}$), we expect the relative VSS to be non-negative.

Relative VSS for each instance for different values of protection level α are reported in Table 11. The difference in length between the stochastic and the deterministic sequences, noted ΔSeq., is also reported. We remark that the highest relative values of the stochastic solutions (10.21% and 10.79%) correspond to instances 10_607_623_W and 10_619_634_W with the test parameter $\alpha = 50\%$ and $\lambda = 1$. These instances have the two most dense planned schedules and both feature wide IAF time windows. However, as the protection level against IAF separation loss α increases, VSS sharply decreases for all instances, and almost vanishes for instance 10_624_640 with $\alpha = 95\%$ (VSS = 0.05%). Recall that for high values of α (typically 90% or 95%), the IAF separation is enlarged, as shown in Table 7. Such buffered separations contribute to hedge against uncertainty as follows. If target IAF times are spaced out more than the minimal requirement $\underline{S}^I$, the actual IAF times are expected to be less disrupted when the uncertainty is revealed. Therefore, the recourse cost to restore the target sequence while not deviating much from the unconstrained landing times, is expected to be smaller, yielding a small expected second-stage cost.

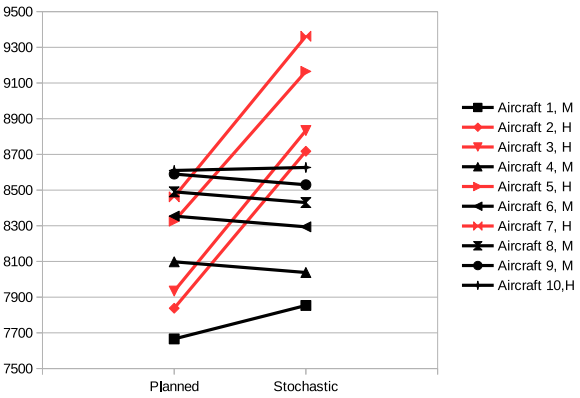
We conclude that the benefit of solving a two-stage stochastic program is more prominent in the situation of high-density air traffic, and when aircraft and controllers have many degrees of freedom for re-scheduling. Also, hedging against uncertainty using buffered separations may lead to a better performance of a deterministic scheduling policy.

In order to study further the features of the stochastic solutions with the highest VSS, we display in Figures 3 and 4 the planned, deterministic, and stochastic IAF schedule for instances 10_607_623_W and 10_619_634_W with the test parameter $\alpha = 50\%$. In these figures, high-turbulence-category-aircraft time plots are shown in red color, while those of medium-turbulence-category aircraft are in black color. Also, when an aircraft changes its relative position in the subsequence of aircraft of the same turbulence category between the planned and the studied IAF schedule (stochastic or deterministic), we use dashed lines to link that aircraft time plots in the two schedules. For example, in instance 10_607_623_W, aircraft 1 is from a medium-turbulence category and was planned first to cross the IAF. In the deterministic schedule, this aircraft is scheduled fourth behind three medium aircraft (4, 6 and 8). Accordingly, the two time plots of aircraft 1 are linked with a dashed line in Figure 3a. For aircraft that do not change their relative position, a continuous line is drawn.

Firstly, we remark that in both stochastic and deterministic solutions aircraft are sequenced according to the rule "lighter aircraft first". Indeed, in both instances, all heavy-turbulence-category aircraft are delayed and put at the end of the sequence. This sequencing yields a minimum first-stage cost (i.e., minimum sequence length in terms of final-approach separations). However, IAF target times in stochastic schedules are more spaced out than in deterministic schedules, although only the minimum IAF separation $\underline{S}^I$ is required (since for $\alpha = 50\%$ no buffer is added). Here, let us stress that larger pairwise separations over the IAF help to limit IAF schedule disruptions, and thereby IAF separation loss, once the uncertainty is revealed. Moreover, stochastic schedules exhibit fewer position shifts among aircraft from the same turbulence category than their



(a) Planned vs Deterministic



(b) Planned vs Stochastic

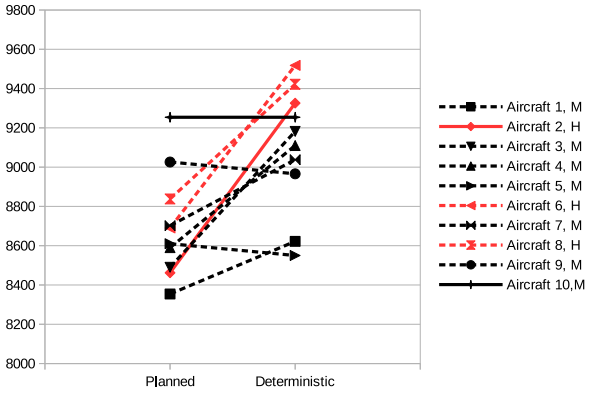
Figure 3: Planned, deterministic, and stochastic schedule for instance 10_607_623_W and $\alpha = 50\%$.

deterministic counterparts, which ensures fairness among aircraft. As a drawback, we remark that stochastic schedules span over longer time frames than deterministic schedules, which may decrease the arrival/landing rate in low-to-moderate density traffic situations, as observed in Khassiba et al. 29.

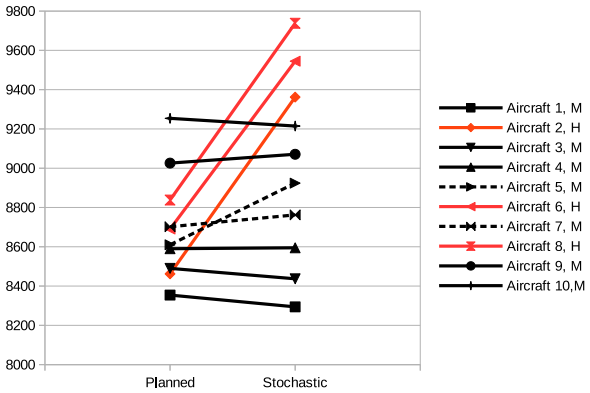
From an operational viewpoint, we conclude that an efficient solution to our two-stage stochastic problem may be obtained first by building a sequence using the rule "lighter aircraft first" and where the relative positions of aircraft from the same wake-turbulence category are conserved. Remark that a similar optimality property is proved in the context of a deterministic version of the aircraft landing problem (see Briskorn and Stolletz [8, Lemma 1]). Once the target sequence is fixed, IAF target times can be deduced recursively by enforcing a buffered IAF separation between successive aircraft.

Effect of the weighting parameter λ to trade off first-stage and second-stage costs

We run a subsidiary numerical experiment where the weighting parameter value λ is increased to 4.0. More focus is thereby put on the expected second-stage cost, comparatively to the baseline test case where $\lambda = 1.0$. We limit our experiment to the instance 607_623_10_W, with $\alpha = 50\%$. The main results are given in Table 10. We remark that:



(a) Planned vs Deterministic



(b) Planned vs Stochastic

Figure 4: Planned, deterministic, and stochastic schedule for instance 10_619_634_W and $\alpha = 50\%$.

- more scenarios are needed to satisfy the required stability condition ($n^*_{\mathcal{S}} = 500$ versus 100 scenarios);
- larger VSS are observed (30.28% versus 10.21%), suggesting that two-stage stochastic programming is more relevant when the second-stage cost is of high importance.

Regarding the difference between the stochastic and the deterministic solutions, we remark that, unlike previous tests, sequences with different lengths (in terms of the sum of final-approach separations) are returned. In the stochastic solution, the target sequence is 745-second long, while in the deterministic solution, the sequence is shorter (693 seconds). This experiment recalls that the deterministic approach overlooks the variability of second-stage outcomes, while the stochastic approach proposes a solution that is "suboptimal" for the first-stage problem (the sequence is not the shortest one) but that appears to be better when we take into account both the first and the second stages. Figure 5 displays the planned, deterministic, and stochastic IAF schedules for instance 10_607_623_W with test parameter values $\alpha = 50\%$ and $\lambda = 4.0$.

Finally, computation times exhibit a dramatic increase as λ increases. This is partially due to the fact that the appropriate number of scenarios also increases with λ, yielding a larger problem to solve. However, we remark that even for $n_{\mathcal{S}} = 500$, the computation time for 607_623_10_W with $\alpha = 50\%$ and $\lambda = 1.0$ is 466.99 seconds (see the appendix),

Table 10: Main results on instance 607_623_10_W with $\alpha = 50\%$ for different values of λ.

	$\lambda = 1.0$	$\lambda = 4.0$
$n^*_{\mathcal{S}}$	100	500
VSS	10.21%	30.28%
CPU (sec)	19.46	1737.82

CPU times are obtained by solving the deterministic equivalent problem directly by CPLEX and averaged over 10 replications.

Table 11: Relative VSS (and ΔSeq.) for different values of α.

	α		
Instance Id	50%	90%	95%
10_559_618_N	4.74% (0)	2.81% (0)	1.93% (0)
10_559_618_W	9.79% (0)	2.78% (0)	1.83% (0)
10_607_623_N	2.24% (0)	1.49% (0)	INF
10_607_623_W	10.21% (0)	2.51% (0)	1.70% (0)
10_619_634_N	7.54% (0)	INF	INF
10_619_634_W	10.79% (0)	1.29% (0)	0.25% (0)
10_624_640_N	8.09% (0)	1.81% (0)	0.05% (0)
10_624_640_W	6.40% (0)	2.30% (0)	1.53% (0)
10_634_659_N	6.24% (0)	1.48% (0)	1.59% (0)
10_634_659_W	8.61% (0)	2.18% (0)	0.40% (0)

For each instance, the stochastic solution corresponds to the solution with the highest validation score obtained when solving 10 replications of the SAA-problem formulated using the appropriate number of scenarios $n^*_{\mathcal{S}}$ from Table 9.

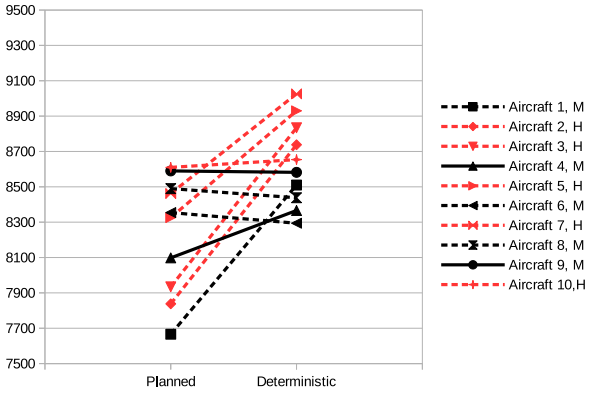
which is three times shorter than with $\lambda = 4.0$.

We conclude that as the importance of the second-stage cost increases from the stakeholder viewpoint, the benefit of two-stage stochastic programming increases. Meanwhile, the problem becomes more difficult in two senses: more scenarios are needed to reach the desired level of stability, and computation times to reach optimality are longer. These facts express the need for fast solution methods. In the next subsection, we compare different solution methods for our two-stage stochastic programs in terms of computation time.

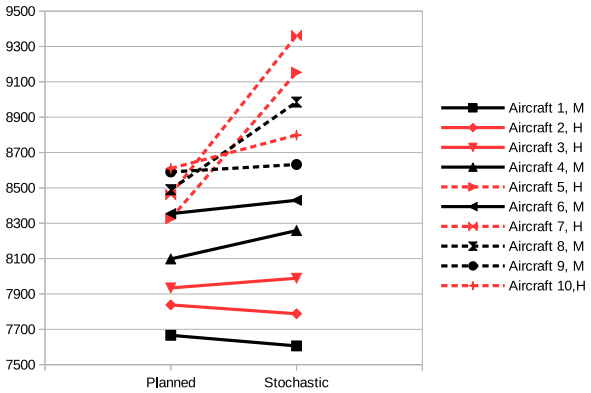
5.4 Solution-method performance comparison

This subsection aims at comparing the performance of the following solution methods applied to our two-stage stochastic program:

1. solving the deterministic equivalent problem with CPLEX using default parameters;
2. applying CPLEX automatic Benders decomposition with Benders strategy FULL;



(a) Planned vs Deterministic



(b) Planned vs Stochastic

Figure 5: Planned, deterministic, and stochastic schedule for instance 10_607_623_W, $\alpha = 50\%$ and $\lambda = 4.0$.

3. applying CPLEX automatic Benders decomposition with Benders strategy USER and specifying completely disaggregated Benders subproblems;

4. applying CPLEX automatic Benders decomposition with Benders strategy USER and specifying partially-aggregated Benders subproblems, where every n_c second-stage scenario problems are aggregated into a single Benders subproblem.

In our two-stage stochastic mixed-integer program, there are both binary and continuous variables in the first stage (δ and x), while all second-stage variables are continuous (y^s, z^{s-}, z^{s+}, and z^{s++} for $s \in \mathcal{S}$). We expect that under Benders strategy FULL, CPLEX is not able to decompose the problem according to its two-stage structure: the Benders master problem will only contain the binary variables δ, while the linking continuous variables, x, will be put inappropriately in the Benders subproblem, and thereby will hinder any further decomposition of the subproblem. With Benders strategy USER, we add annotations to our variables specifying that the first-stage variables (both the binary and the continuous ones) must be kept in the master problem, while second-stage variables must be in one or different subproblems, according to the specified level of aggregation.

In Table 12, we report average solution times obtained applying the above four solution methods. "Determ. Eq." stands for CPLEX solving the deterministic equivalent with default parameters. "Auto. Benders Disagg" and "Auto. Benders 5-Agg" correspond

to using CPLEX automatic Benders decomposition with Benders strategy USER. In the former, a complete disaggregation by scenarios is specified for the Benders subproblem, while in the latter, a partial aggregation every five scenarios is considered ($K = \frac{n_{\mathcal{S}}}{5}$ clusters; the first cluster contains scenarios 1 to 5, the second cluster contains scenarios 6 to 10, and so on). "Auto. Benders FULL" refers to using CPLEX automatic Benders decomposition with Benders strategy FULL. Results for two instances from our test bed, 10_634_659_N and 10_634_659_W, are not reported because very short computation times are achieved by the four solution methods, which makes the instances not relevant for the comparison.

As expected, the computation times obtained with Benders strategy FULL are the worst. This bad performance is surely due to the inappropriate automatic partitioning that CPLEX applies. Based on this, we conclude that tackling a two-stage stochastic program with a bad decomposition can be even worse than solving it without decomposition.

On the other hand, we remark that clearly CPLEX automatic Benders decomposition under Benders strategy USER (that follows any of our two user-defined decompositions), performs the best for most of the test cases. In fact, both disaggregated and partially-aggregated versions perform better than CPLEX Branch-and-Cut in 16 cases out of 21. This confirms that the structure must be exploited to develop efficient solution methods for two-stage stochastic programs. In addition, the partially-aggregated version ranks as the best solution method among the four studied ones, in 15 cases. This indicates that partially aggregating Benders subproblems may be a successful approach to improve computation times. Hence, subproblem clustering strategies are worthwhile to be explored.

6 Conclusion and perspectives

In this paper, we propose a chance-constrained two-stage stochastic mixed-integer programming model for the extended aircraft arrivals management problem under uncertainty. In the first stage, aircraft are captured 2 to 3 hours away from the IAF. The first-stage problem finds a target sequence and target times of aircraft arrival over the IAF so as to minimize the landing sequence length. First-stage constraints are IAF time windows and IAF separation constraints. The first-stage problem is enriched by chance constraints to limit the risk of IAF separation violations (once uncertainties are revealed) to an acceptable level, that we call the protection level. The second-stage problem considers aircraft shortly before arriving at the IAF up to landing, when actual IAF times become known with certainty. It aims at finding target landing times so as to minimize a time-deviation impact cost function. Second-stage constraints are landing time windows and final-approach separations. The two-stage stochastic program minimizes the weighted sum of the landing sequence length, and the expected second-stage time-deviation impact cost function. We show that under mild conditions first-stage chance constraints can be transformed into linear separation constraints with a buffered minimal IAF separation that depends on the protection level. Also, we approximate the expectation term in the true model using a sample average. We are then left with a large-scale deterministic mixed-integer linear problem. In addition to the extended formulation of the deterministic equivalent problem, we propose a partially-aggregated Benders reformulation, and we explore some acceleration techniques of Benders decomposition.

We carry out an extensive computational study on a realistic test bed consisting of 10 instances corresponding to aircraft arrivals on Paris Charles-de-Gaulle airport. The

Table 12: Performance comparison between CPLEX B&C, CPLEX automatic Benders with disaggregated and partially-aggregated subproblems.

			Determ. Eq.	Auto. Benders Disagg	Auto. Benders 5-Agg	Auto. Benders FULL
Instance Id	α	$n^*_{\mathcal{S}}$	CPU	CPU	CPU	CPU
10_559_618_N	50%	200	**3.51**	10.47*	7.56*^{2}	14.35
	90%	100	**1.07**	2.92*	2.19*^{3}	3.14
	95%	50	**0.46**	0.90	0.63	0.88
10_559_618_W	50%	100	9.05	7.69	**5.89**	36.73
	90%	100	7.33	4.17	**3.65**	14.86
	95%	50	2.35	1.43	**1.14**	3.12
10_607_623_N	50%	500	186.58	88.07*^{3}	**32.06***^{8}	585.57
	90%	500	27.63	15.13	**11.64***	96.57
	95%	INF	-	-	-	-
10_607_623_W	50%	100	21.90	9.90*^{2}	**8.40***^{3}	162.67
	90%	100	13.88	4.64	**3.61**	61.57
	95%	100	9.65	4.62	**3.50**	43.34
10_619_634_N	50%	200	14.31	25.87*	**11.49***^{2}	88.05
	90%	INF	-	-	-	-
	95%	INF	-	-	-	-
10_619_634_W	50%	100	31.42	7.63	**7.39**	133.20
	90%	100	13.34	6.22	**5.76**	41.13
	95%	100	34.08	7.02	**6.63**	71.99
10_624_640_N	50%	200	14.06	**10.43***	14.16*	56.56
	90%	100	**1.41**	2.61	2.02	5.01
	95%	200	**2.63**	4.79	3.25	10.75
10_624_640_W	50%	100	67.21	15.01*	**9.38***^{2}	405.57
	90%	100	35.48	6.51	**5.91**	97.26
	95%	100	39.61	5.30	**4.82**	63.73

CPU times (seconds) are obtained using a Python 2.7 code with DOCplex package and CPLEX 12.7.1. The label ($*^k$) indicates that there are k replications (over 10) not solved to optimality and for which CPLEX stopped raising a computational error. The value of k is dropped from the label when $k = 1$.

analysis of the value of stochastic solution leads to the conclusion that the benefit of solving a two-stage stochastic program is more prominent in the situation of high-density air traffic, and when aircraft and controllers have many degrees of freedom for re-scheduling. Also, since a deterministic approach to our problem overlooks the variability of second-stage outcomes, the benefit of two-stage stochastic programming is shown to increase when the second-stage cost has a great importance for the stakeholder. In this case, the stochastic approach may propose an arrival schedule for the first-stage problem that appears "suboptimal" but that is indeed better when we take into account both the first and the second-stage costs. Moreover, we observe a sharp decrease of the VSS with the presence of chance constraints. This indicates that hedging against uncertainty using buffered separations leads to a better performance of the deterministic approach. We also observe that as the benefit of two-stage stochastic programming increases, computation times to reach optimality increase. This fact expresses the need for fast solution methods.

We compare the performance of several solution methods applied to our two-stage stochastic program: CPLEX with default parameter values, automatic Benders decomposition by CPLEX with two Benders strategies, and different Benders subproblem aggregation levels. We remark that clearly CPLEX automatic Benders decomposition, under Benders strategy USER, that follows the user-defined decomposition, performs the best for most of the test cases. Also, partially aggregating Benders subproblems is shown to be a successful approach to improve computation times.

Future work will focus on extending the proposed model to the case with multiple IAFs and multiple runways. Our perspectives also include solving the dynamic case where the arrival set evolves in time. Also, while in this paper we focus on Monte-Carlo sampling, more scenario-generation techniques can be explored in an attempt to reduce the appropriate number of scenarios to reach a satisfying level of stability. In terms of solution methods based on partially-aggregated-cut Benders decomposition, more scenario-subproblems clustering policies can be explored. Finally, as suggested by an anonymous referee, the particular structure of the Benders subproblem could be exploited to solve it more efficiently. As a matter of fact, a recent work of Faye [14] proposes a dynamic-programming approach to solve the problem of determining aircraft landing times, given a fixed sequence. A work is in progress to adapt these algorithms to a manual implementation of Benders decomposition for further acceleration.

References

[1] Shabbir Ahmed. Two-stage stochastic integer programming: A brief introduction. In J. J. Cochran, L.A. Cox, P. Keskinocak, J.P. Kharoufeh, and J.C. Smith, editors, *Wiley Encyclopedia of Operations Research and Management Science*. Wiley Online Library, 2011.

[2] Hamsa Balakrishnan and Bala G. Chandran. Algorithms for scheduling runway operations under constrained position shifting. *Operations Research*, 58(6):1650–1665, 2010. doi: 10.1287/opre.1100.0869.

[3] John E. Beasley, Mohan Krishnamoorthy, Yazid M. Sharaiha, and David Abramson. Scheduling aircraft landings: The static case. *Transportation Science*, 34(2):180–197, 2000. doi: 10.1287/trsc.34.2.180.12302.

[4] Julia A. Bennell, Mohammad Mesgarpour, and Chris N. Potts. Airport runway scheduling. *4OR*, 9(2):115–138, 2011. doi: 10.1007/s10288-011-0172-x.

[5] Julia A. Bennell, Mohammad Mesgarpour, and Chris N. Potts. Dynamic scheduling of aircraft landings. *European Journal of Operational Research*, 258(1):315–327, 2017. doi: 10.1016/j.ejor.2016.08.015.

[6] John R. Birge and François Louveaux. *Introduction to stochastic programming*. Springer Science & Business Media, 2011.

[7] Christabelle S. Bosson and Dengfeng Sun. Optimization of airport surface operations under uncertainty. *Journal of Air Transportation*, 24(3):84–92, 2016. doi: 10.2514/1.D0013.

[8] Dirk Briskorn and Raik Stolletz. Aircraft landing problems with aircraft classes. *Journal of Scheduling*, 17(1):31–45, 2014.

[9] Bala G. Chandran and Hamsa Balakrishnan. A dynamic programming algorithm for robust runway scheduling. In *Proceedings of the 2007 American Control Conference*, pages 1161–1166. IEEE, 2007.

[10] Abraham Charnes and William W. Cooper. Deterministic equivalents for optimizing and satisficing under chance constraints. *Operations Research*, 11(1):18–39, 1963. doi: 10.1287/opre.11.1.18.

[11] Richard de Neufville, Amedeo Odoni, Peter Belobaba, and Tom Reynolds. *Airport Systems: Planning, Design and Management*. McGraw-Hill, 2nd edition, 2013.

[12] Roger G. Dear. The dynamic scheduling of aircraft in the near terminal area. Technical Report R76-9, Massachusetts Institute of Technology. Flight Transportation Laboratory, 1976.

[13] Aviation Systems Division. Traffic Management Advisor, 2016. URL `https://www.aviationsystemsdivision.arc.nasa.gov/research/foundations/tma.shtml`.

[14] Alain Faye. A quadratic time algorithm for computing the optimal landing times of a fixed sequence of planes. *European Journal of Operational Research*, 270:1148–1157, 2018. doi: 10.1016/j.ejor.2018.04.021.

[15] Matteo Fischetti and Domenico Salvagnin. An in-out approach to disjunctive optimization. In *International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) Techniques in Constraint Programming*, pages 136–140. Springer, 2010.

[16] Matteo Fischetti, Ivana Ljubić, and Markus Sinnl. Benders decomposition without separability: A computational study for capacitated facility location problems. *European Journal of Operational Research*, 253(3):557–569, 2016. ISSN 0377-2217. doi: 10.1016/j.ejor.2016.03.002. URL `http://www.sciencedirect.com/science/article/pii/S0377221716301126`.

[17] Matteo Fischetti, Ivana Ljubić, and Markus Sinnl. Redesigning benders decomposition for large-scale facility location. *Management Science*, 63(7):2146–2162, 2016. doi: 10.1287/mnsc.2016.2461.

[18] Michael C. Fu et al. *Handbook of Simulation Optimization*, volume 216. Springer, 2015.

[19] Fabio Furini, Martin P. Kidd, Carlo A. Persiani, and Paolo Toth. Improved rolling horizon approaches to the aircraft sequencing problem. *Journal of Scheduling*, 18(5): 435–447, 2015.

[20] Jean-Louis Garcia. MAESTRO-A metering and spacing tool. In *Proceedings of the 1990 American Control Conference*, pages 502–507. IEEE, 1990.

[21] Nathalie Hasevoets and Paul Conroy. Arrival Manager - Implementation Guidelines and Lessons Learned. Technical report, EUROCONTROL, 2010. URL `https://www.eurocontrol.int/sites/default/files/article/content/documents/nm/fasti-aman-guidelines-2010.pdf`.

[22] Andreas Heidt. *Uncertainty Models for Optimal and Robust ATM Schedules*. PhD thesis, Friedrich Alexander University, Erlangen, Germany, 2017.

[23] Andreas Heidt, Hartmut Helmke, Manu Kapolke, Frauke Liers, and Alexander Martin. Robust runway scheduling under uncertain conditions. *Journal of Air Transport Management Part A*, 56:28–37, 2016. doi: 10.1016/j.jairtraman.2016.02.009.

[24] IBM. Benders Decomposition in CPLEX 12.7.0, at CPLEX School June 2017, Montreal, Canada, June 2017.

[25] IBM. IBM ILOG CPLEX Optimization Studio CPLEX User's Manual Version 12 Release 7, 2017.

[26] Manu Kapolke, Norbert Fürstenau, Andreas Heidt, Frauke Liers, Monika Mittendorf, and Christian Weiß. Pre-tactical optimization of runway utilization under uncertainty. *Journal of Air Transport Management Part A*, 56:48–56, 2016. doi: 10.1016/j.jairtraman.2016.02.004.

[27] Michal Kaut and Stein W. Wallace. Evaluation of scenario-generation methods for stochastic programming. *Pacific Journal of Optimization*, 3(2):257–271, 2007.

[28] James E Kelley. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

[29] Ahmed Khassiba, Fabian Bastin, Bernard Gendron, Sonia Cafieri, and Marcel Mongeau. Extended aircraft arrival management under uncertainty: A computational study. *Journal of Air Transportation*, 27(3):131–143, 2019. doi: 10.2514/1.D0135.

[30] Ed Klotz. Automatic Benders Decomposition in CPLEX, October 2017.

[31] Gilbert Laporte and François V. Louveaux. The integer L-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, 13(3): 133–142, 1993.

[32] Hanbong Lee. Tradeoff evaluation of scheduling algorithms for terminal-area air traffic control. Master's thesis, Massachusetts Institute of Technology, 2008.

[33] Dale McDaniel and Mike Devine. A modified Benders' partitioning algorithm for mixed integer programming. *Management Science*, 24(3):312–319, 1977. doi: 10.1287/mnsc.24.3.312.

[34] Larry A. Meyn and Heinz Erzberger. Airport arrival capacity benefits due to improved scheduling accuracy. In *Proceedings of the 5th Aviation, Technology Integration and Operations and the 16th Lighter-Than-Air Systems Technology and Balloon Systems Conferences*, 2005.

[35] Bruce L. Miller and Harvey M. Wagner. Chance constrained programming with joint constraints. *Operations Research*, 13(6):930–945, 1965. doi: 10.1287/opre.13.6.930.

[36] Clair E. Miller, Albert W. Tucker, and Richard A. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM*, 7(4):326–329, 1960.

[37] Frank Neuman and Heinz Erzberger. Analysis of sequencing and scheduling methods for arrival traffic. Technical report, National Aeronautics and Space Administration, 1990.

[38] Ragheb Rahmaniani, Teodor G. Crainic, Michel Gendreau, and Walter Rei. The Benders decomposition algorithm: A literature review. *European Journal of Operational Research*, 259(3):801–817, 2017.

[39] Alexander Shapiro and Tito Homem-de Mello. On the rate of convergence of optimal solutions of Monte Carlo approximations of stochastic programs. *SIAM Journal on Optimization*, 11(1):70–86, 2000. doi: 10.1137/S1052623498349541.

[40] Gustaf Sölveling. *Stochastic programming methods for scheduling of airport runway operations under uncertainty*. PhD thesis, Georgia Institute of Technology, Atlanta, GA, 2012.

[41] Gustaf Sölveling and John-Paul Clarke. Scheduling of airport runway operations using stochastic branch and bound methods. *Transportation Research Part C*, 45: 119–137, 2014. doi: 10.1016/j.trc.2014.02.021.

[42] Gustaf Sölveling, Senay Solak, John-Paul Clarke, and Ellis Johnson. Runway operations optimization in the presence of uncertainties. *Journal of Guidance, Control, and Dynamics*, 34(5):1373–1382, 2011. doi: 10.2514/6.2010-9252.

[43] Maarten Tielrooij, Clark Borst, Marinus M. Van Paassen, and Max Mulder. Predicting arrival time uncertainty from actual flight information. In *Proceedings of the 11th USA/Europe Air Traffic Management Research and Development Seminar*, pages 577–586. FAA/EUROCONTROL, 2015.

[44] Uwe Völckers. Arrival planning and sequencing with COMPAS-OP at the Frankfurt ATC-Center. In *Proceedings of the 1990 American Control Conference*, pages 496–501. IEEE, 1990.

[45] Richard D. Wollmer. Two-stage linear programming under uncertainty with 0–1 integer first stage variables. *Mathematical Programming*, 19(1):279–288, 1980.

7 Appendix: Extensive results

Tables 13 to 22 report extensive results of tests carried out on the ten instances from our test bed for a number of scenarios $n_{\mathcal{S}}$ ranging from 10 to 500, and for three values of the protection level α against IAF separation loss (50%, 90%, and 95%). In column "CPU", the average CPLEX solving time over 10 replications is expressed in seconds. Column "$\bar{v} \pm I_{95\%}$" gives the average objective-function value, $\bar{v}$, over the 10 replications as well as the mid-length Student-based 95% confidence interval ($I_{95\%}$). Likewise, the columns "Validation score" and "Validation gap" report respectively the average validation score and the average validation gap over the 10 replications as well as the mid-length Student-based 95% confidence intervals. The last column "# Seq." reports the number of different sequences over the solutions of the 10 replications (given that only one solution is retained per replication).

Table 13: Results of instance 10_559_618_N.

$n_{\mathcal{S}}$	α	CPU (s)	Status (Gap)	$\bar{v} \pm I_{95\%}$	Validation score	Validation Gap	# Seq.
10	50%	0.16	Opt. (0.0%)	812.2 ±3.5	822.3 ±3.3	-1.22% ±0.63	3
	90%	0.13	Opt. (0.0%)	854.4 ±2.1	858.4 ±0.5	-0.46% ±0.28	3
	95%	0.10	Opt. (0.0%)	857.8 ±4.0	858.3 ±0.6	-0.05% ±0.50	1
50	50%	0.58	Opt. (0.0%)	813.5 ±1.9	816.4 ±0.4	-0.36% ±0.24	2
	90%	0.53	Opt. (0.0%)	855.2 ±0.7	857.1 ±0.3	-0.21% ±0.07	2
	95%	0.37	Opt. (0.0%)	857.1 ±0.9	857.7 ±0.2	-0.08% ±0.11	1
100	50%	1.18	Opt. (0.0%)	814.5 ±1.1	816.0 ±0.1	-0.19% ±0.13	2
	90%	1.06	Opt. (0.0%)	855.7 ±0.5	856.7 ±0.2	-0.11% ±0.07	2
	95%	0.88	Opt. (0.0%)	857.4 ±0.6	857.6 ±0.1	-0.02% ±0.07	1
200	50%	4.49	Opt. (0.0%)	814.8 ±0.9	815.9 ±0.1	-0.13% ±0.11	2
	90%	2.51	Opt. (0.0%)	856.0 ±0.4	856.5 ±0.2	-0.07% ±0.05	2
	95%	1.96	Opt. (0.0%)	857.3 ±0.4	857.5 ±0.1	-0.03% ±0.05	1
500	50%	32.22	Opt. (0.0%)	814.9 ±0.4	815.7 ±0.1	-0.09% ±0.05	2
	90%	16.58	Opt. (0.0%)	856.1 ±0.1	856.3 ±0.1	-0.03% ±0.02	2
	95%	13.45	Opt. (0.0%)	857.3 ±0.1	857.5 ±0.0	-0.02% ±0.02	1

Solution method: Deterministic equivalent problem solved by CPLEX

Table 14: Results of instance 10_559_618_W.

$n_{\mathcal{S}}$	α	CPU (s)	Status (Gap)	$\bar{v} \pm I_{95\%}$	Validation score	Validation Gap	# Seq.
10	50%	0.76	Opt. (0.0%)	751.1 ±3.0	756.5 ±2.0	-0.71% ±0.45	6
	90%	0.55	Opt. (0.0%)	799.0 ±0.0	805.0 ±1.1	-0.74% ±0.13	3
	95%	0.52	Opt. (0.0%)	799.0 ±0.0	803.3 ±1.3	-0.54% ±0.16	4
50	50%	3.98	Opt. (0.0%)	751.9 ±1.0	753.1 ±0.2	-0.16% ±0.13	6
	90%	2.38	Opt. (0.0%)	799.0 ±0.0	800.3 ±0.4	-0.17% ±0.05	10
	95%	2.11	Opt. (0.0%)	799.0 ±0.0	800.2 ±0.4	-0.15% ±0.05	10
100	50%	9.86	Opt. (0.0%)	752.0 ±0.8	752.8 ±0.1	-0.11% ±0.11	7
	90%	6.78	Opt. (0.0%)	799.0 ±0.0	799.6 ±0.1	-0.07% ±0.01	10
	95%	7.28	Opt. (0.0%)	799.0 ±0.0	799.5 ±0.1	-0.06% ±0.01	10
200	50%	31.75	Opt. (0.0%)	752.3 ±0.6	752.7 ±0.1	-0.05% ±0.08	4
	90%	23.17	Opt. (0.0%)	799.0 ±0.0	799.3 ±0.1	-0.04% ±0.01	10
	95%	23.64	Opt. (0.0%)	799.0 ±0.0	799.4 ±0.1	-0.05% ±0.01	10
500	50%	138.39	Opt. (0.0%)	752.2 ±0.3	752.6 ±0.0	-0.05% ±0.05	7
	90%	150.40	Opt. (0.0%)	799.0 ±0.0	799.1 ±0.0	-0.01% ±0.00	8
	95%	93.28	Opt. (0.0%)	799.0 ±0.0	799.1 ±0.0	-0.02% ±0.00	8

Solution method: Deterministic equivalent problem solved by CPLEX

Table 15: Results of instance 10_607_623_N.

$n_{\mathcal{S}}$	α	CPU (s)	Status (Gap)	$\bar{v} \pm I_{95\%}$	Validation score	Validation Gap	# Seq.
10	50%	0.48	Opt. (0.0%)	812.7 ±7.3	824.8 ±3.0	-1.50% ±0.89	8
	90%	0.21	Opt. (0.0%)	863.9 ±7.2	862.8 ±1.1	0.12% ±0.78	2
	95%		INF	±	±	%	
50	50%	2.59	Opt. (0.0%)	813.6 ±2.7	819.8 ±0.1	-0.76% ±0.34	7
	90%	0.83	Opt. (0.0%)	860.8 ±2.0	862.2 ±0.1	-0.16% ±0.23	2
	95%		INF	±	±	%	
100	50%	6.48	Opt. (0.0%)	815.6 ±2.5	819.5 ±0.2	-0.48% ±0.32	6
	90%	1.38	Opt. (0.0%)	861.5 ±1.8	862.1 ±0.1	-0.07% ±0.21	2
	95%		INF	±	±	%	
200	50%	24.19	Opt. (0.0%)	816.6 ±2.3	819.5 ±0.3	-0.36% ±0.28	7
	90%	3.54	Opt. (0.0%)	861.3 ±1.6	862.0 ±0.1	-0.08% ±0.18	2
	95%		INF	±	±	%	
500	50%	179.82	Opt. (0.0%)	817.6 ±1.0	819.3 ±0.2	-0.20% ±0.12	7
	90%	27.19	Opt. (0.0%)	862.0 ±0.9	862.0 ±0.0	0.00% ±0.11	2
	95%		INF	±	±	%	

Solution method: Deterministic equivalent problem solved by CPLEX

Table 16: Results of instance 10_607_623_W.

$n_{\mathcal{S}}$	α	CPU (s)	Status (Gap)	$\bar{v} \pm I_{95\%}$	Validation score	Validation Gap	# Seq.
10	50%	1.05	Opt. (0.0%)	705.8 ±1.2	711.7 ±2.3	-0.83% ±0.42	4
	90%	0.85	Opt. (0.0%)	745.0 ±0.0	751.5 ±1.2	-0.86% ±0.16	10
	95%	0.75	Opt. (0.0%)	745.0 ±0.0	748.1 ±0.7	-0.41% ±0.09	10
50	50%	7.10	Opt. (0.0%)	706.7 ±1.3	707.4 ±0.3	-0.09% ±0.20	2
	90%	4.88	Opt. (0.0%)	745.0 ±0.0	746.6 ±0.3	-0.21% ±0.04	10
	95%	4.70	Opt. (0.0%)	745.0 ±0.0	746.2 ±0.3	-0.16% ±0.04	10
100	50%	19.46	Opt. (0.0%)	707.0 ±1.0	707.1 ±0.3	-0.02% ±0.13	2
	90%	14.04	Opt. (0.0%)	745.0 ±0.0	745.7 ±0.2	-0.10% ±0.02	9
	95%	10.65	Opt. (0.0%)	745.0 ±0.0	745.7 ±0.1	-0.09% ±0.02	9
200	50%	76.26	Opt. (0.0%)	706.6 ±0.5	707.0 ±0.1	-0.05% ±0.07	1
	90%	156.63	Opt. (0.0%)	745.0 ±0.0	745.4 ±0.1	-0.05% ±0.01	9
	95%	84.80	Opt. (0.0%)	745.0 ±0.0	745.4 ±0.1	-0.06% ±0.01	10
500	50%	466.99	Opt. (0.0%)	706.9 ±0.4	706.8 ±0.0	0.01% ±0.06	1
	90%	1446.55	Opt. (0.0%)	745.1 ±0.0	745.3 ±0.0	-0.03% ±0.01	10
	95%	973.40	Opt. (0.0%)	745.1 ±0.0	745.3 ±0.0	-0.03% ±0.01	10

Solution method: Deterministic equivalent problem solved by CPLEX

Table 17: Results of instance 10_619_634_N.

$n_{\mathcal{S}}$	α	CPU (s)	Status (Gap)	$\bar{v} \pm I_{95\%}$	Validation score	Validation Gap	# Seq.
10	50%	0.29	Opt. (0.0%)	778.7 ±5.8	784.4 ±2.0	-0.73% ±0.73	1
	90%		INF	±	±	% ±	
	95%		INF	±	±	% ±	
50	50%	1.53	Opt. (0.0%)	778.4 ±1.8	780.1 ±0.3	-0.22% ±0.23	1
	90%		INF	±	±	% ±	
	95%		INF	±	±	% ±	
100	50%	3.69	Opt. (0.0%)	779.5 ±1.3	780.0 ±0.1	-0.07% ±0.17	1
	90%		INF	±	±	% ±	
	95%		INF	±	±	% ±	
200	50%	12.33	Opt. (0.0%)	779.0 ±0.8	779.9 ± 0.1	-0.11% ±0.11	1
	90%		INF	±	±	% ±	
	95%		INF	±	±	% ±	
500	50%	106.28	Opt. (0.0%)	779.3 ±0.7	779.7 ±0.0	-0.06% ±0.08	1
	90%		INF	±	±	% ±	
	95%		INF	±	±	% ±	

Solution method: Deterministic equivalent problem solved by CPLEX

Table 18: Results of instance 10_619_634_W.

$n_{\mathcal{S}}$	α	CPU (s)	Status (Gap)	$\bar{v} \pm I_{95\%}$	Validation score	Validation Gap	# Seq.
10	50%	0.51	Opt. (0.0%)	666.0 ±0.0	672.5 ±2.4	-0.96% ±0.35	9
	90%	0.42	Opt. (0.0%)	666.0 ±0.0	670.6 ±0.9	-0.69% ±0.13	8
	95%	0.94	Opt. (0.0%)	666.0 ±0.0	668.6 ±0.5	-0.39% ±0.08	5
50	50%	12.27	Opt. (0.0%)	666.4 ±0.1	667.8 ±0.3	-0.21% ±0.04	5
	90%	8.79	Opt. (0.0%)	666.4 ±0.1	667.7 ±0.2	-0.20% ±0.03	5
	95%	10.86	Opt. (0.0%)	666.5 ±0.2	667.5 ±0.1	-0.16% ±0.03	5
100	50%	38.84	Opt. (0.0%)	666.8 ±0.2	667.5 ±0.1	-0.11% ±0.03	5
	90%	19.84	Opt. (0.0%)	666.8 ±0.2	667.5 ±0.1	-0.11% ±0.03	6
	95%	37.38	Opt. (0.0%)	666.8 ±0.2	667.4 ±0.1	-0.09% ±0.03	4
200	50%	90.81	Opt. (0.0%)	666.9 ±0.1	667.4 ±0.1	-0.08% ±0.03	4
	90%	63.78	Opt. (0.0%)	667.1 ±0.4	667.6 ±0.3	-0.07% ±0.03	6
	95%	137.41	Opt. (0.0%)	666.9 ±0.1	667.3 ±0.0	-0.07% ±0.03	4
500	50%	404.1	Opt. (0.0%)	667.1 ±0.1	667.3 ±0.0	-0.03% ±0.02	4
	90%	266.02	Opt. (0.0%)	667.1 ±0.1	667.3 ±0.0	-0.03% ±0.02	4
	95%	500.38	Opt. (0.0%)	667.1 ±0.1	667.3 ±0.0	-0.03% ±0.02	4

Solution method: Deterministic equivalent problem solved by CPLEX

Table 19: Results of instance 10_624_640_N.

$n_{\mathcal{S}}$	α	CPU (s)	Status (Gap)	$\bar{v} \pm I_{95\%}$	Validation score	Validation Gap	# Seq.
10	50%	0.31	Opt. (0.0%)	809.3 ±2.2	819.4 ±1.9	-1.23% ±0.28	7
	90%	0.17	Opt. (0.0%)	812.7 ±3.0	818.6 ±2.5	-0.72% ±0.53	9
	95%	0.12	Opt. (0.0%)	828.8 ±4.8	833.7 ±2.1	-0.59% ±0.55	7
50	50%	1.83	Opt. (0.0%)	812.2 ±1.1	815.0 ±0.4	-0.34% ±0.14	7
	90%	0.63	Opt. (0.0%)	814.8 ±1.3	816.1 ±0.3	-0.16% ±0.14	8
	95%	0.53	Opt. (0.0%)	830.5 ±2.4	832.1 ±0.1	-0.19% ±0.27	4
100	50%	4.67	Opt. (0.0%)	813.0 ±0.6	814.6 ±0.2	-0.20% ±0.08	7
	90%	1.37	Opt. (0.0%)	815.4 ±0.9	816.0 ±0.2	-0.07% ±0.12	6
	95%	0.98	Opt. (0.0%)	831.0 ±1.7	832.0 ±0.1	-0.12% ±0.20	5
200	50%	13.66	Opt. (0.0%)	813.4 ±0.6	814.5 ±0.2	-0.13% ±0.09	6
	90%	3.65	Opt. (0.0%)	815.4 ±0.8	815.9 ±0.2	-0.06% ±0.10	7
	95%	2.40	Opt. (0.0%)	831.2 ±1.3	832.0 ±0.0	-0.10% ±0.15	5
500	50%	61.91	Opt. (0.0%)	813.7 ±0.2	814.2 ±0.1	-0.07% ±0.03	6
	90%	26.84	Opt. (0.0%)	815.6 ±0.4	815.8 ±0.0	-0.02% ±0.05	5
	95%	26.56	Opt. (0.0%)	831.8 ±0.8	831.9 ±0.1	-0.01% ±0.09	5

Solution method: Deterministic equivalent problem solved by CPLEX

Table 20: Results of instance 10_624_640_W.

$n_{\mathcal{S}}$	α	CPU (s)	Status (Gap)	$\bar{v} \pm I_{95\%}$	Validation score	Validation Gap	# Seq.
10	50%	2.32	Opt. (0.0%)	718.0 ±0.0	724.7 ±2.1	-0.93% ±0.29	10
	90%	1.29	Opt. (0.0%)	718.0 ±0.0	722.3 ±1.1	-0.59% ±0.15	10
	95%	1.00	Opt. (0.0%)	718.0 ±0.0	722.4 ±1.5	-0.60% ±0.21	10
50	50%	17.00	Opt. (0.0%)	718.0 ±0.0	719.3 ±0.2	-0.18% ±0.03	10
	90%	6.00	Opt. (0.0%)	718.0 ±0.0	719.4 ±0.3	-0.19% ±0.03	10
	95%	5.39	Opt. (0.0%)	718.0 ±0.0	719.2 ±0.2	-0.17% ±0.02	10
100	50%	63.56	Opt. (0.0%)	718.0 ±0.0	718.9 ±0.1	-0.13% ±0.01	9
	90%	35.66	Opt. (0.0%)	718.0 ±0.0	719.0 ±0.1	-0.13% ±0.02	10
	95%	29.48	Opt. (0.0%)	718.0 ±0.0	718.9 ±0.1	-0.12% ±0.01	10
200	50%	536.53	Opt. (0.0%)	718.2 ±0.1	718.8 ±0.0	-0.09% ±0.01	10
	90%	284.40	Opt. (0.0%)	718.2 ±0.1	718.8 ±0.0	-0.09% ±0.01	10
	95%	230.54	Opt. (0.0%)	718.2 ±0.1	718.8 ±0.1	-0.09% ±0.01	10
500	50%	3164.99	9 Opt. (0.0%)	718.3 ±0.1	718.7 ±0.0	-0.05% ±0.01	10
	90%	1540.18	Opt. (0.0%)	718.3 ±0.1	718.7 ±0.0	-0.06% ±0.01	10
	95%	1056.55	Opt. (0.0%)	718.3 ±0.1	718.7 ±0.0	-0.05% ±0.01	10

Solution method: Deterministic equivalent problem solved by CPLEX

Table 21: Results of instance 10_634_659_N.

$n_{\mathcal{S}}$	α	CPU (s)	Status (Gap)	$\bar{v} \pm I_{95\%}$	Validation score	Validation Gap	# Seq.
10	50%	0.09	Opt. (0.0%)	770.0 ±0.0	777.5 ±1.6	-0.97% ±0.20	6
	90%	0.07	Opt. (0.0%)	770.0 ±0.0	772.9 ±0.5	-0.38% ±0.06	5
	95%	0.06	Opt. (0.0%)	770.0 ±0.0	772.2 ±1.2	-0.28% ±0.16	2
50	50%	0.34	Opt. (0.0%)	770.0 ±0.1	771.3 ±0.2	-0.16% ±0.04	2
	90%	0.27	Opt. (0.0%)	770.0 ±0.1	771.4 ±0.3	-0.18% ±0.04	3
	95%	0.18	Opt. (0.0%)	770.0 ±0.1	771.0 ±0.1	-0.12% ±0.02	2
100	50%	0.85	Opt. (0.0%)	770.1 ±0.1	770.8 ±0.2	-0.10% ±0.02	2
	90%	0.62	Opt. (0.0%)	770.1 ±0.1	770.8 ±0.1	-0.09% ±0.02	2
	95%	0.47	Opt. (0.0%)	770.1 ±0.1	770.7 ±0.1	-0.08% ±0.02	2
200	50%	2.01	Opt. (0.0%)	770.1 ±0.1	770.4 ±0.1	-0.04% ±0.01	2
	90%	1.27	Opt. (0.0%)	770.1 ±0.1	770.4 ±0.1	-0.04% ±0.01	2
	95%	0.99	Opt. (0.0%)	770.1 ±0.1	770.4 ±0.0	-0.04% ±0.01	2
500	50%	12.56	Opt. (0.0%)	770.2 ±0.1	770.3 ±0.0	-0.01% ±0.01	2
	90%	4.32	Opt. (0.0%)	770.2 ±0.0	770.3 ±0.0	-0.01% ±0.01	2
	95%	2.99	Opt. (0.0%)	770.2 ±0.0	770.3 ±0.0	-0.01% ±0.01	2

Solution method: Deterministic equivalent problem solved by CPLEX

Table 22: Results of instance 10_634_659_W.

$n_{\mathcal{S}}$	α	CPU (s)	Status (Gap)	$\bar{v} \pm I_{95\%}$	Validation score	Validation Gap	# Seq.
10	50%	0.15	Opt. (0.0%)	718.0 ±0.0	725.6 ±1.4	-1.05% ±0.19	10
	90%	0.09	Opt. (0.0%)	718.0 ±0.0	722.2 ±0.6	-0.58% ±0.08	2
	95%	0.10	Opt. (0.0%)	718.0 ±0.0	720.8 ±0.5	-0.39% ±0.07	3
50	50%	0.39	Opt. (0.0%)	718.0 ±0.0	719.2 ±0.4	-0.16% ±0.06	6
	90%	0.21	Opt. (0.0%)	718.0 ±0.0	719.3 ±0.3	-0.18% ±0.04	2
	95%	0.20	Opt. (0.0%)	718.0 ±0.0	719.1 ±0.2	-0.15% ±0.03	2
100	50%	0.87	Opt. (0.0%)	718.0 ±0.0	718.5 ±0.1	-0.08% ±0.02	10
	90%	0.50	Opt. (0.0%)	718.0 ±0.0	718.7 ±0.2	-0.09% ±0.03	3
	95%	0.48	Opt. (0.0%)	718.0 ±0.0	718.6 ±0.1	-0.09% ±0.02	3
200	50%	2.23	Opt. (0.0%)	718.0 ±0.0	718.3 ±0.1	-0.04% ±0.01	10
	90%	1.58	Opt. (0.0%)	718.0 ±0.0	718.3 ±0.1	-0.05% ±0.01	6
	95%	1.60	Opt. (0.0%)	718.0 ±0.0	718.3 ±0.1	-0.05% ±0.01	6
500	50%	11.56	Opt. (0.0%)	718.0 ±0.0	718.1 ±0.0	-0.02% ±0.00	10
	90%	6.39	Opt. (0.0%)	718.0 ±0.0	718.1 ±0.0	-0.02% ±0.00	6
	95%	6.95	Opt. (0.0%)	718.0 ±0.0	718.1 ±0.0	-0.02% ±0.01	6

Solution method: Deterministic equivalent problem solved by CPLEX