Programmation stochastique à deux étapes pour l'ordonnancement des arrivées d'avions sous incertitude

Ahmed Khassiba

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par l'Université Toulouse 3 - Paul Sabatier

Cotutelle internationale : Université de Montréal

Présentée et soutenue par

Ahmed KHASSIBA

Le 5 mars 2020

Two-stage stochastic programming for aircraft arrival scheduling under uncertainty - Programmation stochastique à deux étapes pour l'ordonnancement des arrivées d'avions sous incertitude

Ecole doctorale : **EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse**

Spécialité : **Mathématiques et Applications**

Unité de recherche :
Laboratoire de Recherche ENAC

Thèse dirigée par
Marcel MONGEAU et Fabian BASTIN

Jury

Mme Hamsa BALAKRISHNAN, Rapporteure
M. Alain FAYE, Rapporteur
M. Pierre L'ECUYER, Examinateur
M. Jean-Baptiste HIRIART-URRUTY, Examinateur
M. André ROSSI, Examinateur
M. Claus GWIGGNER, Examinateur
Mme Sonia CAFIERI, Co-directrice de thèse
M. Bernard GENDRON, Co-directeur de thèse

Thèse en vue de l'obtention du doctorat délivré par

Université de Toulouse 3 - Paul Sabatier

École Doctorale Mathématiques, Informatique
et Télécommunications de Toulouse

En cotutelle avec

Université de Montréal

Faculté des Arts et des Sciences

Thèse intitulée

Programmation stochastique à deux étapes pour l'ordonnancement des arrivées d'avions sous incertitude

Two-stage stochastic programming for aircraft arrival scheduling under uncertainty

présentée par

Ahmed Khassiba

soutenue le 05 mars 2020

Thèse dirigée par

Fabian Bastin	Université de Montréal
Sonia Cafieri	ENAC
Bernard Gendron	Université de Montréal
Marcel Mongeau	ENAC

Jury

Hamsa Balakrishnan	Massachusetts Institute of Technology
Alain Faye	ENSIIE
Pierre L'Ecuyer	Université de Montréal
Jean-Baptiste Hiriart-Urruty	Université de Toulouse 3 – Paul Sabatier
André Rossi	Université Paris-Dauphine
Claus Gwiggner	Capgemini Invent

Résumé

Dans le contexte d'une augmentation soutenue du trafic aérien et d'une faible marge d'expansion des capacités aéroportuaires, la pression s'accroît sur les aéroports les plus fréquentés pour une utilisation optimale de leur infrastructure, telle que les pistes, reconnues comme le goulot d'étranglement des opérations aériennes. De ce besoin opérationnel est né le problème d'ordonnancement des atterrissages d'avions, consistant à trouver pour les avions se présentant à un aéroport la séquence et les heures d'atterrissage optimales par rapport à certains critères (utilisation des pistes, coût total des retards, etc) tout en respectant des contraintes opérationnelles et de sécurité. En réponse à ce besoin également, depuis les années 1990 aux États-Unis et en Europe, des outils d'aide à la décision ont été mis à la disposition des contrôleurs aériens, afin de les assister dans leur tâche d'assurer la sécurité et surtout la performance des flux d'arrivée.

Un certain nombre de travaux de recherche se sont focalisés sur le cas déterministe et statique du problème d'atterrissage d'avions. Cependant, le problème plus réaliste, de nature stochastique et dynamique, a reçu une attention moindre dans la littérature. De plus, dans le cadre du projet européen de modernisation des systèmes de gestion de trafic aérien, il a été proposé d'étendre l'horizon opérationnel des outils d'aide à la décision de manière à prendre en compte les avions plus loin de l'aéroport de destination. Cette extension de l'horizon opérationnel promet une meilleure gestion des flux d'arrivées via un ordonnancement précoce plus efficient. Néanmoins, elle est inévitablement accompagnée d'une détérioration de la qualité des données d'entrée, rendant indispensable la prise en compte de leur stochasticité.

L'objectif de cette thèse est l'ordonnancement des arrivées d'avions, dans le cadre d'un horizon opérationnel étendu, où les heures effectives d'arrivée des avions sont incertaines. Plus précisément, nous proposons une approche basée sur la programmation stochastique à deux étapes. En première étape, les avions sont pris en considération à 2–3 heures de leur atterrissage prévu à l'aéroport de destination. Il s'agit de les ordonnancer à un point de l'espace aérien aéroportuaire, appelé IAF (*Initial Approach Fix*). Les heures effectives de passage à ce point sont supposées suivre des distributions de probabilité connues. En pratique, cette incertitude peut engendrer un risque à la bonne séparation des avions nécessitant l'intervention des contrôleurs. Afin de limiter la charge de contrôle conséquente, nous introduisons des contraintes en probabilité traduisant le niveau de tolérance aux risques de sécurité à l'IAF après révélation de l'incertitude. La deuxième étape correspond au passage effectif des avions considérés à l'IAF. Comme l'incertitude est révélée, une décision de recours est prise afin d'ordonnancer les avions au seuil de piste en minimisant un critère de deuxième étape (charge de travail des contrôleurs, coût du retard, etc).

La démonstration de faisabilité et une étude numérique de ce problème d'ordonnancement des arrivées d'avions en présence d'incertitude constituent la première contribution de la thèse. La modélisation de ce problème sous la forme d'un problème de programma-

tion stochastique à deux étapes et sa résolution par décomposition de Benders constituent la deuxième contribution. Finalement, la troisième contribution étend le modèle proposé au cas opérationnel, plus réaliste où nous considérons plusieurs points d'approche initiale.

Mots-clés : programmation stochastique à deux étapes, décomposition de Benders, ordonnancement des arrivées d'avions.

Abstract

Airport operations are well known to be a bottleneck in the air traffic system, which puts more and more pressure on the world busiest airports to optimally schedule landings, in particular, and also – but to a smaller extent – departures. The Aircraft Landing Problem (ALP) has arisen from this operational need. ALP consists in finding for aircraft heading to a given airport a landing sequence and landing times so as to optimize some given criteria (optimizing runway utilization, minimizing delays, etc) while satisfying operational constraints (safety constraints mainly). As a reply to this operational need, decision support tools have been designed and put on service for air traffic controllers since the early nineties in the US as well as in Europe.

A considerable number of publications dealing with ALP focus on the deterministic and static case. However, the aircraft landing problem arising in practice has a dynamic nature riddled with uncertainties. In addition, operational horizon of current decision support tools are to be extended so that aircraft are captured at larger distances from the airport to hopefully start the scheduling process earlier. Such a horizon extension affects the quality of input data which enlarges the uncertainty effect.

In this thesis, we aim at scheduling aircraft arrivals under uncertainty. For that purpose, we propose an approach based on two-stage stochastic programming. In the first stage, aircraft are captured at a large distance from the destination airport. They are to be scheduled on the same initial approach fix (IAF), a reference point in the near-to-airport area where aircraft start their approach phase preparing for landing. Actual IAF arrival times are assumed to be random variables with known probability distributions. In practice, such an uncertainty may cause loss of safety separations between aircraft. In such situations, air traffic controllers are expected to intervene to ensure air traffic safety. In order to alleviate the consequent air traffic control workload, chance constraints are introduced so that the safety risks around the IAF are limited to an acceptable level once the uncertainty is revealed. The second stage corresponds to the situation where aircraft are actually close to the IAF. In this stage, the uncertainty is revealed and a recourse decision is made in order to schedule aircraft on the runway threshold so that a second-stage cost function is minimized (e.g., air traffic control workload, delay cost, etc).

Our first contribution is a proof of concept of the extended aircraft arrival management under uncertainty and a computational study on optimization parameters and problem characteristics. Modeling this problem as a two-stage stochastic programming model and solving it by a Benders decomposition is our second contribution. Finally, our third contribution focuses on extending our model to the more realistic case, where aircraft in the first stage are scheduled on several IAFs.

This manuscript is mainly written in English, in the exception of chapters 1, 2, 6, and the two appendices A and B, that are written in French. Chapter 1, with appendices A and B, present basic air-traffic-management concepts and an introduction to two-stage stochastic programming and Benders decomposition. Chapter 2 is a literature review.

Chapters 3, 4, and 5 correspond respectively to a published, an accepted, and a working English-written articles, which present the three contributions of the thesis. Chapter 6 concludes the manuscript.

Keywords : two-stage stochastic programming, Benders decomposition, aircraft arrival scheduling.

Remerciements

Mes tous premiers remerciements sont réservés à ceux sans qui tout cela n'aurait été qu'une belle histoire perdue dans le tas du possible :

- ⋆ À ma mère Faiza, celle qui mérite bien son nom de "gagnante", celle qui s'est donnée comme projet de vie la réussite de ses enfants, celle qui a tout sacrifié pour nous voir grandir, suivre ses pas, et la dépasser, celle qui nous a toujours guidé et accompagné par ses conseils et par ses invocations, celle à qui ces mots ne lui rendraient rien de l'hommage qui lui est dû ;
- ⋆ À mon père Kamel, celui qui a su choisir la mère de ses enfants, celui dont le soucis était toujours de nous voir exceller ; celui qui ne rate pas une occasion pour nous féliciter, ou nous tester.

Il va sans dire que cette thèse ne serait pas accomplie sans l'encadrement exceptionnel de mes quatre directeurs de thèse : Fabian, Sonia, Bernard, et Marcel. J'estime qu'il est d'une rareté extrême de voir se réunir autant d'encadrants aussi compétents et dévoués, tout en étant si bienveillants, pour accompagner un même doctorant. Je m'estime très chanceux d'avoir eu le privilège d'un tel encadrement. De tout cœur, je vous remercie pour votre patience, pour votre engagement exemplaire tout le long de cette thèse, pour toutes les relectures minutieuses, pour tous vos conseils, et pour avoir cru en moi depuis le début de la thèse et jusqu'à son aboutissement. Plus nominativement :

Je remercie Marcel pour m'avoir montré la voie de la recherche opérationnelle lorsque j'atterrissais de l'Extrême-Orient (de retour d'un échange académique) et que je me retrouvais dans un dense brouillard toulousain, et pour m'avoir ensuite proposé cette thèse. Marcel, merci pour toutes les relectures scrupuleuses et pour ta grande positivité.

Je remercie Fabian pour son accueil à l'Université de Montréal, pour son soutien tout au long de mon séjour à Montréal et bien au-delà. Un grand merci à toi, Fabian, pour m'avoir offert une belle expérience à Montréal, aussi riche scientifiquement que culturellement et humainement.

Je remercie Sonia pour son encadrement rapproché à Toulouse, pour sa confiance, et pour toutes les opportunités qu'elle a ouvertes pour moi. Sonia, je pense que tu peux te féliciter pleinement à chaque fois que l'un de tes étudiants arrive au bout du chemin.

Je remercie Bernard pour avoir accepté de prendre part à mon encadrement dès le début de ma thèse, pour ses idées ingénieuses, et pour sa touche d'expert. Bernard, ton cours de programmation linéaire en nombres entiers est l'un des cours les plus inspirants que j'ai suivi dans toute ma scolarité.

Je voudrais exprimer ma profonde gratitude aux professeurs Hamsa Balakrishnan et Alain Faye pour avoir accepté d'être rapporteurs pour ma thèse. Un grand merci à tous les membres du jury, Jean-Baptiste Hiriart-Urruty, Pierre l'Écuyer, André Rossi, et Claus Gwiggner, qui m'ont fait l'honneur d'assister à ma soutenance, de poser leurs questions,

et de me faire part de leurs appréciations.

Un remerciement particulier à Serge Roux, mon voisin de bureau à l'ENAC, pour sa bonne humeur, et son soutien humoristique et informatique qui était vital au bon déroulement de la thèse.

Je garderai à l'esprit mes collègues et camarades à l'ENAC et à l'Université de Montréal, avec lesquels j'ai partagé ne serait-ce qu'un bout du chemin : Mamadou T., Wyean C., Wael S., Isabelle S., Florian M., Vincent C., Ji M., Paolo S., Romaric B., Clément B., Gabriel J., Philippe M., Imen D., Michael T. et Almoctar H..

Ma sincère gratitude très fraternelle à toutes les personnes qui ont fait de mes séjours à Toulouse et à Montréal des expériences bien plus riches que celles d'un simple doctorant enfoui dans son bureau : Menouar A., Ramzi I., Brahim A., Ramzi M., Younes M., Youssef B., Youssef N., Achraf J., Khaled M., Aissa A., Ahmed A. B., Abdelkrim M., Ghazi M., Yacine B., et la liste est bien plus longue.

À ma sœur unique Zainab et à ma grande cousine Faten : un grand merci pour vous être chargées de la préparation du pot de thèse, qui était en lui-même une seconde réussite à mes yeux.

Ma dernière reconnaissance, et non des moindres, ainsi que mes excuses pour ma petite famille, qui est née et qui a grandi au courant de cette thèse ; mon épouse Yosra, et mes deux enfants, M. Wassim et Omar. Les meilleurs jours sont à venir.

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction

La croissance soutenue du trafic aérien mondial pendant la dernière décennie, avec des prévisions non moins optimistes pour les années à venir, place les acteurs du monde de l'aviation face à des enjeux de sécurité, d'efficacité et de capacité de plus en plus grands. En termes de capacité, un des principaux goulots d'étranglement du système de trafic aérien mondial se situe dans les phases transitoires d'atterrissage et de décollage, et moins dans la phase de vol stable où les avions sont en croisière, suite aux progrès technologiques et opérationnels récentes. Autrement dit, aujourd'hui, ce sont surtout les capacités aéroportuaires qui font défaut [68]. Une solution, évidente en apparence, pour augmenter la capacité aéroportuaire, exprimée en nombre de mouvements (atterrissages et/ou décollages) par heure, serait de construire de nouveaux aéroports ou d'augmenter le nombre de pistes dans les aéroports actuels. Or, de nos jours, de tels projets sont qualifiés de très difficiles voire d'impossibles [21], surtout dans les marchés les plus matures tels que l'Europe et l'Amérique du nord. Ces deux régions du monde, plus que les autres, s'attelleront plutôt à l'optimisation de l'utilisation de leurs infrastructures aéroportuaires actuelles.

Un tel objectif opérationnel global peut se décliner en différentes problématiques qui relèvent du domaine de la gestion de trafic aérien (*Air Traffic Management - ATM*), auquel nous donnons une brève introduction dans l'annexe A. En gestion du trafic aérien, trois niveaux de décision sont souvent distinguées : stratégique, pré-tactique, et tactique.

Le niveau stratégique s'étend sur un horizon temporel de plusieurs mois avant le vol jusqu'à 24 heures avant le décollage. À ce niveau, il est question d'adapter la demande de trafic aérien (dont les compagnies aériennes sont à l'origine, en demandant l'autorisation d'effectuer certains vols bien définis) à la capacité du système de trafic aérien, incluant celle de l'espace aérien (nombre maximal d'avions pouvant être gérés simultanément dans un volume prédéfini de l'espace aérien) et celle des aéroports. Un tel niveau de décision est de nature macroscopique : les vols peuvent être déplacés de plusieurs jours afin d'atteindre l'équilibre recherché entre la demande et la capacité.

Le niveau pré-tactique, dans la gestion de trafic aérien, correspond à la fenêtre de temps de 24 heures à 3 heures avant le décollage. Les vols peuvent être déplacés de plusieurs heures ou annulés, par exemple à cause de phénomènes météorologiques réduisant considérablement la capacité aéroportuaire.

Enfin, le niveau tactique correspond à l'horizon de 3 heures avant le décollage jusqu'à l'atterrissage. Les décisions possibles au cours d'un vol sont limitées à l'heure de décollage, la trajectoire spatio-temporelle suivie, et l'heure d'atterrissage. La modification des va-

leurs nominales est à la fois plus fine et plus limitée au niveau tactique par rapport aux autres niveaux de décision. Un tel niveau de décision est de nature microscopique. Dans le contexte de cette thèse, nous nous plaçons au niveau tactique de décision.

Selon de Neufville et al. [21, chap. 10], la capacité aéroportuaire, et plus précisément celle des pistes, est complexe à estimer car elle dépend de plusieurs facteurs, en plus du nombre et de la configuration des pistes. Parmi ces facteurs figure la séquence des avions à la piste, qui pour un même ensemble d'avions peut être plus ou moins longue selon l'ordre des opérations et les types des avions opérants. Il est donc primordial, au plan tactique, de minimiser la longueur de la séquence à la piste afin d'augmenter la capacité aéroportuaire. Ceci nous mène à considérer le problème d'optimisation, appelé *le problème d'ordonnancement des atterrissages* (*Aircraft Landing Problem - ALP*). En plus de déterminer la séquence des avions (et éventuellement les choix de piste dans un aéroport multi-pistes), ce problème consiste classiquement à affecter, aux avions se présentant à un aéroport, des heures d'atterrissage afin de maximiser la capacité des pistes, tout en respectant des contraintes opérationnelles, principalement de sécurité des vols. L'analogue de ce problème traitant les avions au parking prêts à décoller, est appelé *le problème d'ordonnancement des décollages* (*Aircraft Take-off Problem - ATP*). Plus généralement, le problème traitant conjointement les atterrissages et les décollages est appelé le *problème d'ordonnancement d'avions* (*Aircraft Scheduling/Sequencing Problem - ASP*).

D'un point de vue opérationnel, la sécurité et la fluidité du trafic aérien dans une zone bien définie de l'espace aérien est de la responsabilité des contrôleurs aériens. Ces aiguilleurs du ciel communiquent aux pilotes, traversant leur zone de responsabilité, des actions de contrôle que ces derniers sont tenus d'appliquer. Il est, ainsi, du ressort des contrôleurs aériens de guider les avions à former une séquence à la piste, non seulement garantissant la sécurité des vols mais aussi maximisant la capacité des pistes. Pour aider les contrôleurs à ordonnancer les atterrissages d'avions au(x) seuil(s) de(s) piste(s), de nos jours, des outils d'aide à la décision, appelés AMAN (*Arrival MANager*) [36], ont été mis en place en Europe comme aux États-Unis depuis la fin des années 1980. Généralement, ces outils captent les avions à 30–45 minutes de l'atterrissage (à moins de 100 *miles nautiques* environ) [70], ensuite résolvent le problème d'ordonnancement afin de fournir aux contrôleurs la séquence et les heures cibles *optimales* à la piste. Les contrôleurs ont le choix des actions de contrôle afin de mener les avions à respecter cette séquence : par exemple, accélération ou décélération des avions, raccourcissement ou rallongement de trajectoire, *etc.*

Pour gérer des situations de fort trafic aérien, les aiguilleurs du ciel peuvent recourir aux circuits d'attente (*holding patterns*), consistant à maintenir les avions, séparés en altitude, en vol circulaire au-dessus de points prédéfinis de l'espace aérien à proximité de l'aéroport de destination. Cette attente, qualifiée de *circulaire*, est dépréciée des compagnies aériennes pour ses coûts élevés de consommation de carburant. Les contrôleurs tentent également d'éviter ce type d'attente à cause de son risque pour la sécurité et sa charge de travail élevée. Par opposition, la réduction de vitesse subie par les avions dans leur phase de croisière, appelée attente *linéaire*, présente un faible coût pour les compagnies aériennes ainsi qu'une charge de travail plus acceptable pour les contrôleurs. Cette remarque opérationnelle a motivé l'extension des horizons opérationnels des outils d'aide à la décision jusqu'à 500 miles nautiques (à quelques heures de l'atterrissages) afin de limiter au maximum l'attente circulaire au profit de l'attente linéaire [70].

Ceci rend d'actualité le problème d'ordonnancement des avions où l'horizon opérationnel est étendu. L'objectif final est d'optimiser l'utilisation des pistes sans compliquer la tâche des contrôleurs aériens. La fonction objectif peut être exprimée sous plusieurs formes en fonction des points de vue des acteurs considérés. Toutefois, elle prend souvent la forme de la minimisation du coût du retard ou la minimisation de l'heure du dernier mouvement ordonnancé, c'est-à-dire la maximisation du rendement des pistes, correspondant au point de vue des contrôleurs. Les principales contraintes opérationnelles sont des contraintes de séparation entre les paires d'avions afin d'en garantir la sécurité. La séparation minimale entre deux avions dépend de leurs types, leurs performances, leurs mouvements respectifs (atterrissage ou décollage) mais aussi des conditions météo. Ces séparations minimales forment la principale limitation à la capacité des pistes.

Dans la littérature, l'attention est principalement tournée vers le problème d'ordonnancement des atterrissages d'avions, avec un rayon opérationnel réduit, depuis la publication du premier algorithme de séquencement d'avions [22] il y a plus de quatre décennies. Ainsi, aussi bien le cas statique (la séquence initiale des avions est complètement connue) que le cas dynamique (la séquence initiale des avions est révélée progressivement au cours du temps) ont été traités. Le panel des méthodes de résolution appliquées est très large allant des méta-heuristiques (algorithmes génétiques, recuit simulé, recherche tabou, etc) aux méthodes exactes (programmation dynamique, branch-and-bound, branch-and-price, etc) en passant par des heuristiques. Toutefois, très peu d'auteurs se sont intéressés au problème d'ordonnancement des avions en présence d'incertitude. Dans le contexte de l'ordonnancement des arrivées d'avions avec un horizon opérationnel étendu, le caractère stochastique doit être nécessairement pris en compte, vu que les heures prévues des arrivées d'avions ne sont pas connues avec certitude à quelques heures de l'atterrissage [67, 70].

Cette thèse traite du problème d'ordonnancement des arrivées d'avions avec prise en compte de l'incertitude dans le contexte d'un horizon opérationnel étendu, en utilisant le paradigme de la programmation stochastique à deux étapes.

Dans la suite de cette introduction, nous revisitons les principales contraintes opérationnelles du problème d'ordonnancement des atterrissages qui sont les minima de séparation. Ensuite, nous donnons un aperçu de l'outil d'aide à la décision européen AMAN et de sa version avec un horizon opérationnel étendu, E-AMAN. Comme il est question d'appliquer le paradigme de la programmation stochastique à deux étapes à notre problème opérationnel, une brève introduction à ce paradigme est donnée, suivie d'une présentation de la décomposition de Benders (une méthode de choix pour la résolution des programmes stochastiques à deux étapes). Enfin, le lecteur est invité à se diriger vers les annexes A, B afin d'apprendre davantage sur le vocabulaire et les concepts opérationnels évoqués dans ce document.

NM : mile nautique, mesure de distance (1 NM = 1852 m)
kts : nœud, mesure de vitesse (1 kts = 1 NM/h)

Table des matières

1.1 Minima de séparation et turbulences de sillage

Afin de garantir la sécurité des vols, les services de contrôle aérien doivent s'assurer que les avions dans les espaces aériens contrôlés évoluent dans des volumes propres définis par des normes de séparation, longitudinale et verticale. Les deux normes de séparation les plus connues sont la *séparation radar* et la *séparation de turbulence de sillage* [30]. Nous expliquons chacune de ces séparations dans la suite.

1.1.1 Séparation radar

Elle s'applique dans la phase de vol en-route, où la distance de séparation verticale est de 1000 pieds (environ 300 mètres) et la distance horizontale est de 5 NM (3 NM dans les espaces congestionnés). La figure 1.1 représente le volume de séparation radar autour d'un avion.

Au seuil de piste, et plus généralement à la phase d'approche finale, les séparations de turbulence de sillage sont les plus contraignantes.

1.1.2 Séparation de turbulence de sillage

Suivant ses dimensions et sa masse, un aéronef en décollage, en croisière ou en atterrissage génère derrière lui des tourbillons d'air (ou turbulences), souvent invisibles, pouvant durer jusqu'à plusieurs minutes et s'étendre sur plusieurs miles nautiques. Ces turbulences, quoique de courte durée, déstabilisent les aéronefs qui les reçoivent menaçant leur sécurité. Les effets des turbulences de sillage sont plus notables dans les phases d'atterrissage et de décollage. Pour cette raison, le contrôle aérien est tenu d'appliquer une norme longitudinale de séparation entre les paires d'avions dont la valeur dépend directement de leurs catégories de turbulence.

L'Organisation de l'Aviation Civile Internationale (OACI) définit quatre catégories d'avions selon leur masse maximale au décollage :

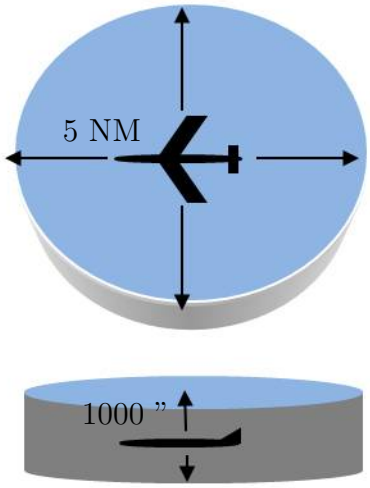


FIGURE 1.1 – Cylindre de séparation en-route

— **très forte :** *Super Heavy - (S)*[1]
— **forte :** *Heavy - (H)*
— **moyenne :** *Medium - (M)*
— **faible :** *Light - (L)*

En s'appuyant principalement sur cette catégorisation, les services de contrôle aérien en approche assurent une séparation minimale entre toutes les paires d'avions en décollage et/ou en atterrissage afin de prévenir tout incident ou accident qui serait causé par les turbulences de sillage. Plus précisément, ces séparations minimales dépendent non seulement des catégories de turbulence de sillages de l'avion meneur et de l'avion suiveur (appelées aussi classes de turbulence de sillage) mais aussi de leurs mouvements respectifs (atterrissage ou décollage) et des conditions météorologiques.

Concernant les atterrissages, ces séparations sont d'abord indiquées en distance (NM) mais souvent converties en temps en utilisant des vitesses représentatives d'atterrissage [21, 66]. La table 1.1 donne un exemple de ces séparations exprimées en miles nautiques.

		Avion suiveur		
		H	M	L
Avion meneur	H	4	5	6
	M	2,5	2,5	4
	L	2,5	2,5	2,5

TABLE 1.1 – Matrice de séparation au seuil de piste (en NM) selon les catégories de turbulence de sillage de l'OACI (de Neufville et al. [21])

Pour les décollages, les minima de séparation sont directement spécifiés en temps par les autorités d'aviation civile. Finalement, d'autres séparations minimales temporelles existent pour le cas d'un décollage et d'un atterrissage qui se succèdent sur une même piste. Pour information, nous présentons dans la table 1.2, une matrice de séparation temporelle sur une piste unique où les quatre cas de figures apparaissent : deux atterrissages, deux décollages, un atterrissage suivi par un décollage et l'inverse.

1. La catégorie des très fortes turbulences (*Super Heavy*) contient uniquement l'Airbus A380.

			Avion suiveur					
			atterrissage			décollage		
			H	M	L	H	M	L
Avion meneur	atterrissage	H	96	157	196	75	75	75
		M	60	69	131	75	75	75
		L	60	69	82	75	75	75
	décollage	H	60	60	60	90	120	120
		M	60	60	60	60	60	60
		L	60	60	60	60	60	60

TABLE 1.2 – Matrice des séparations minimales temporelles (secondes) sur une piste unique entre les quatre combinaisons d'opérations possibles : deux atterrissages, deux décollages, un atterrissage suivi d'un décollage et vice versa [51].

Re-catégorisation Jugée obsolète et trop protectrice, la catégorisation de l'OACI a été révisée par EUROCONTROL [2] dans le cadre du projet RECAT-EU [3]. Grâce à une meilleure compréhension du phénomène de tourbillon de sillage et de son effet sur les avions, une nouvelle catégorisation plus fine a été établie :

— *Super Heavy - (SH)*
— *Upper Heavy - (UH)*
— *Lower Heavy - (LH)*
— *Upper Medium - (UM)*
— *Lower Medium - (LM)*
— *Light - (L)*

Les minima de séparation révisés apparaissent dans le tableau 1.3, où les tirets "-" correspondent au minimum de séparation réglementaire (2,5 ou 3 NM selon les cas). Des minima mieux étudiés permettent d'augmenter la capacité des pistes actuelles.

		Avion suiveur					
		SH	UH	LH	UM	LM	L
Avion meneur	SH	-	6	6	7	7	8
	UH	-	3	4	5	5	6
	LH	-	4	3	5	5	6
	UM	-	-	-	-	-	5
	LM	-	-	-	-	-	5
	L	-	-	-	-	-	-

TABLE 1.3 – Matrice de séparation au seuil de piste (en NM) selon RECAT-EU d'EUROCONTROL

Quant au projet RECAT 2 [4] d'EUROCONTROL, il a pour objectif de définir des séparations encore plus fines, propres à chaque paire d'avions, selon les types des appareils. Ce projet donnera lieu à une matrice de séparation de taille 100×100 environ.

2. EUROCONTROL est l'organisation européenne pour la sécurité de la navigation aérienne. "Sa mission est d'harmoniser et d'unifier la gestion de la navigation aérienne en Europe" (source : `fr.wikipedia.org/wiki/Eurocontrol`).
3. `http://recat-project.eu/solutions/recat-eu`
4. `http://recat-project.eu/solutions/next-steps/recat-2-pair-wise-separations`

1.2 Outils d'aide à la décision en ordonnancement des avions

***Center TRACON Automation System* (CTAS)** est l'outil pionnier d'aide à la décision en ordonnancement des arrivées conçu par la NASA à la fin des années 1980 [58, 72]. Les outils équivalents européens ont émergé rapidement [31] et sont de nos jours appelés *Arrival Manager* (AMAN) [5].

***Arrival Manager* (AMAN)** et ***Departure Manager* (DMAN)** sont des outils d'aide à la décision destinés aux contrôleurs aériens (en approche et à la tour de contrôle) pour le séquencement des atterrissages et des décollages respectivement. De nos jours, plusieurs aéroports européens sont équipés d'outils AMAN et DMAN.

En raison de l'importance de l'enjeu de l'ordonnancement des atterrissages comparé à celui des décollages, les systèmes AMAN se sont développés plus tôt et ont reçu plus d'intérêt que les systèmes DMAN. Il en résulte qu'à l'heure actuelle les systèmes AMAN sont considérés plus matures. L'intégration de AMAN et DMAN est prévue dans le cadre du programme européen SESAR [6]. En France, le système AMAN mis en place s'appelle MAESTRO, délivré par Thales Air System SAS. Il équipe, parmi d'autres, l'aéroport de Paris Charles de Gaulle (CDG).

1.2.1 AMAN

AMAN peut être décomposé en trois modules comme schématisé dans la figure 1.2 :

Prédiction de trajectoire : Ce module intègre des modèles de performance avion, les plans de vols, les données radar et les données météo pour estimer l'heure d'approche sans contraintes (*Expected Approach Time - EAT*) de chaque avion.

Séquencement : les avions sont généralement séquencés selon leur ordre d'arrivée "premier arrivé, premier servi". D'autres règles peuvent être appliquées telles que l'équité, la distribution du retard ou le groupement par classe de turbulence. Dès que deux avions ne satisfont pas les critères de séparation, la séquence est révisée. Il en résulte de nouvelles heures d'approche sous contraintes.

Recommandations aux contrôleurs : La différence entre les heures d'approche sans et avec contraintes est calculée par avion en termes de temps à gagner (*Time To Gain - TTG*) ou de temps à perdre (*Time To Lose - TTL*). Certaines implémentations sophistiquées d'AMAN proposent des suggestions de manœuvres (par exemple, guidage radar, changement de vitesse) pour les contrôleurs.

1.2.2 E-AMAN

E-AMAN (Extended AMAN) est une version de AMAN dont l'horizon opérationnel a été étendu au-delà de la région de contrôle (*Terminal Maneuvring Area - TMA*). Le rayon maximal de ce nouvel outil d'aide à la décision peut atteindre 500 NM [41]. L'intérêt de cette extension est de pouvoir commencer à ordonnancer les avions à destination du même

5. `https://www.eurocontrol.int/sites/default/files/article/content/documents/nm/fasti-aman-guidelines-2010.pdf`

6. Single European Sky ATM Research (SESAR) est le programme de modernisation de la gestion de trafic aérien en Europe.

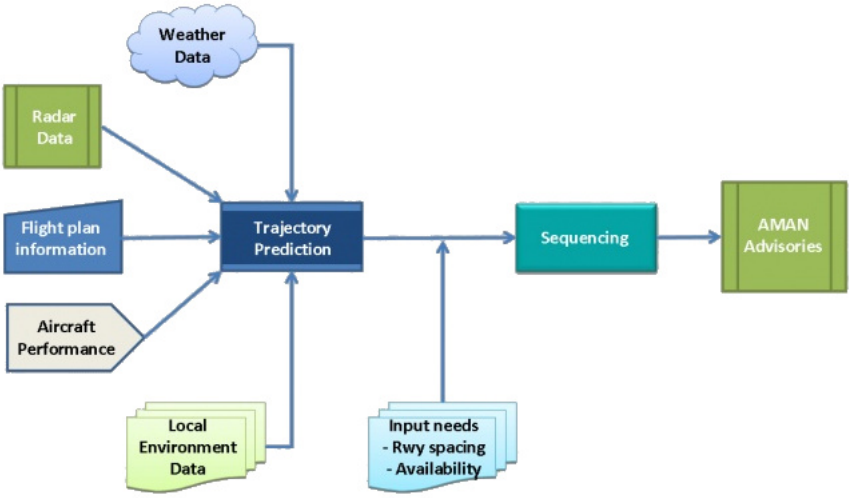


FIGURE 1.2 – Modules AMAN (source :`https://www.skybrary.aero/index.php/Arrival_Manager_(AMAN)`)

aéroport pendant la phase d'en-route, par exemple en ajustant convenablement leurs vitesses de croisière. Ceci a pour avantage de minimiser et idéalement éliminer l'attente circulaire, au niveau des circuits d'attente situés à la frontière de la TMA.

XMAN (Cross-border AMAN) est l'appellation donnée à AMAN quand son rayon d'action englobe plusieurs pays. Dans le contexte européen, les E-AMAN doivent être également des XMAN.

La figure 1.3 montre des horizons opérationnels d'un AMAN actuel, d'un XMAN et d'un E-AMAN gérant les arrivées sur la capitale britannique Londres.

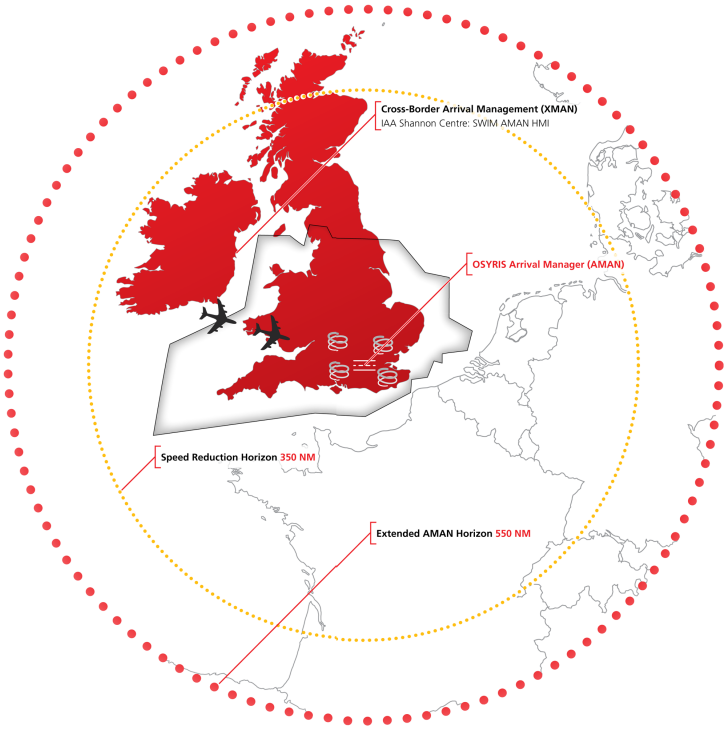


FIGURE 1.3 – Horizons opérationnels de AMAN, E-AMAN et XMAN au Royaume-Uni [59]

1.3 Programmation stochastique à deux étapes

La programmation stochastique est un paradigme de modélisation de problèmes d'optimisation prenant en compte l'incertitude sur certaines données d'entrée du problème. Ces données incertaines sont supposées des variables aléatoires dont les lois de probabilité sont connues à l'avance.

La programmation stochastique à deux étapes distingue deux étapes de décision. Dans la première étape, les valeurs des données incertaines sont considérées inconnues. Une décision, dite *de première étape*, doit être prise afin d'optimiser un certain critère de première étape. Dans la deuxième étape, l'incertitude est supposée levée ; les données incertaines sont alors connues avec certitude. Ainsi, un nouveau problème d'optimisation – déterministe –, dit *de deuxième étape*, est formé. Une nouvelle décision, dite également *de deuxième étape* ou *de recours*, doit être prise afin d'optimiser un critère de deuxième étape. Remarquons que la définition du problème de deuxième étape dépend des valeurs incertaines révélées. Ainsi, chaque levée de l'incertitude peut donner lieu à un problème d'optimisation de deuxième étape différent.

La programmation stochastique à deux étapes permet de lier les problèmes de première et de deuxième étape en considérant la première étape comme une étape maîtresse et la deuxième comme une sous-étape dépendante des décisions de la première étape et de la levée de l'incertitude. Notons qu'une telle structure se retrouve également dans l'optimisation bi-niveau [4]. La fonction objectif d'un problème stochastique à deux étapes s'exprime, classiquement, comme la somme du critère de première étape et l'*espérance* du critère de deuxième étape.

La solution optimale d'un tel problème est une décision de première étape *uniquement*, dite une *décision a priori*, qui optimise cette fonction objectif bi-critère. Cette décision *a priori* a l'avantage de ne pas être myope par rapport à la deuxième étape, mais surtout d'être la solution permettant d'obtenir le meilleur "coût *en moyenne*", si le problème de deuxième étape serait à formuler et à résoudre un grand nombre de fois (selon la loi des grands nombres). Aussi faut-il préciser qu'il n'est pas question dans la programmation stochastique à deux étapes de fournir une "politique" de décision qui indiquerait, en plus de la décision optimale de la première étape, la décision optimale de la deuxième étape en fonction de la décision de la première étape et de la réalisation de l'incertitude (comme, par exemple, dans un processus de décision de Markov ou dans la programmation dynamique stochastique [11, chap. 2.10]).

En toute généralité, une décision de recours (ou de deuxième étape) peut représenter une révision de la décision de première étape ou bien une nouvelle décision correspondant à un nouveau problème d'optimisation, qui émerge "naturellement" suite à la révélation de l'incertitude. Dans la suite, nous rapportons deux exemples de la littérature représentant les deux points de vue.

Problème du vendeur de journaux : En première étape, ne connaissant pas la demande de sa clientèle du lendemain, le vendeur doit décider de la quantité de journaux à approvisionner (décision de première étape). Le lendemain, le profit est fonction du minimum entre son stock et la demande observée, mais il peut éventuellement revendre, à perte, le surplus de journaux (décision de deuxième étape) si son stock procuré la veille excède la demande observée.

Problème du fermier : Dans l'exemple du fermier, présenté dans [11], la décision de première étape consiste à décider des surfaces à planter pour chaque type de culture (blé, canne à sucre, etc), le rendement étant inconnu au préalable. Après la récolte (deuxième étape), le fermier doit décider des quantités à vendre sur le marché et de celles à conserver ou à se procurer pour satisfaire la consommation de ses bétails. Le problème de deuxième étape n'a pas de sens tant que la récolte n'a pas eu lieu. Il est plutôt un nouveau problème qui apparaît après révélation de l'incertitude.

La révélation de l'incertitude donne lieu à différents scénarios, dont le nombre peut être arbitrairement grand, voire infini dans le cas de variables aléatoires continues par exemple.

La programmation stochastique à deux étapes ne cherche pas à se protéger contre les scénarios extrêmes du futur (contrairement à l'optimisation robuste, par exemple) mais plutôt à fournir des décisions de première étape permettant d'optimiser en espérance les critères de première et de deuxième étape sur l'ensemble des scénarios considérés.

Il important de préciser qu'optimiser l'espérance du critère de deuxième étape n'est pas équivalent à optimiser ledit critère sur le scénario espéré (ou moyen). D'ailleurs, l'écart entre les valeurs des solutions optimales de ces deux problèmes d'optimisation constitue une quantité importante en programmation stochastique, appelée la *valeur de la solution stochastique*.

Dans la suite de cette section, nous présentons la forme générale d'un programme stochastique à deux étapes, ainsi que différentes notions générales telles que les types de recours et la valeur de la solution stochastique. Ensuite, nous présentons la méthode d'approximation par moyenne empirique, ou SAA ("Sample Average Approximation"), utilisée sur des problèmes stochastiques avec un grand nombre de scénarios. Finalement, nous abordons les méthodes de résolution pour les programmes stochastiques, notamment la décomposition de Benders.

1.3.1 Formulation et notions générales

Soit m_1, m_2, n_1, n_2 et r des entiers naturels. Soit $x \in \mathbb{R}^r \cup \mathbb{Z}^{n_1-r}$ un vecteur de variables de décision, A une matrice de $\mathbb{R}^{m_1} \times \mathbb{R}^{n_1}$ et b un vecteur de $\mathbb{R}^{m_1}$. Soit $\boldsymbol{\omega}$ un vecteur de variables aléatoires de lois connues. Une réalisation d'un tel vecteur est notée ω . Soit f_1 une fonction linéaire définie sur $\mathbb{R}^{n_1}$ à valeurs dans $\mathbb{R}$. Soit $y \in \mathbb{R}^{n_2}$ un vecteur de variables de décision, $B(\omega)$ une matrice de $\mathbb{R}^{m_2} \times \mathbb{R}^{n_1}$, C une matrice de $\mathbb{R}^{m_2} \times \mathbb{R}^{n_2}$, et $d(x, \omega)$ un vecteur de $\mathbb{R}^{m_2}$. Soit f_2 une fonction linéaire en y et à valeurs dans $\mathbb{R}$. Enfin, notons $\mathbb{E}_{\boldsymbol{\omega}}[.]$ l'opérateur de l'espérance relativement à la variable aléatoire $\boldsymbol{\omega}$.

Le programme suivant (2SSP) est un programme stochastique à deux étapes, où le vecteur x représente les décisions de première étape, et le vecteur y les décisions de deuxième étape :

$$
\begin{aligned}
&\min_{x} && f_1(x) + \mathbb{E}_{\boldsymbol{\omega}}[Q(x, \omega)] && \text{(2SSP)}\\
&\text{s.c.} && Ax = b\\
& && x \in \mathbb{R}^r \cup \mathbb{Z}^{n_1-r}\\
&\text{avec : } Q(x, \omega) = \min_{y} && f_2(x, y, \omega)\\
& && B(\omega)\, x + Cy = d(x, \omega),
\end{aligned}
$$

où $Q(x, \omega)$ représente la valeur optimale de la fonction objectif du problème de deuxième étape, défini pour une solution de première étape x et une réalisation ω du vecteur aléatoire. Tel que présenté, le problème de première étape est un problème à variables mixtes, alors que le problème de deuxième étape est un problème linéaire à variables continues.

Type de recours : Selon la forme du problème de deuxième étape, il est possible que certaines solutions de première étape puissent donner des problèmes de deuxième étape non réalisables. Ceci nous amène à distinguer différents types de problèmes de deuxième étape, appelés aussi *types de recours* :

recours complet : un recours est dit complet si le problème de deuxième étape admet des solutions réalisables, qu'il y ait ou non des solutions pour le problème de première étape.

recours relativement complet : un recours est dit relativement complet si le problème de deuxième étape admet des solutions réalisables pour toute solution réalisable du problème de première étape.

recours fixe : le recours fixe correspond au cas où la matrice des contraintes C ne dépend pas de x et ω (comme présentée ici).

recours simple : le recours simple correspond au cas où la matrice des contraintes C a une structure particulière : $C = (I_{m_2} \quad -I_{m_2})$, où I_{m_2} est la matrice identité définie sur $\mathbb{R}^{m_2} \times \mathbb{R}^{m_2}$. Notons également que le recours simple est un cas particulier du recours complet. Le problème du vendeur des journaux est un exemple d'un problème stochastique à deux étapes avec un recours simple [11].

Notons $\mathcal{Q} : x \mapsto \mathbb{E}_{\boldsymbol{\omega}}[Q(x, \omega)]$ la fonction de *recours minimum espéré*. Alors, la projection de (2SSP) sur l'espace des x s'écrit :

$$\begin{aligned} \min_{x} \quad & f_1(x) + \mathcal{Q}(x) \\ & Ax = b \end{aligned} \tag{Proj-2SSP}$$

Théorème 1.3.1. *Dans le cas de la programmation stochastique linéaire à deux étapes (i.e., les problèmes de première et de deuxième étape sont linéaires en leurs variables de décision, x et y respectivement) avec recours fixe (la matrice C ne dépend pas de x et ω), alors $\mathcal{Q}$ est convexe en x [11].*

Le théorème 1.3.1 exprime le fait que la programmation stochastique linéaire avec recours fixes s'inscrit bien sous le volet de la programmation convexe. Toutefois, le calcul de $\mathcal{Q}(x)$ est souvent très difficile et nécessite le calcul d'une intégrale multiple [11, chap. 3.1]. Dans la suite, nous traitons les deux cas possibles des valeurs des variables aléatoires selon leur cardinalité (dénombrables ou non dénombrables), et nous présentons comment le calcul, ou à défaut l'approximation, de $\mathcal{Q}(x)$ est possible.

Cas de variables aléatoires discrètes à support fini : Dans le cas de variables aléatoires discrètes à support fini, le calcul de l'espérance se résume à un calcul de moyenne. Notons $\Omega = \{\omega_1, \omega_2 \ldots, \omega_K\}$ le support de la variable aléatoire $\boldsymbol{\omega}$, où chaque réalisation ω_k, pour $k \in \{1, \ldots, K\}$, a une probabilité p_k. Rappelons qu'à chaque réalisation ω_k et pour un vecteur de décision de première étape donné, x, un problème k de deuxième

étape est formulé. Notons y_k le vecteur de décision pour ce problème k de deuxième étape. Alors, (2SSP) peut être réécrit comme suit :

$$\begin{aligned} \min_{x} \quad & f_1(x) + \sum_{k=1}^{K} p_k \left(\min_{y_k} f_2(x, y_k, \omega_k) \right) \\ \text{s.c.} \quad & Ax = b \\ & B(\omega_k)\, x + C y_k = d(x, \omega_k) \ , \quad k \in \{1, \dots, K\} \end{aligned}$$

Ou encore :

$$\begin{aligned} \min_{\substack{x, y_k \\ k=1,\dots,K}} \quad & f_1(x) + \sum_{k=1}^{K} p_k f_2(x, y_k, \omega_k) \qquad \text{(Determ. Eq.)} \\ \text{s.c.} \quad & Ax = b \\ & B(\omega_k)\, x + C y_k = d(x, \omega_k) \ , \quad k \in \{1, \dots, K\} \end{aligned}$$

Cette dernière formulation (Determ. Eq.), appelée l'*équivalent déterministe*, se présente comme un programme linéaire déterministe (à une seule étape), comprenant K copies du problème de deuxième étape, et donc potentiellement de grande taille. La résolution de (Determ. Eq.) peut être envisagée par Branch-and-Cut via un solveur générique de programmes linéaires en variables mixtes (tel que CPLEX). Néanmoins, quand le nombre de scénarios augmente, (Determ. Eq.) peut devenir de très grande taille et sa résolution classique par Branch-and-Cut sera inefficace. Heureusement, sa matrice des contraintes est creuse, présentant des colonnes liantes ($B(\omega_k)\, x$) et des blocs séparables par scénario (Cy_k pour $k \in \{1, \dots, K\}$). Une telle structure se prête bien à une *décomposition de Benders* avec des sous-problèmes séparables par scénario. Quand elle est appliquée à la programmation stochastique à deux étapes, la décomposition de Benders est souvent évoquée sous le nom de l'algorithme *L-Shaped* [11, 71]. Dans la sous-section 1.4, nous présentons la décomposition de Benders, ainsi que son application à la programmation stochastique à deux étapes.

Cas de variables aléatoires à support infini (ou de très grande taille) : Quand le support des variables aléatoires est infini ou de très grande taille, l'approximation par moyenne empirique, appelée *SAA* (pour *Sample Average Approximation*), permet de réduire le calcul de l'espérance au calcul d'une moyenne sur un ensemble fini de scénarios, appelé un *échantillon*. Cette méthode sera présentée dans la Sous-section 1.3.3. En utilisant la méthode SAA, le programme stochastique résultant, dit *programme d'approximation*, présente une structure similaire à (Determ. Eq.), suggérant ainsi l'utilisation des mêmes techniques de résolution.

Pour résumer, la programmation stochastique à deux étapes permet de modéliser un problème d'optimisation comprenant deux étapes de décision entre lesquelles des données incertaines, supposées suivre des lois de probabilité connues, sont révélées. Le problème de deuxième étape dépend de l'incertitude révélée. La fonction objectif à optimiser est la somme du critère de la première étape et de l'espérance du critère de la deuxième étape. Le cas de la programmation stochastique linéaire à deux étapes avec recours fixe relève de l'optimisation convexe. Un programme stochastique linéaire à deux étapes, formulé sur un ensemble fini de scénarios, présente une structure en blocs séparables par scénario

avec des variables liantes. Une telle structure est favorable à une résolution efficace par décomposition de Benders.

Quels que soient les détails de la modélisation, un programme stochastique est souvent plus difficile à résoudre qu'un programme déterministe où les données incertaines sont réduites à leurs valeurs moyennes. Pour un décideur, il est important de pouvoir justifier son choix de modélisation de l'incertitude comme des variables aléatoires et son choix du paradigme de la programmation stochastique. La prise en compte de l'incertitude via la formulation d'un programme stochastique à deux étapes peut être justifiée en calculant une quantité appelée la *valeur de la solution stochastique*. Nous en donnons la définition dans la suite.

1.3.2 Valeur de la solution stochastique

En présence d'incertitude sur les données d'entrée, le décideur peut se poser la question sur la pertinence de la prise en compte de l'incertitude via des variables aléatoires (de lois connues) face à la réduction des données incertaines à leurs valeurs moyennes. La première alternative conduit le décideur à formuler un problème stochastique, éventuellement à deux étapes tel qu'il est le cas dans le présent manuscrit. Un tel choix de modélisation peut avoir l'avantage de mieux représenter le problème réel. La deuxième alternative amène à la formulation d'un problème d'optimisation déterministe, utilisant uniquement les valeurs moyennes des données, appelé *problème du scénario moyen* (ou "*Expected Value Problem*" en anglais).

Afin de quantifier l'avantage de la modélisation plus fine de l'incertitude par des variables aléatoires, il est possible de calculer *la valeur de la solution stochastique*, notée VSS. D'une part, par définition, une solution stochastique, notée x_{SP}, est une solution optimale obtenue en résolvant le problème stochastique à deux étapes :

$$x_{\mathrm{SP}} \in \arg\min \left\{ f_1(x) + \mathbb{E}_{\boldsymbol{\omega}} \left[Q(x, \omega) \right] \ | Ax = b \right\} \tag{1.1}$$

Notons v_{SP} la valeur de la fonction objectif correspondant à la solution stochastique x_{SP}.

D'autre part, notons x_{EV} la solution du problème du scénario moyen (appelée aussi solution *déterministe*), définie comme suit :

$$x_{\mathrm{EV}} \in \arg\min \left\{ f_1(x) + Q\left(x, \mathbb{E}\left[\boldsymbol{\omega}\right]\right) \ | Ax = b \right\} \tag{1.2}$$

La valeur de la solution déterministe, notée v_{EV}, correspond à l'évaluation de x_{EV} sur le problème stochastique à deux étapes. Plus formellement, supposons que $A\, x_{\mathrm{EV}} = b$ et $\mathcal{Q}(x_{\mathrm{EV}}) < \infty$ (i.e., x_{EV} est une solution réalisable de (2SSP)), alors v_{EV} est définie comme suit :

$$v_{\mathrm{EV}} = f_1(x_{\mathrm{EV}}) + \mathcal{Q}(x_{\mathrm{EV}}) \tag{1.3}$$

La valeur de la solution stochastique, VSS, est définie comme la différence absolue entre la valeur optimale du problème stochastique à deux étapes, v_{SP}, et l'évaluation de la solution du problème du scénario moyen sur le problème stochastique à deux étapes, v_{EV} :

$$\mathrm{VSS} = v_{\mathrm{SP}} - v_{\mathrm{EV}} \tag{1.4}$$

1.3.3 Approximation par moyenne empirique

Nous présentons ici une brève introduction à la méthode d'approximation par moyenne empirique (de l'anglais : *Sample Average Approximation - SAA*) est tirée de [29, chap. 8] et de [64]. Nous présenterons trois résultats de convergence de la méthode SAA qui sont utilisés dans les chapitres 3 et 4 de ce manuscrit.

Dans le contexte de la programmation stochastique à deux étapes, le terme d'espérance, $\mathbb{E}_{\boldsymbol{\omega}}[Q(x,\omega)]$, peut être approximé par une moyenne empirique sur un échantillon donné de $n_{\mathcal{S}}$ scénarios, $\mathcal{S} = \left\{\omega^1, \ldots, \omega^{n_{\mathcal{S}}}\right\}$. Soit le programme suivant, appelé *programme d'approximation SAA*, défini pour l'échantillon $\mathcal{S}$:

$$\begin{aligned} \min_{x} \quad & f_1(x) + \frac{1}{n_{\mathcal{S}}} \sum_{s=1}^{n_{\mathcal{S}}} Q(x,\omega^s) && \text{(Approx-2SSP)} \\ \text{s.c.} \quad & Ax = b \\ \text{avec : } Q(x,\omega) = \min_{y} \quad & f_2(x,y,\omega) \\ & B(\omega)\,x + Cy = d(x,\omega) \end{aligned}$$

Notons $v^\star$ la valeur optimale du problème d'origine (2SSP). Notons $\hat{v}(n_{\mathcal{S}})$ la valeur optimale de (Approx-2SSP), pour un échantillon $\mathcal{S}$ quelconque de taille $n_{\mathcal{S}}$. Remarquons qu'il est plus correct de noter la valeur optimale de (Approx-2SSP) comme $\hat{v}(\mathcal{S})$. Mais, afin de présenter les résultats de convergence, la quantité d'intérêt est la taille de l'échantillon $n_{\mathcal{S}}$, plus que l'échantillon même. Remarquons que $\hat{v}(n_{\mathcal{S}})$ est une variable aléatoire. Nous avons le résultat de convergence suivant :

Théorème 1.3.2 (Théorème 5.3, [64])**.** *$\hat{v}(n_{\mathcal{S}}) \to v^\star$ quand $n_{\mathcal{S}} \to \infty$*

Théorème 1.3.3 (Proposition 8.6, [29], Proposition 5.6, [64])**.** *Pour tout $n_{\mathcal{S}} \geq 1$, nous avons :*

- $\mathbb{E}[\hat{v}(n_{\mathcal{S}})] \leq \mathbb{E}[\hat{v}(n_{\mathcal{S}}+1)]$
- $\mathbb{E}[\hat{v}(n_{\mathcal{S}})] \leq v^\star$

Rappelons que pour deux fonctions $f : n \mapsto f(n)$ et $g : n \mapsto g(n)$, f est un grand "O" de g, noté $f = \mathcal{O}(g)$, veut dire que f ne croît pas plus rapidement que g, quand $n \to \infty$. Quand $f(n) \to l$ et $g(n) \to l$ pour $n \to \infty$, $f = \mathcal{O}(g)$ signifie que f converge vers l *au mieux* aussi rapidement que g.

Sous des hypothèses faibles de régularité, nous avons le théorème suivant sur le taux de convergence de la méthode SAA (comme une application du théorème 5.7, [64]) :

Théorème 1.3.4. *Le biais $\mathbb{E}[\hat{v}(n_{\mathcal{S}})] - v^\star$ est de l'ordre de $\mathcal{O}\left(n_{\mathcal{S}}^{-1/2}\right)$.*

1.4 Décomposition de Benders

La décomposition de Benders est une méthode de résolution de programmes linéaires en variables mixtes, qui a été d'abord proposée par [7]. Cette méthode est motivée par la remarque que dans un programme linéaire en variables mixtes, si les variables entières sont fixées, alors le problème se réduit à un programme linéaire, souvent facile à résoudre. Le problème en variables mixtes peut alors être partitionné en deux problèmes : un *problème maître de Benders*, comprenant toutes les variables entières, et un *sous-problème de Benders*, comprenant les variables continues. Une variable continue, dite *de reformulation*, est

ajoutée à la fonction objectif du problème maître afin de modéliser la valeur optimale du sous-problème. Le schéma de résolution classique d'un tel programme selon cette partition est un schéma itératif, où à chaque itération il y a résolution du problème maître, ensuite résolution du sous-problème, et enfin une génération de coupes, dits *de Benders*, à intégrer au problème maître.

Dans la suite, nous présentons le principe de la reformulation de Benders et de la résolution par décomposition de Benders. Ensuite, nous donnons quelques techniques d'accélération de la décomposition de Benders. Le lecteur intéressé peut se référer à la revue de littérature [62].

1.4.1 Principe

Les notations dans cette section sont indépendantes du reste du manuscrit. Notons (ORG) un programme linéaire en variables mixtes, défini comme suit :

Entrées : Soient n, n_1, n_2, et m des entiers naturels. Soient les vecteurs de coûts $c \in \mathbb{R}^{n_1}$, $f \in \mathbb{R}^{n_2}$, les matrices des contraintes $A \in \mathbb{R}^m \times \mathbb{R}^{n_1}$, $G \in \mathbb{R}^m \times \mathbb{R}^{n_2}$ et le vecteur membre de droite $b \in \mathbb{R}^m$.

Variables de décisions : Soit le vecteur de décision $x \in \mathcal{X}$, où $\mathcal{X} \subset \mathbb{R}^{n_1-n} \cup \mathbb{Z}^n$ est un polyèdre. Soit le vecteur de décision $y \in \mathbb{R}^{n_2}_+$.

Modèle :

$$\begin{aligned} v_{ORG} = \min_{x,y} \quad & c^T x + f^T y \\ \text{s.c.} \quad & Ax + Gy \geq b \\ & x \in \mathcal{X}, \quad \text{(certaines composantes peuvent être entières)} \\ & y \geq 0 \end{aligned} \tag{ORG}$$

Si x est fixé à une valeur de la région réalisable $\mathcal{X}$, alors nous obtenons le *sous-problème primal de Benders*, noté (SPB(x)) :

$$v_{SPB}(x) = \min_y \left\{ f^T y \mid Gy \geq b - Ax,\ y \geq 0 \right\} \tag{SPB(x)}$$

Notons $u \in \mathbb{R}^m$ le vecteur des variables duales correspondant aux contraintes $Gy \geq b - Ax$. Le sous-problème dual de Benders (SDB(x)) est alors :

$$v_{SDB}(x) = \max_u \left\{ (b - Ax)^T u \mid G^T u \leq f,\ u \geq 0 \right\} \tag{SDB(x)}$$

Soit $\mathcal{D}$ la région réalisable de (SDB(x)), supposée non vide :

$$\mathcal{D} = \left\{ u \mid G^T u \leq f,\ u \geq 0 \right\}$$

Notons $\mathcal{R} = \{e^r, r = 1, 2 \ldots, n_{\mathcal{R}}\}$ et $\mathcal{T} = \{u^t, t = 1, 2 \ldots, n_{\mathcal{T}}\}$ l'ensemble des rayons extrêmes et l'ensemble des points extrêmes de $\mathcal{D}$, respectivement, où $n_{\mathcal{R}}$ et $n_{\mathcal{T}}$ représentent les cardinaux de $\mathcal{R}$ et $\mathcal{T}$, respectivement. Notons ν une variable continue, dite de reformulation. Alors, le programme suivant est une reformulation de (ORG), appelée la

reformulation de Benders :

$$
\begin{aligned}
v_{ORG} = \quad & \min_{x,\nu} \quad c^T x + \nu \\
& \text{s.c.} \quad 0 \geq (b - Ax)^T e^r && r = 1, 2 \ldots |\mathcal{R}| && (1.5) \\
& \qquad \nu \geq (b - Ax)^T u^t && t = 1, 2 \ldots |\mathcal{T}| && (1.6) \\
& \qquad x \in \mathcal{X}, \quad \text{certaines composantes peuvent être entières}
\end{aligned}
$$

Les contraintes (1.5) sont appelées coupes de réalisabilité. Les contraintes (1.6) sont appelées coupes d'optimalité. Une telle reformulation n'est pas compacte, car le nombre des points et des rayons extrêmes peut être exponentiel. Aussi, trouver les points extrêmes et les rayons extrêmes d'un polyèdre est lui-même un problème NP-difficile.

L'intérêt de la reformulation de Benders est d'être propice à une résolution par génération dynamique de points extrêmes et de rayons extrêmes. Un tel processus de résolution est appelé *décomposition de Benders*. Dans la suite, nous détaillons les étapes de ce processus.

Décomposition de Benders La décomposition de Benders commence par la résolution du problème maître de Benders relâché initial (PMR^0), qui correspond à la reformulation de Benders mais sans aucune coupe d'optimalité, ni de réalisabilité. Une solution du problème maître relâché $(\hat{x}^0, \hat{\nu}^0)$ est trouvée. Ensuite, itérativement, le sous-problème de Benders correspondant à la solution $\hat{x}^0$, noté ($\text{SPB}(\hat{x}^0)$), est formulé et résolu. Si ($\text{SPB}(\hat{x}^0)$) est non-réalisable, alors une coupe de réalisabilité de la forme (1.5) est ajoutée au problème maître relâché. Sinon, autrement dit si ($\text{SPB}(\hat{x}^0)$) est optimal, et si $\hat{\nu}^0 < v_{SPB}(\hat{x}^0)$, alors une coupe d'optimalité de la forme (1.6) est ajoutée au problème maître relâché.

Suite à l'ajout de coupes au problème maître relâché (PMR^0), un nouvelle itération commence avec le nouveau problème maître de Benders relâché (PMR^1). Ce dernier est résolu une nouvelle fois, pour donner une nouvelle solution $(\hat{x}^1, \hat{\nu}^1)$.

L'algorithme de décomposition de Benders s'arrête quand pour une itération $i\,(\geq 0)$ le sous-problème de Benders ($\text{SPB}(\hat{x}^i)$) est optimal et $\hat{\nu}^i = v_{SPB}(\hat{x}^i)$.

La solution optimale de (ORG) est $(\hat{x}^i, \hat{y}^i)$, où $\hat{y}^i$ est la solution optimale de ($\text{SPB}(\hat{x}^i)$).

1.4.2 Application à la programmation stochastique à deux étapes

La décomposition de Benders a été appliquée avec succès à la résolution de programmes stochastiques à deux étapes [71, 62]. Dans ce contexte, il est souvent appelé l'algorithme du *L-Shaped* [11]. En effet, la structure d'un programme stochastique à deux étapes coïncide avec le partitionnement en un problème maître, correspondant à la première étape, et un sous-problème, correspondant à la deuxième étape. En outre, la reformulation de Benders peut être vue comme la projection du problème d'origine (ORG) sur l'espace des variables du problème maître, x. Ainsi, la valeur de la variable de reformulation ν peut être vue comme l'approximation, par valeurs inférieures, de $\mathcal{Q}(x)$. Les coupes de Benders (1.5) et (1.6) servent à corriger cette approximation. Comme $\mathcal{Q}$ est une fonction convexe (théorème 1.3.1), les coupes de Benders peuvent être vues comme des hyperplans-supports de $\mathcal{Q}$.

Remarquons que dans l'introduction ci-dessus à la décomposition de Benders, nous avons considéré le cas général où le sous-problème de Benders ne présente pas de structure

particulière. Par contre, en programmation stochastique à deux étapes, le sous-problème de Benders est séparable par scénario, donnant lieu à plusieurs sous-problèmes, où chaque sous-problème correspond à un problème de deuxième étape pour un scénario donné. Un des avantages d'une telle séparabilité est de pouvoir résoudre chaque sous-problème indépendamment des autres. La résolution exploitant la séparabilité est souvent plus efficace que la résolution directe d'un sous-problème unique issu de l'agrégation des sous-problèmes par scénarios. Également, la parallélisation peut être envisagée surtout quand le nombre de scénarios de deuxième étape est très grand.

Dans l'algorithme du *L-Shaped* [11], il est préconisé de vérifier d'abord que tous les sous-problèmes sont réalisables. En cas de non réalisabilité, uniquement les coupes de réalisabilité sont ajoutées. Sinon, autrement dit si tous les sous-problèmes sont réalisables, alors les coupes d'optimalité sont ajoutées (si besoin).

Enfin, suite à la résolution séparée de chaque sous-problème, trois approches sont possibles en vue de générer et d'ajouter les coupes de Benders au problème maître relâché :

1. D'abord, il est possible d'agréger les solutions sur tous les sous-problèmes pour retrouver la solution du sous-problème agrégé, et ensuite d'appliquer le schéma de résolution décrit plus haut. Une seule coupe (au plus) peut alors être générée par itération. Nous ferons référence à cette approche sous le nom de la *décomposition de Benders simple coupe*.
2. Une deuxième approche, qualifiée de *désagrégée*, est de générer une coupe de Benders par sous-problème et d'ajouter ces coupes au problème maître relâché. Cette approche sera appelée la *décomposition de Benders multi-coupes*.
3. Une troisième approche "*partiellement agrégée*" est d'agréger les solutions de certains sous-problèmes d'un même *groupe* de scénarios. Ainsi, il sera possible de générer une coupe qui correspond au sous-problème issu de l'agrégation des sous-problèmes d'un même groupe de scénario. Notons que le partitionnement des scénarios en groupes reste à définir par l'utilisateur. Ainsi, une coupe sera potentiellement générée pour chaque groupe de sous-problèmes. Une telle approche est investiguée dans le chapitre 4.

L'avantage de la version multi-coupes est de pouvoir générer des coupes de meilleure qualité, dans le sens d'approcher plus finement la fonction $\mathcal{Q}$. D'autre part, son principal inconvénient est le grand nombre de coupes pouvant s'accumuler au fil des opérations. Par contre, l'approche "simple coupe" permet de réduire le nombre des coupes générées par itération au détriment de la qualité des coupes. Enfin, la version partiellement agrégée peut présenter un bon compromis, surtout si les scénarios peuvent être regroupés d'une façon non triviale.

Dans la suite, nous présentons l'algorithme de décomposition de Benders simple coupe.

Décomposition de Benders simple coupe

Étape 0 $i \leftarrow 0$

Étape 1 Résoudre le problème maître de Benders relâché courant (PMR^i). Retenir $\hat{x}^i$ et $\hat{\nu}^i$.

Étape 2 POUR chaque scénario $s \in \mathcal{S}$ FAIRE :

Étape 2.1 Résoudre le sous-problème primal de Benders, (SPB$(s, \hat{x}^i)$).

Étape 2.2 SI (SPB$(s, \hat{x}^i)$) est non réalisable, ALORS $r \leftarrow s$, retenir le rayon extrême $\hat{e}^{is}$, ALLER À l'Étape 4.

Étape 2.3 Retenir la valeur de la fonction objectif $v_{SPB}(s, \hat{x}^i)$ et le point extrême $\hat{u}^{is}$.

Étape 3 Aggréger les valeurs des fonction objectifs et des points extrêmes : $v_{SPB}(\hat{x}^i) \leftarrow \sum_{s \in \mathcal{S}} v_{SPB}(s, \hat{x}^i)$, $u^{it} \leftarrow \cup_{s \in \mathcal{S}} \hat{u}^{is}$. ALLER À l'Étape 5.

Étape 4 Ajouter la coupe de réalisabilité $0 \geq (b - Ax)^T \hat{e}^{ir}$ à (PMR^i) ; $i \leftarrow i + 1$ et ALLER À l'Étape 1.

Étape 5 SI $\hat{\nu}^i < v_{SPB}(\hat{x}^i)$, ALORS ajouter la coupe d'optimalité $\nu \geq (b - Ax)^T \hat{u}^{it}$ à (PMR^i) ; $i \leftarrow i + 1$ et ALLER À l'Étape 1.

Étape 6 SI $\hat{\nu}^i = v_{SPB}(\hat{x}^i)$, ALORS FIN.

1.4.3 Techniques d'accélération

Même si la décomposition de Benders est une méthode efficace de résolution des programmes linéaires en variables mixtes, sa mise en œuvre informatique peut être difficile et sa performance peut ne pas être à la hauteur des attentes, même quand il s'agit de l'appliquer sur des cas bien adaptés comme des programmes stochastiques à deux étapes. Il n'est donc pas surprenant qu'une implémentation "classique" de la décomposition de Benders puisse être moins performante qu'un Branch-and-Cut d'un solveur de pointe. De nombreux auteurs ont étudié les raisons de la mauvaise performance en pratique de la décomposition de Benders. Leurs travaux ont conduit à proposer de nombreuses techniques algorithmiques d'accélération. Le lecteur intéressé est référé à l'article [62] pour un panorama assez complet de ces techniques d'accélération. Notons que, récemment, à partir de sa version 12.7, le solveur CPLEX d'IBM intègre un algorithme de décomposition de Benders automatique qui profite de plusieurs techniques d'accélération. Dans la suite, nous décrivons quelques techniques classiques d'accélération de la décomposition de Benders.

Un seul arbre de recherche ou Branch-and-Benders-Cut

De nos jours, les solveurs de pointe à base de Branch-and-Bound offrent des fonctionnalités avancées comme les *callbacks*, permettant à l'utilisateur de modifier le comportement de l'algorithme en cours de résolution. Plus particulièrement, un *cut callback* permet d'ajouter des coupes dynamiquement. Une telle fonctionnalité, facile à prendre en main, permet d'implémenter une version de la décomposition de Benders dite *Branch-and-Benders-Cut*, où la décomposition de Benders est intégrée dans un arbre de recherche de type Branch-and-Bound.

L'algorithme de la décomposition de Benders présenté au paragraphe 1.4.2 suppose qu'à chaque itération, le problème maître de Benders est résolu à optimalité, avant de résoudre le sous-problème. En effet, les solutions entières non optimales peuvent aussi servir pour formuler des sous-problèmes et générer des coupes de Benders valides. Selon cette observation, à chaque nœud de l'arbre de recherche du Branch-and-Bound, si une solution entière est trouvée, alors il est possible de générer des coupes de Benders. Ce fonctionnement est facile à mettre en place grâce au *lazy constraint callback* appelé par le solveur CPLEX chaque fois qu'il trouve une solution entière.

Dans une telle implémentation, un seul arbre de recherche est dressé, au bout duquel la solution optimale retournée correspond à la solution optimale du problème d'origine. Les implémentations récentes de la décomposition de Benders utilisent, le plus souvent, un seul arbre de recherche.

Remarquons qu'il n'est pas nécessaire qu'une solution d'un problème maître de Benders soit entière pour donner des coupes de Benders valides. Ainsi, mêmes les solutions fractionnaires peuvent être exploitées, au même titre que les solutions entières.

Schéma de résolution en deux phases

McDaniel and Devine [54] proposent d'appliquer la décomposition de Benders en deux phases. Dans la première phase, la relaxation linéaire du problème maître de Benders relâché est résolue par décomposition de Benders. Les auteurs montrent que les coupes générées pendant cette première phase sont valides pour le problème d'origine, linéaire en variables mixtes. Dans la seconde phase, les contraintes d'intégralité sont réintroduites et la décomposition de Benders est relancée sur le problème maître relâché, mais cette fois-ci enrichi des coupes trouvées en première phase.

Une telle approche peut être aisément incluse dans un Branch-and-Benders-Cut en implémentant une décomposition de Benders "traditionnelle" en première phase uniquement, pendant laquelle les coupes de Benders sont ajoutées explicitement comme des contraintes au problème maître. Cette procédure ressemble à une méthode classique de plans sécants.

Coupes Pareto-optimales

Magnanti and Wong [53] observent que, dans plusieurs applications, le sous-problème dual de Benders est *dégénéré*, c'est-à-dire qu'il possède plusieurs solutions optimales. Sachant que chaque solution duale du sous-problème de Benders peut donner une coupe (d'optimalité) de Benders, potentiellement différente, Magnanti and Wong [53] montrent qu'il existe une relation de dominance (voir la définition en chapitre 4) entre toutes ces coupes. Plus particulièrement, il est possible de trouver un ensemble de coupes de Benders qui ne sont dominées par aucune autre coupe. Ces coupes de Benders non-dominées sont dites des coupes *Pareto-optimales*. En générant des coupes Pareto-optimales, le nombre de coupes de Benders ainsi que le nombre d'itérations nécessaires pour résoudre le problème sont réduits. Par conséquent, le temps de résolution peut être réduit à son tour. Afin d'obtenir une coupe Pareto-optimale à une itération donnée, l'article [53] propose de résoudre un programme linéaire auxiliaire, appelé *le problème de Magnanti-Wong*. En pratique, l'apport des coupes Pareto-optimales est jugé selon le compromis entre le temps passé à résoudre les problèmes auxiliaires et le temps gagné en raccourcissant le chemin de résolution (grâce aux coupes de meilleure qualité). Une difficulté pratique supplémentaire dans l'implémentation des coupes Pareto-optimales est de formuler correctement le problème de Magnanti-Wong qui requiert un point cœur ("core point") ; défini comme un point dans l'intérieur relatif de l'enveloppe convexe du problème maître de Benders.

D'autres techniques telles que la séparation "In-Out" [25, 26] ou une version simplifiée de la décomposition de Benders partielle [20] peuvent être testées rapidement avec un effort d'implémentation relativement réduit, dans le cas d'un programme stochastique à deux étapes.

Chapitre 2

État de l'art

Résumé du chapitre

Dans ce chapitre, nous proposons une revue de littérature du problème d'ordonnancement des atterrissages d'avions dans le cas déterministe et dans le cas sous incertitude.

Le problème d'ordonnancement des atterrissages d'avions a été largement étudié depuis la publication du premier algorithme de séquencement d'avions [22] il y a plus de quatre décennies. Depuis cette date, plusieurs énoncés et formulations mathématiques correspondantes ont été proposés et un large panel de méthodes de résolution a été appliqué. Tout d'abord, commençons par un énoncé de base de notre problème.

Considérons n avions en vol demandant d'atterrir à un même aéroport de destination. Notons $\mathcal{A} = \{1, 2, \dots n\}$ l'ensemble des indices de ces avions. Chaque avion $i \in \mathcal{A}$ possède une fenêtre de temps $[E_i, L_i]$ pendant laquelle l'avion doit *absolument* atterrir et une heure préférentielle d'atterrissage, notée T_i, vérifiant $T_i \in [E_i, L_i]$.

Le problème d'ordonnancement des atterrissages consiste à trouver pour chaque avion $i \in \mathcal{A}$ une heure cible d'atterrissage $y_i \in [E_i, L_i]$ en optimisant un objectif donné et en satisfaisant certaines contraintes opérationnelles. En plus de ces **variables de décision** réelles, des variables de décision entières sont nécessaires pour déterminer la séquence des avions. Pour une paire $(i, j) \in \mathcal{A} \times \mathcal{A}$ et $i \neq j$, notons δ_{ij} la variable binaire de séquencement prenant la valeur 1 si i atterrit avant j et 0 sinon. Soulignons le fait que dès que l'atterrissage de l'avion i est planifié avant celui de l'avion j, δ_{ij} prend la valeur

1 sans que cette précédence soit forcément directe (c'est-à-dire : d'autres avions peuvent s'insérer entre i et j).

Les **principales contraintes** du problème d'ordonnancement des atterrissages sont les contraintes de **fenêtre de temps** et de **séparation de turbulence de sillage**, présentée dans la sous-section 1.1.2 du chapitre 1.

Certaines contraintes ou hypothèses supplémentaires peuvent être ajoutées donnant lieu à des problèmes plus restreints et souvent de complexité réduite. Nous citons à titre d'exemple le changement de position contraint (*Constrained Position Shifting - CPS*) [3, 22], présenté dans la sous-section 2.1.1 du présent chapitre.

Plusieurs fonctions objectif peuvent être considérées en fonction du point de vue de(s) l'acteur(s) impliqué(s) (contrôleurs aériens, compagnies aériennes, aéroport, gouvernement, etc). Dans leur revue de littérature, Bennel et al. [8] énumèrent les objectifs correspondant à chacun de ces acteurs. Néanmoins, la finalité de de l'ordonnancement dans notre étude reste de maximiser le rendement des pistes. La **fonction objectif** prend alors souvent la forme de la minimisation de l'heure du dernier atterrissage : $\min \max_{i \in \mathcal{A}} y_i$. Un tel objectif correspond au point de vue des contrôleurs dans un contexte de fort trafic aérien.

Le problème d'ordonnancement des atterrissages, tel qu'énoncé, s'identifie à la variante du problème de voyageur de commerce avec fenêtres de temps (*Traveling Salesman Problem with Time Windows – TSPTW*). En prenant pour tout $i \in \mathcal{A}$, $E_i = 0$ et $L_i = \infty$, il est aisé de voir que cette variante se réduit à un TSP classique. Nous en déduisons que la complexité de notre problème d'ordonnancement des atterrissages est NP-difficile. Cette analogie ainsi que la complexité de différentes variantes sont détaillées dans la sous-section 2.1.3.

Au-delà de ces éléments de base définissant le problème d'ordonnancement des atterrissages, nous pouvons classifier les différents énoncés dans la littérature selon certains attributs des données d'entrée.

D'abord, nous pouvons regrouper les énoncés selon l'*évolutivité* des données d'entrée au cours du temps :

- **cas statique :** l'ensemble des avions à ordonnancer est inchangé au cours du temps ;
- **cas dynamique :** l'ensemble des avions à ordonnancer évolue progressivement au cours du temps (des avions entrent dans l'horizon opérationnel et d'autres atterrissent).

Les énoncés peuvent également être classés selon la *certitude* sur les données d'entrée :

- **cas déterministe :** toutes les données d'entrée sont connues avec certitude ;
- **cas sous incertitude :** certaines données d'entrée sont incertaines. Deux cas peuvent alors être distingués selon le niveau d'information sur l'incertitude :
  - **stochastique :** les données d'entrée incertaines suivent des lois de probabilités connues ;
  - **robuste :** les données d'entrée incertaines prennent leur valeur dans un ensemble de valeurs possibles (appelé en anglais *uncertainty set*).

À notre connaissance, c'est le cas statique déterministe qui a reçu le plus d'intérêt dans la littérature spécifique à notre problème [8]. Les contributions pour le cas dynamique déterministe sont moins présents dans la littérature et moins structurées [9]. Finalement,

les articles avec prise en compte de l'incertitude sont encore plus rares [38, 65]. Dans la suite, nous commençons par présenter le cas statique déterministe avec ses différentes variantes. Ensuite, nous nous intéressons au cas sous incertitude et plus particulièrement au cas stochastique.

2.1 Ordonnancement déterministe des avions

Plusieurs variantes ont été énoncées et étudiées pour le problème d'ordonnancement d'avions dans le cas statique déterministe. Leurs énoncés peuvent être regroupés selon les éléments suivants :

- **nombre et configuration des pistes :** unique ou multiples. Si pistes multiples, alors indépendantes ou interdépendantes, homogènes ou hétérogènes ;
- **opérations ordonnancées :** atterrissages et/ou décollages ;
- **fonction objectif à minimiser :** heure de la dernière opération planifiée, retard maximal, retard total, coût total du retard, etc ;
- **contraintes opérationnelles :** contraintes de séparations (successives, complètes, diagonales), fenêtres de temps, changement de position contraint, contraintes de précédence ;
- **hypothèses supplémentaires :** fenêtres de temps ordonnées, classes d'avions, matrice des séparations symétrique, etc.

Nombre et configuration des pistes. Le cas de piste unique a été largement traité dans la littérature. Néanmoins, le cadre des pistes multiples a motivé également plusieurs auteurs [5, 51].

Par configuration des pistes d'un aéroport, nous entendons la disposition géométrique relative des pistes entre elles, et les types d'avions et de mouvements autorisés par piste. Dans le cadre des pistes multiples, différentes configurations peuvent amener à différents énoncés :

- pistes **indépendantes** : les opérations sur une piste ne sont pas affectées par les opérations sur les autres pistes ;
- pistes **interdépendantes** : les opérations sur une piste dépendent des opérations sur les autres pistes ;
- pistes **homogènes** : pistes acceptant les mêmes types d'avions et les mêmes types d'opérations (atterrissages et/ou décollages) ;
- pistes **hétérogènes** : par exemple certaines pistes peuvent être exclusivement dédiées aux atterrissages des avions les plus lourds, d'autres aux décollages.

Pour davantage de détails sur les configurations de piste dans la littérature du problème d'ordonnancement d'avions, le lecteur peut consulter l'article [51] où les auteurs fournissent une liste étendue des articles regroupés par nombre et configuration des pistes.

Opérations ordonnancées. Comme observé dans Bennell et al. [8] et présenté dans le chapitre 1, plusieurs auteurs se sont intéressés à l'ordonnancement des atterrissages uniquement (ALP). En effet, dans la pratique, les contrôleurs d'approche sont concentrés sur le flux des arrivées en premier chef tout en essayant d'y intégrer le flux des décollages. Même si dans cet état de l'art nous ne le couvrons pas, le problème de l'ordonnancement des décollages (ATP) a été lui aussi sujet d'étude. Plus généralement, plusieurs auteurs ont traité conjointement les atterrissages et les décollages (ASP) en étendant la matrice des

minima de séparation (aux cas : décollage-décollage, atterrissage-décollage, et décollage-atterrissage) sans changer la formulation mathématique du problème [30].

Fonction objectif. Les formes usuelles sont celles liées à la *performance* telles que l'heure de la dernière opération à la piste (atterrissage ou décollage). Aussi, parmi les objectifs fréquents dans la littérature, nous comptons la minimisation du retard total (vu comme un indicateur de performance de l'aéroport) ou du coût total du retard (qui correspondrait au point de vue des compagnies aériennes).

Contraintes opérationnelles. Les principales contraintes sont les minima de séparation de turbulence de sillage entre les paires d'avions, présentés dans la sous-section 1.1.2 du chapitre 1, et les fenêtres de temps à l'atterrissage.

Plusieurs types de séparation de turbulence de sillage ont été modélisés dans la littérature, parmi lesquelles la séparation : *successive*, *complète* et *diagonale*. Le premier type concerne deux avions opérant successivement sur une même piste. La séparation diagonale doit être assurée entre deux avions opérant sur deux pistes différentes mais interdépendantes. Enfin, il est nécessaire de s'assurer de la séparation complète entre toutes les paires d'avions, quelles que soient leur séquence (se suivant directement ou non), leur opération (deux atterrissages, deux décollages, ou un atterrissage et un décollage) et leur piste.

Quant aux fenêtres de temps, elles désignent les intervalles de temps pendant lesquelles les avions doivent opérer (atterrir ou décoller). Le plus souvent, elles sont traités comme des contraintes *dures* (dont la violation n'est pas permise) car elles traduisent des limitations physiques de l'avion (vitesse maximale ou minimale, quantité de carburant restante, etc).

Des contraintes opérationnelles supplémentaires peuvent s'ajouter :

— **Changement de position contraint (*Constraint Position Shifting - CPS*) :** cette contrainte a été introduite la première fois par Dear [22]. Elle consiste à limiter l'amplitude de changement de position des avions par rapport à leur ordre d'arrivée. Ainsi, un avion ne peut être dévié de sa position initiale dans la séquence "premier arrivé, premier servi" (*First-Come First-Served - FCFS*) que d'un nombre limité de positions (en anglais, *Maximum Position Shifting - MPS*). Cette limitation au changement de séquence émane de la réalité opérationnelle qu'un avion ne peut être expédié ou retardé que d'une durée limitée dans l'horizon opérationnel considéré. Cette contrainte a pour vertu de garantir une certaine équité entre les avions et de générer des séquences faciles à composer opérationnellement par les contrôleurs.

— **Contraintes de précédence :** certains avions prioritaires doivent atterrir avant d'autres. Certaines compagnies aériennes peuvent exprimer des préférences par rapport à l'ordre d'atterrissage de leurs vols en fonction de leurs schémas de connections qu'ils offrent [16]. Ces mêmes contraintes peuvent être pertinentes pour des avions arrivant d'une même route aérienne, exprimant ainsi une certaine équité entre ces avions.

Hypothèses supplémentaires. Parmi les hypothèses supplémentaires phares de notre problème, nous citons le regroupement des avions par classe de turbulence. Comme les minima de séparation les plus utilisés en pratique ne sont pas liés à la paire d'avions

individuels considérés mais à leur classe de turbulence, le problème peut alors se simplifier. Cette idée a motivé plusieurs travaux dont [2, 15, 51, 61]. Une autre hypothèse est celle des fenêtres de temps ordonnées, dont nous donnons la définition dans la suite.

Définition 2.1.1 (Fenêtres de temps ordonnées)**.** *Deux fenêtres de temps,* $[E_i, L_i]$ *et* $[E_j, L_j]$*, sont dites* ***ordonnées*** *si :* $E_i \leq E_j \Leftrightarrow L_i \leq L_j$*.*

Dans la suite, nous présentons les énoncés les plus connus de notre problème en commençant par l'énoncé ayant traité le cas de piste unique ensuite celui de pistes multiples.

2.1.1 Ordonnancement des avions sur une piste unique

Nous présentons brièvement l'ordonnancement des atterrissages sur une piste unique avant de passer au cas où les décollages sont considérés simultanément avec les atterrissages lors de l'optimisation.

Ordonnancement des atterrissages

Cet énoncé correspond à ce que nous avons présenté en début de ce chapitre. C'est l'énoncé le plus simple et autour duquel les articles sont les plus présents dans la littérature [8]. Pour rappel, le problème est d'affecter une heure d'atterrissage à chaque avion en respectant sa fenêtre de temps et les contraintes de séparation.

Notons que dans le cas de l'ordonnancement des atterrissages sur une seule piste, les séparations de turbulence de sillage (voir table 3.1) vérifient l'*inégalité triangulaire.*

Définition 2.1.2 (inégalité triangulaire)**.** *Notons* S_{ij} *la séparation de turbulence de sillage minimale entre les avions* i *et* j*. Si pour tout triplet d'avions* $(i, j, k) \in \mathcal{A} \times \mathcal{A} \times \mathcal{A}$ *tels que* $i \neq j \neq k$ *et* $i \neq k$*, on a* $S_{ij} + S_{jk} \geq S_{ik}$ *alors, on dit que la séparation de turbulence de sillage minimale satisfait l'inégalité triangulaire.*

Grâce à cette propriété, dès que la séparation entre toutes les paires d'avions *successifs* est vérifiée, la séparation est alors vérifiée entre tous les avions. Autrement dit, dans le cas de l'ordonnancement des atterrissages sur une piste unique, la séparation *successive* implique la séparation *complète.*

Très souvent, l'objectif est de minimiser l'heure du dernier atterrissage ou bien le coût total des déviations par rapport aux heures prévues d'atterrissage de tous les avions. Un tel énoncé a été formulé et résolu comme un programme linéaire mixte dans Beasley et al. [5]. Cet article étant la référence la plus citée dans la littérature, nous rapportons, dans la sous-section 2.1.6, le modèle d'ordonnancement des atterrissages sur une piste unique qui y est proposé.

Balakrishnan et Chandran ont travaillé sur un problème d'ordonnancement des atterrissages sur une piste unique avec changement de position contraint (CPS) et des contraintes de précédence. Les auteurs ont proposé une approche par programmation dynamique [3].

Ordonnancement simultané des atterrissages et des décollages

Dans plusieurs aéroports, les atterrissages et les décollages sont opérés sur une même piste. Ainsi, il est souvent question d'ordonnancer simultanément les atterrissages et les

décollages sur une piste unique. Dans un tel cas de figure, les minima de séparation ne vérifient plus l'inégalité triangulaire. Ainsi, la séparation successive n'est plus suffisante. Il faut s'assurer explicitement de la séparation entre toutes les paires d'avions et non seulement entre les paires d'avions successifs. Vérifier la séparation *complète* est donc nécessaire.

Le problème est d'affecter une heure d'atterrissage ou de décollage à chaque avion en respectant sa fenêtre de temps et les contraintes de séparation avec tous les avions précédents dans la séquence.

Comme précédemment, l'objectif est de minimiser l'heure de la dernière opération (atterrissage ou décollage) ou bien le coût total des déviations par rapport aux heures prévues d'opération de tous les avions.

Furini et al. [30] ont énoncé un tel problème. Néanmoins, ils considèrent uniquement des contraintes de séparation successive. Par contre, Balakrishnan and Chandran [3] étendent leur travail à l'ordonnancement simultané des atterrissages et des décollages où les minima de séparation ne satisfont pas nécessairement l'inégalité triangulaire.

2.1.2 Ordonnancement sur des pistes multiples

Selon la revue de littérature de Bennell et al. [8], le problème sur une seule piste a été plus largement étudié dans la littérature que le problème multi-piste, souvent plus complexe. Tout d'abord, le problème multi-piste requiert des décisions supplémentaires d'allocation de pistes : allouer les pistes disponibles aux opérations d'atterrissages / décollages. La configuration de pistes la plus simple et la plus traitée est celle des pistes parallèles suffisamment espacées pour être qualifiées d'indépendantes. Briskorn et Stolletz [15] proposent des algorithmes de programmation dynamique de complexité polynomiale pour l'ordonnancement des atterrissages avec classes d'avions sur des pistes parallèles. Ghoniem et al. [33] traitent également une telle configuration de pistes considérant simultanément les atterrissages et les décollages. Ces derniers auteurs ont opté pour la génération de colonnes comme méthode de résolution.

Bien souvent, en réalité, les configurations des pistes sont plus complexes : les opérations sur des pistes différentes sont interdépendantes. Par conséquent, les contraintes de séparation *diagonales* sont requises. Beasley et al. [5] étendent leur modèle sur piste unique au cas avec plusieurs pistes homogènes et interdépendantes. Ils choisissent de modéliser les interdépendances entre pistes par des séparations diagonales spécifiques à la paire d'avions considérée mais identiques pour toutes les paires de pistes, autrement dit la même séparation diagonale est requise pour une même paire d'avions dès que ces derniers utilisent deux pistes différentes. À notre connaissance, les seuls auteurs ayant traité la configuration la plus générale des pistes (hétérogènes et interdépendantes) sont Lieder et Stolletz [51], où l'interdépendance des pistes est spécifique aussi bien à la paire d'avions qu'à la paire de pistes utilisées. Notons cependant qu'ils font l'hypothèse de classes d'avions et de fenêtres de temps ordonnées.

Ayant résumé les énoncés variés du cas statique déterministe, nous passons à présent à l'étude des analogies de notre problème avec des problèmes classiques de la littérature. Nous en déduisons également les complexités des variantes énoncées.

2.1.3 Analogies et complexités

En toute généralité, un problème réel peut rarement être formulé comme une version "pure" d'un problème classique de recherche opérationnelle [6]. Il est cependant clair que le problème d'ordonnancement de tâches et le problème de tournées de véhicules présentent des grandes similarités avec notre problème. Tout d'abord, rappelons la notation "*des trois champs*" spécifique au domaine de l'ordonnancement des tâches et qui nous sera utile dans la suite. C'est une notation introduite par Graham [34] permettant de décrire les problèmes d'ordonnancement des tâches. Elle est constituée de trois champs $\alpha|\beta|\gamma$, où α décrit les machines, β les tâches et γ la fonction objectif.

Dans la suite, nous présenterons ces deux analogies permettant de déduire la complexité de notre problème. Nous évoquerons plus particulièrement des variantes polynomiales traitées dans la littérature. Finalement, nous conclurons sur les limites de ces analogies.

Première analogie : problème d'ordonnancement de tâches

Classiquement, le lien a été établi dans [5, 15] entre le problème d'ordonnancement des atterrissages et le problème d'ordonnancement de tâches avec temps de réglage dépendants de la séquence (*sequence-dependent setup times*). Dans [15], la correspondance se dessine comme suit :

- les pistes sont les machines
- les avions (opérations d'atterrissage) sont les tâches
- le minimum de séparation entre deux atterrissages successifs, i suivi de j sur une même piste, peut être interprété comme la somme de la durée de la tâche i (occupation de la piste) et le temps de réglage (temps d'inactivité) entre i et j.

Notons que selon cette analogie, les durées des tâches sont considérées constantes et égales (*equal processing times*) alors que les temps de réglage sont dépendants de la séquence (*sequence-dependent setup time*).

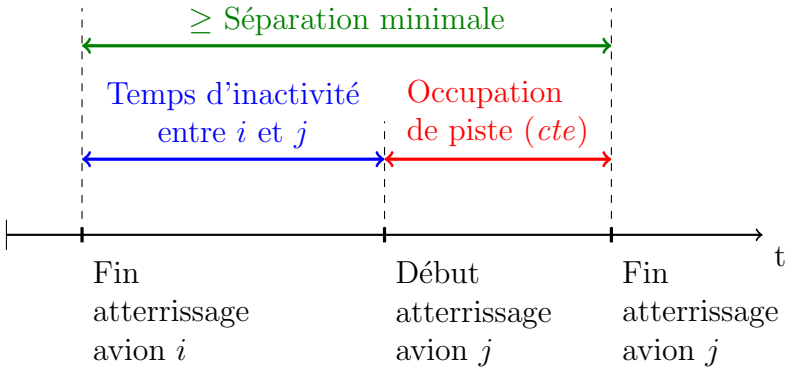


FIGURE 2.1 – Correspondance entre évènements sur la piste et contrainte de séparation dans [15]

Deux autres correspondances sont suggérées dans Beasley et al. [5] : soit considérer des durées de tâches dépendantes de la séquence et des temps de réglage nuls, soit l'inverse. Dans sa thèse [65], consacrée au cas stochastique, Sölveling dessine la deuxième option

de correspondance : des durées de tâches nulles et des temps de réglage dépendants de la séquence.

L'objectif peut être de minimiser le temps de fin de la dernière tâche (du dernier atterrissage) dit le *makespan* (C_{max}) dans le vocabulaire de l'ordonnancement. Dans le cas d'une piste unique, le nombre de machines est réduit à une seule et le problème d'ordonnancement s'écrit dans la notation des trois champs : $1|s_{jk}|C_{max}$, où s_{jk} modélise les temps de réglage dépendants de la séquence. Ce problème d'ordonnancement correspond au problème classique de voyageur de commerce (TSP) connu pour être NP-difficile au sens fort [60].

Le problème peut être complexifié en ajoutant des fenêtres de temps propres aux avions tout en gardant l'analogie avec les problèmes d'ordonnancement :

— les heures d'atterrissage au plus tôt des avions sont les heures d'arrivée / de disponibilité (*release dates*, notées r_j dans la notation de Graham) des tâches
— les heures d'atterrissage au plus tard correspondent aux heures d'échéance (*due dates* ou *deadlines*, notées d_j dans la notation de Graham)

Notons que ce problème d'ordonnancement, $1|r_j, d_j, s_{jk}|C_{max}$, correspond au problème de voyageur de commerce avec fenêtres de temps (TSPTW) également NP-difficile au sens fort.

Aussi, le cas de pistes homogènes indépendantes peut se décliner en un problème à machines identiques parallèles : $P_m|r_j, d_j, s_{jk}|C_{max}$. Par contre, si les pistes sont hétérogènes, c'est-à-dire si chaque piste n'est éligible qu'à certains types d'avions, alors le problème d'ordonnancement peut se noter : $P_m|r_j, d_j, s_{jk}, M_j|C_{max}$ où M_j désigne des contraintes de compatibilité entre les tâches et les machines.

Finalement, d'autres fonction objectifs sont intéressantes dans notre contexte, telles que la somme des retards ou le coût total du retard. Dans le premier cas, la fonction objectif se note $\sum T_j$ où T_j est le retard [1] de la tâche j. Ainsi, le problème d'ordonnancement sur machine unique et avec temps de réglages dépendants de la séquence se note : $1|s_{jk}|\sum T_j$. Ce dernier correspond à une variante bien connue du problème de voyageur de commerce (TSP) appelé *Traveling Repairman Problem - TRP*. Cette dernière analogie a été mentionnée dans [28]. La variante avec fenêtres de temps est appelée TRPTW : $1|r_j, d_j, s_{jk}|\sum T_j$. La variante avec plusieurs réparateurs est appelée k-TRP : $P_m|s_{jk}|\sum T_j$.

Dans le cas de fonctions de coûts linéaires, l'objectif correspondrait à la minimisation de la somme pondérée des retards : $\sum w_j T_j$ où w_j est le coût unitaire du retard de la tâche j. Le problème d'ordonnancement correspondant est $P_m|r_j, d_j, s_{jk}|\sum w_j T_j$.

La fonction objectif peut aussi prendre en compte le coût total de l'avance. Si les coûts d'avance sont linéaires, la fonction objectif peut se noter par extension : $\sum w'_j E_j + \sum w''_j T_j$ où E_j est l'avance [2] de la tâche j et w'_j et w''_j sont respectivement les coûts unitaires d'avance et du retard de la tâche j. [3]

1. Notons ici que le retard est calculé par rapport à l'heure préférentielle, ou cible (*target time*) d'atterrissage et non par rapport à l'heure d'échéance (*due date*) comme il est le cas dans l'ordonnancement classique.

2. De même que pour le retard, l'avance est calculée par rapport à l'heure préférentielle, ou cible (*target time*) d'atterrissage et non par rapport à l'heure d'échéance (*due date*) comme il est le cas dans l'ordonnancement classique.

3. Un tel objectif, n'étant pas forcément croissant avec les heures de fin (C_j) des tâches, est dit

Cette dernière fonction objectif correspond exactement à la formulation la plus citée dans la littérature de notre problème de séquencement des atterrissages, celle de Beasley et al. [5].

Dans tous ces cas, notre problème est une généralisation du problème d'ordonnancement $1||\sum w_j T_j$ connu pour être NP-difficile au sens fort [60].

Seconde analogie : problème de tournées de véhicules (VRP)

La seconde analogie se fait avec le problème de tournées de véhicules (*Vehicle Routing Problems - VRP*). Rappelons qu'historiquement le problème de séquencement des atterrissages sur une piste unique ayant pour objectif de minimiser le temps d'atterrissage du dernier avion a été tout d'abord reconnu par Psaraftis [61] comme une instance du problème de voyageur de commerce. Dans ce contexte, les avions correspondent aux clients et la piste au voyageur de commerce. La distance entre deux clients i et j peut correspondre simplement au temps de séparation minimale entre i et j, S_{ij}. Par extension, le cas de pistes multiples et indépendantes est reconnu comme un problème de tournées de véhicules sans contraintes de capacité (*uncapacitated* VRP) où chaque piste correspond à un véhicule. À cela s'ajoutent les contraintes des fenêtres de temps : les temps d'arrivée des avions et leurs temps d'atterrissage au plus tard. Ces dernières contraintes sont classiques dans le contexte du problème de tournées de véhicules avec fenêtres de temps (VRPTW), où chaque client souhaite être livré pendant un intervalle de temps donné et potentiellement à une heure préférentielle donnée. Classiquement, l'objectif à minimiser est la somme des retards subis par les clients ou le coût total du retard.

Le cas de pistes hétérogènes peut se décliner en considérant une flotte hétérogène de véhicules et des clients demandant des services délivrés exclusivement par certains types de véhicules. Par contre, le cas de pistes interdépendantes n'a pas de correspondance *directe* dans le problème de tournées de véhicules.

2.1.4 Limites des analogies

Nous remarquons que le cadre plus large des problèmes d'ordonnancement nous permet de formuler différentes variantes de notre problème sans grande difficulté. Cependant, certains aspects ne sont pas évidents à retrouver dans les énoncés classiques.

Fonction objectifs générales. La littérature classique de l'ordonnancement ne permet pas de formuler directement l'objectif de minimiser le coût de la déviation par rapport à l'heure prévue où la forme de la fonction objectif est quelconque. Souvent, les auteurs se rapportent à des fonctions linéaires ou convexes linéaires par morceaux, permettant de se ramener à une fonction objectif linéaire.

Pistes interdépendantes. Parmi les aspects résistants aux analogies précédentes, on trouve la modélisation des pistes avec des configurations générales, plus précisément en cas d'interdépendance des pistes. Pour rappel, sur le plan opérationnel, ceci se traduit par des séparations supplémentaires, dites *diagonales*, entre avions opérant sur des pistes interdépendantes. L'analogie avec les problèmes classiques précédents n'est alors pas évidente.Néanmoins, certains travaux ont émergé sur des variantes du TSP avec véhicules

non-régulier dans la littérature de l'ordonnancement.

coopératifs [32]. Que faire, par exemple, pour les problèmes mettant en jeu des véhicules en conflit ou en compétition ? Est-ce qu'une hybridation avec la planification de systèmes multi-agents, par exemple, serait envisageable ? L'adaptation de telles méthodes à notre problématique reste une question ouverte.

2.1.5 Variantes polynomiales

En dépit de la complexité NP-difficile de notre problème, certaines variantes peuvent s'avérer polynomiales. Tel est le cas sous la contrainte de "Changement de Position Contraint" (CPS) ou en regroupant les avions en classes selon leurs catégories de turbulence.

Changement de Position Contraint

Rappelons que, sous cette contrainte, un avion ne peut être dévié de sa position initiale dans la séquence "premier arrivé, premier servi" que d'un nombre limité de positions (MPS) garantissant ainsi une certaine équité entre les avions. Sous une telle hypothèse, Balakrishnan et Chandran[3] présentent des algorithmes de programmation dynamique pour le cas de piste unique. Ces algorithmes sont polynomiaux en le nombre des avions et exponentiels en le nombre maximal de changement de positions, sachant qu'en pratique, le nombre maximal de changement de position est limité à MPS = 3 [21]. Remarquons qu'en imposant une telle contrainte, l'ensemble des séquences candidates est réduit considérablement, rendant le problème traitable.

Classes d'avions et fenêtres de temps ordonnées

Également, la complexité de la variante présentée dans [15, 61], avec regroupement par classes d'avions et fenêtres de temps ordonnées, a été démontrée polynomiale. Briskorn et Stolletz [15] en établissent l'analogie avec le problème de tournées de véhicules en associant le regroupement des avions dans des classes au regroupement des clients dans des villes, où :

- la distance entre deux clients dans une même ville est identique
- la distance entre deux clients dans deux villes différentes dépend uniquement des deux villes

Les auteurs démontrent que, sous ces hypothèses, il est optimal de planifier les avions appartenant à une même classe selon l'ordre "premier arrivé, premier servi" (*FCFS*). Notons que, dans cette variante, les fonctions coûts du retard sont définies par classe d'avions et non pas par avion individuel.

Pour cette variante, Briskorn et Stolletz [15] ont présenté des algorithmes polynomiaux de programmation dynamique pour le problème des atterrissages sur des pistes homogènes et indépendantes, mais ces algorithmes étaient d'une complexité intraitable en pratique : $O(n^{17})$ pour deux pistes et deux classes d'avions. Plus tard, Lieder et al. [52] ainsi que Lieder et Stolletz [51] ont réussi à implémenter efficacement ces algorithmes et à les étendre à l'ordonnancement simultané des atterrissages et des décollages sur des pistes en configuration générale (pistes interdépendantes et hétérogènes), notamment grâce à des règles de dominance.

Malgré ces résultats aboutis, l'hypothèse de classes d'avions est considérée très simplificatrice, puisqu'elle interdit la définition de grandeurs spécifiques aux avions. Même si une telle restriction est justifiée pour les minima de séparation, elle est très restrictive quant aux coûts de retard. En effet, parmi les facteurs contribuant au coût du retard d'un vol, on compte la consommation de carburant de l'aéronef, le nombre de passagers et les éventuelles connections liées à ce vol. Or, tous ces facteurs sont spécifiques à chaque vol et diffèrent entre les vols avec des avions d'une même classe de turbulence de sillage. Également, Guepet [35] attire l'attention sur des spécificités des décollages non captées par les classes d'avions, telles que les créneaux de décollage (laps de temps alloués aux avions pour décoller) et les segments de montée pour une réelle séparation des décollages.

2.1.6 Modèle de Beasley et al. [5]

La formulation de programmation linéaire en variables mixtes proposée par Beasley et al. [5] est la formulation la plus citée dans la littérature du problème d'ordonnancement des atterrissages. Ici, nous ne présentons que la formulation du problème à une seule piste sachant que les auteurs ont aussi proposé un modèle multi-piste.

Données d'entrée

n **:** nombre d'avions demandant d'atterrir

$\mathcal{A} = \{1, 2, \dots n\}$ **:** ensemble des indices représentant les avions demandant d'atterrir

Pour chaque avion $i \in \mathcal{A}$ **:**

E_i **:** heure d'atterrissage au plus tôt

L_i **:** heure d'atterrissage au plus tard

T_i **:** heure préférentielle d'atterrissage

$[E_i, L_i]$ **:** fenêtre de temps pour l'atterrissage, avec $E_i \leq T_i \leq L_i$

g_i **:** coût (par unité de temps) d'atterrir plus tôt que l'heure préférentielle T_i, avec $g_i \geq 0$

h_i **:** coût (par unité de temps) d'atterrir plus tard que l'heure préférentielle T_i, avec $h_i \geq 0$

Pour chaque paire d'avions $(i, j) \in \mathcal{A} \times \mathcal{A},\ i \neq j$ **:**

S_{ij} **:** minimum de séparation temporelle entre l'atterrissage de l'avion i et celui de l'avion j, où l'avion i atterrit avant l'avion j, $S_{ij} \geq 0$

Variables de décisions

Pour chaque avion $i \in \mathcal{A}$ **:**

y_i **:** heure cible d'atterrissage de l'avion i

α_i **:** temps d'avance de l'heure cible d'atterrissage y_i par rapport à l'heure préférentielle T_i de l'avion i, avec $\alpha_i \geq 0$

β_i **:** durée du retard de l'heure cible d'atterrissage y_i par rapport à l'heure préférentielle T_i de l'avion i, avec $\beta_i \geq 0$

Pour chaque paire d'avions $(i, j) \in \mathcal{A} \times \mathcal{A},\ i \neq j$ **:**

$$\delta_{ij} = \begin{cases} 1 & \text{si l'avion } i \text{ atterrit avant l'avion } j \\ 0 & \text{sinon} \end{cases}$$

Commentaires :

— Pour un avion $i \in \mathcal{A}$, les variables $\delta_{ij},\ j \in \mathcal{A}\backslash\{i\}$, permettent de déterminer la position relative de i par rapport à tous les avions dans la séquence.

— Ainsi, la position de l'avion i est donnée par :

$$n - \sum_{\substack{j \in \mathcal{A} \\ j \neq i}} \delta_{ij}$$

Fonction objectif

$$\sum_{i \in \mathcal{A}} (g_i \alpha_i + h_i \beta_i) \tag{2.1}$$

La fonction objectif est la somme des coûts de déviation par rapport à l'heure préférentielle par avion, que nous souhaitons minimiser. Pour chaque avion, ce coût est supposé linéaire par morceaux avec un seul point de brisure (voir figure 2.2).

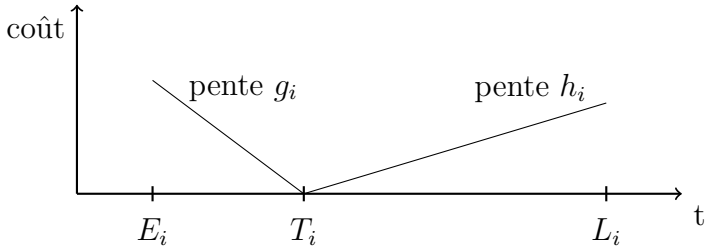


FIGURE 2.2 – Fonction coût de déviation pour un avion $i \in \mathcal{A}$ [5].

Contraintes

$$E_i \leq y_i \leq L_i \qquad i \in \mathcal{A} \tag{2.2}$$

$$\delta_{ij} + \delta_{ji} = 1 \qquad i, j \in \mathcal{A} \quad j > i \tag{2.3}$$

— Les contraintes (2.2) forcent le respect des fenêtres de temps d'atterrissage pour chaque avion.

— Les contraintes (2.3) assurent que les positions relatives de deux avions sont bien définies.

En tenant compte des fenêtres de temps, les auteurs décomposent l'ensemble des paires d'avions en trois sous-ensembles :

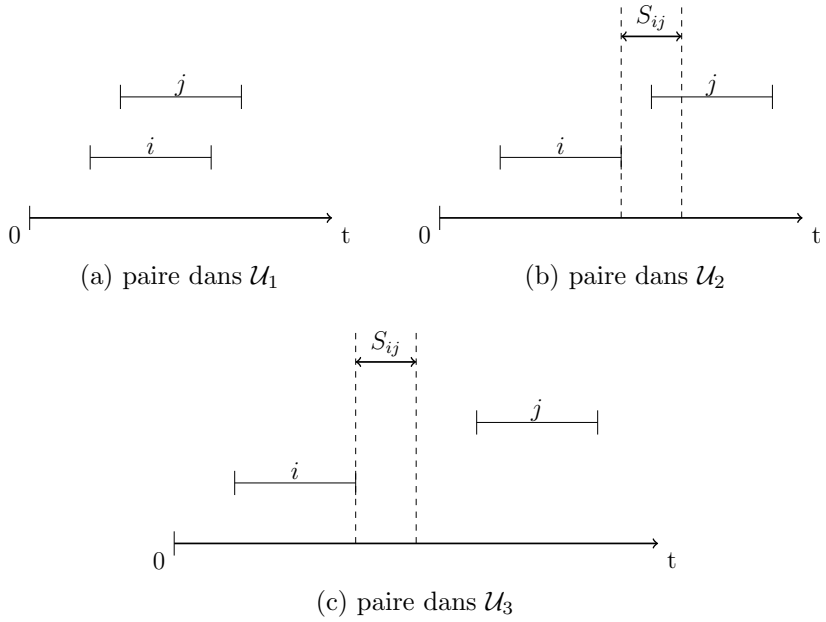
$\mathcal{U}_1$ **:** ensemble des paires d'avions (i, j) dont **l'ordre relatif des atterrissages n'est pas connu à l'avance**.

$\mathcal{U}_2$ **:** ensemble des paires d'avions (i, j) où l'avion i doit **certainement** atterrir **avant** j mais pour lesquels la **séparation** au seuil de piste n'est **pas automatiquement satisfaite**.

$\mathcal{U}_3$ **:** ensemble des paires d'avions (i, j) où l'avion i doit **certainement** atterrir **avant** j et pour lesquels la **séparation** est **automatiquement satisfaite**.

Ces ensembles sont illustrés à l'aide d'un exemple en figure 2.3, et sont définis plus formellement comme suit :

$$\mathcal{U}_1 = \{(i,j) \in \mathcal{A} \times \mathcal{A}, i \neq j \mid E_j \leq E_i \leq L_j \text{ ou } E_j \leq L_i \leq L_j \text{ ou } E_i \leq E_j \leq L_i \text{ ou } E_i \leq L_j \leq L_i\}$$
$$\mathcal{U}_2 = \{(i,j) \in \mathcal{A} \times \mathcal{A}, i \neq j \mid L_i < E_j \quad \text{et} \quad L_i + S_{ij} > E_j\}$$
$$\mathcal{U}_3 = \{(i,j) \in \mathcal{A} \times \mathcal{A}, i \neq j \mid L_i < E_j \quad \text{et} \quad L_i + S_{ij} \leq E_j\}$$



(a) paire dans $\mathcal{U}_1$

(b) paire dans $\mathcal{U}_2$

(c) paire dans $\mathcal{U}_3$

FIGURE 2.3 – Exemples de fenêtres de temps correspondant à des paires d'avions (i,j) dans les ensembles : $\mathcal{U}_1$, $\mathcal{U}_2$ et $\mathcal{U}_3$

Vu la définition de ces ensembles, les contraintes suivantes doivent nécessairement être satisfaites par toute solution de notre problème d'ordonnancement :

$$\delta_{ij} = 1 \qquad (i,j) \in \mathcal{U}_3 \cup \mathcal{U}_2 \qquad (2.4)$$
$$y_j \geq y_i + S_{ij} \qquad (i,j) \in \mathcal{U}_2 \qquad (2.5)$$
$$y_j \geq y_i + S_{ij} - M_{ij}\delta_{ji} \qquad (i,j) \in \mathcal{U}_1 \qquad (2.6)$$

où M_{ij} est grande constante définie ci-après.

- Les contraintes (2.4) imposent que i atterrisse avant j pour toute paire dont l'ordre d'atterrissage est évident, c'est-à-dire pour tout couple $(i,j) \in \mathcal{U}_3 \cup \mathcal{U}_2$.
- Les contraintes (2.5) assurent le minimum de séparation entre les avions i et j, dont l'ordre est déjà imposé par (2.4) mais dont le minimum de séparation n'est pas automatiquement satisfait, c'est-à-dire pour tout couple $(i,j) \in \mathcal{U}_2$.
- Les contraintes (2.6) assurent le minimum de séparation entre les avions i et j dont l'ordre d'atterrissage n'est pas évident, c'est-à-dire pour tout couple $(i,j) \in \mathcal{U}_1$ où M_{ij} est une constante choisie suffisamment grande de façon à ce que la contrainte soit toujours satisfaite lorsque $\delta_{ij} = 1$ (constante dite *big-M* en recherche opérationnelle).

— Notons qu'il suffit de choisir $M_{ij} = (L_i + S_{ij} - E_j)$ et via (2.3), les contraintes (2.6) deviennent :

$$y_j \geq y_i + S_{ij}\delta_{ij} - (L_i - E_j)\delta_{ji} \quad (i,j) \in \mathcal{U}_1 \tag{2.7}$$

Les dernières contraintes établissent les liens entre les variables de décisions y_i, α_i et β_i pour chaque avion $i \in \mathcal{A}$:

$$\alpha_i \geq T_i - y_i \tag{2.8}$$
$$0 \leq \alpha_i \leq T_i - E_i \tag{2.9}$$
$$\beta_i \geq y_i - T_i \tag{2.10}$$
$$0 \leq \beta_i \leq L_i - T_i \tag{2.11}$$
$$y_i = T_i - \alpha_i + \beta_i \tag{2.12}$$

Pour résumer :

Modèle complet de Beasley et al. [5]

$$\min_{y,\alpha,\beta,\delta} \quad \sum_{i\in\mathcal{A}} (g_i\alpha_i + h_i\beta_i)$$

$$\begin{aligned}
\text{s.c.} \quad & \delta_{ij} + \delta_{ji} = 1 , && (i,j) \in \mathcal{A}\times\mathcal{A};\quad j > i \\
& \delta_{ij} = 1 , && (i,j) \in \mathcal{U}_3 \cup \mathcal{U}_2 \\
& y_j \geq y_i + S_{ij} , && (i,j) \in \mathcal{U}_2 \\
& y_j \geq y_i + S_{ij} - M_{ij}\delta_{ji} , && (i,j) \in \mathcal{U}_1 \\
& \alpha_i \geq T_i - y_i , && i \in \mathcal{A} \\
& \beta_i \geq y_i - T_i , && i \in \mathcal{A} \\
& y_i = T_i - \alpha_i + \beta_i , && i \in \mathcal{A} \\
& 0 \leq \alpha_i \leq T_i - E_i , && i \in \mathcal{A} \\
& 0 \leq \beta_i \leq L_i - T_i , && i \in \mathcal{A} \\
& E_i \leq y_i \leq L_i , && i \in \mathcal{A} \\
& \delta_{ij} \in \{0,1\} , && (i,j) \in \mathcal{A}\times\mathcal{A}
\end{aligned}$$

2.2 Ordonnancement des atterrissages d'avions sous incertitude

Par opposition aux nombreux articles traitant de la version déterministe, très peu d'auteurs se sont intéressés au problème d'ordonnancement des atterrissages d'avions en présence d'incertitude. Pourtant, plusieurs études sur des données réelles ont révélé que l'heure réelle (effective) d'atterrissage ne correspond pas à l'heure prévue, donnée par les modèles de prédiction, [67, 69, 70] ni à l'heure cible d'atterrissage, donnée par le contrôle aérien [70]. Cette déviation a souvent été identifiée à une variable aléatoire de distribution connue.

À notre connaissance, la première contribution à l'ordonnancement des atterrissages en présence d'incertitude date de 2005 avec l'article de Meyn et Erzberger [55]. Les auteurs se sont intéressés à l'impact d'améliorer la précision de l'ordonnancement sur la capacité aéroportuaire. À cette fin, les auteurs ont considéré plusieurs systèmes opérationnels, actuels et futuristes, d'ordonnancement caractérisés par différents niveaux d'incertitude sur les données d'entrée. Cette incertitude est modélisée par des bruits gaussiens sur les heures de passage au point de début d'approche (*Initial Approach Fix - IAF*) et sur les heures d'atterrissage. Pour faire face à cette incertitude, Meyn et Erzberger ont calculé des marges de séparation supplémentaire au seuil de piste (*buffers*). L'ordonnancement a été calculé avec les nouvelles séparations augmentées selon la règle "premier arrivé, premier servi" (FCFS). Les auteurs ont conclu qu'améliorer la précision peut augmenter la capacité aéroportuaire jusqu'à 42% de sa capacité.

Dans les années suivantes, 2007 et 2008, des travaux s'inspirant des données numériques dans [55] sur l'incertitude des heures d'atterrissage ont été publiés [17, 49]. Le problème d'ordonnancement robuste en piste est alors résolu par programmation dynamique sous contraintes CPS. Notons que le qualificatif "robuste" faisait référence à la "résistance" aux perturbations sur les heures d'atterrissage modélisées comme des variables aléatoires. De ce fait, en suivant la convention utilisée en début de ce chapitre, le problème serait renommé "ordonnancement *stochastique* en piste".

Chandran et Balakrishnan [17] ont cherché à étudier le compromis entre la robustesse et l'efficacité, en comparant l'ordonnancement "premier arrivé, premier servi" (FCFS) et celui par programmation dynamique sous contraintes CPS. Pour cela, ils ont introduit la notion de *fiabilité* d'un ordonnancement comme une mesure de la robustesse, alors que l'heure du dernier atterrissage a été prise comme mesure de l'efficacité. La fiabilité est définie comme la probabilité qu'aucune séparation inter-arrivées ne soit violée, c'est-à-dire la probabilité jointe que toutes les séparations inter-arrivées soient respectées. De plus, les auteurs ont considéré les séparations inter-arrivées comme des variables aléatoires dépendantes, où la séparation entre une paire d'arrivées successives dépend uniquement de la paire précédente. Dans leur exemple numérique, deux niveaux d'incertitude ont été définis selon que l'avion est équipé d'un système de gestion de vol (*Flight Management System - FMS*) précis ou non. Ces incertitudes sont indépendantes et suivent des lois triangulaires symétriques (± 150 et ± 300 secondes). Les résultats ont montré que la fiabilité et l'efficacité sont des objectifs contradictoires. Les auteurs ont conclu que l'ordonnancement par programmation dynamique sous CPS était meilleur aussi bien en robustesse qu'en efficacité, comparé au FCFS.

Le mémoire de Master de Lee [49] sous la supervision de Balakrishnan, défendu en 2008, reprend les idées principales de [17]. En plus des méthodes d'ordonnancement testées (FCFS, programmation dynamique sous CPS), la possibilité d'avancer les avions de plusieurs minutes par rapport à leurs heures prévues d'atterrissage, appelée *Time Advance*, a été étudiée. Une nouvelle mesure de robustesse (différente de la *fiabilité* présentée dans [17]) a été proposée : la *faiblesse* d'un ordonnancement. Elle est définie comme le maximum des probabilités que la séparation inter-arrivée soit violée pour une paire d'avions. Comparé à la maximisation de la *fiabilité*, minimiser la *faiblesse* est plus équitable entre les paires d'avions en terme de risque de perte de la séparation. L'incertitude est modélisée comme dans [17]. Les résultats numériques ont confirmé l'observation de Chandran et Ba-

lakrishnan [17] : la robustesse, mesurée par la fiabilité, et l'efficacité sont deux objectifs concurrents. Également, dans le cas déterministe, l'auteur a noté qu'il est plus avantageux de minimiser le retard moyen que de maximiser directement le rendement de piste. En effet, ce dernier objectif peut augmenter le retard moyen, alors que minimiser le retard moyen profite aux deux objectifs. Finalement, permettre d'avancer les avions jusqu'à 3 minutes par rapport à leurs heures prévues s'est avéré bénéfique en termes de coût du retard subi par les compagnies aériennes.

2.2.1 Modèles stochastiques à deux étapes

À notre connaissance, Sölveling, Solak, Clarke, et Johnson [65, 66, 67] sont les seuls auteurs à avoir proposé des modèles de programmation stochastique à deux étapes pour la version stochastique du problème d'ordonnancement des avions. Dans la suite, nous présentons ces contributions dans leur ordre de publication.

Article de Sölveling et al. [67]

L'article de Sölveling et al. [67], paru en 2011, s'avère être le premier à adopter l'approche de la programmation stochastique à deux étapes pour le problème d'ordonnancement des avions, c'est-à-dire des atterrissages et des décollages. Les auteurs ont proposé un modèle dans le cas statique puis, dans leur étude numérique, l'ont intégré dans un schéma de résolution par horizon roulant afin de traiter le cas dynamique. Une itération de l'algorithme d'horizon roulant se passe en deux phases ordonnées chronologiquement :

— Première phase : optimiser la séquence des classes de turbulence ;
— Deuxième phase : affecter les avions aux positions dans la séquence optimisée tout en déterminant leurs heures cibles d'atterrissages.

Dans la première phase, les heures prévues d'atterrissage et de décollage sont considérées comme aléatoires. Un programme stochastique à deux étapes est alors résolu pour fournir une séquence optimale des classes de turbulence à opérer sur la piste. Par exemple, la piste opère un atterrissage d'un gros porteur (H), ensuite un décollage d'un avion de turbulence moyenne (M) et enfin un atterrissage d'un avion de faible turbulence (L). À mesure que le temps avance, les heures prévues d'atterrissage et de décollage deviennent connues avec certitude. Ainsi, en deuxième phase, les avions sont affectés, selon leurs classes de turbulence, aux positions dans la séquence à la piste, déterminée en première phase. Les heures cibles d'atterrissage et de décollage sont également calculées en respectant les contraintes de séparation.

Les auteurs modélisent le problème de la première phase comme un programme stochastique à deux étapes, alors que le problème de la deuxième phase est formulé comme un programme linéaire en variables mixtes.

Pour justifier l'intérêt d'optimiser la séquence des classes de turbulence, les auteurs rappellent l'observation de [2] que la qualité d'une séquence à la piste est déterminée par les classes de turbulence des avions et non par les autres caractéristiques individuelles des avions. Également, la séquence des classes de turbulence représente, dans une certaine mesure, une information plus robuste que la séquence des avions individuels.

Programme stochastique à deux étapes. Le problème d'optimisation de la première phase (cherchant à trouver la séquence des classes de turbulence et des opérations

à effectuer sur la piste) est formulé comme un programme stochastique à deux étapes. Les deux étapes de décision sont les suivantes :

— Première étape : affectation des classes de turbulence des avions à des créneaux prédéfinis, formant ainsi la séquence à la piste, en minimisant le coût de retarder le dernier avion dans la séquence initiale ;
— Seconde étape : affectation des avions aux positions dans la séquence des classes de turbulence (trouvée en première étape) en minimisant le coût d'affectation espéré, calculé sur un certain nombre de scénarios.

Remarquons qu'en première étape, le coût de retarder le dernier avion est une fonction linéaire par morceaux du retard de l'heure cible *estimée* associée à la dernière position de la séquence par rapport à la dernière heure prévue (une donnée du problème) des avions à ordonnancer. Également, en deuxième étape, le coût d'affectation d'un avion à une position dans la séquence correspond au coût de déviation entre une *estimation* de l'heure cible associée à la position et l'heure d'opération de l'avion (connue suite à la révélation de l'incertitude). En conséquence, les coûts dans le programme stochastique à deux étapes ne sont que des estimations des coûts réels qu'ils représentent.

Résolution du programme stochastique à deux étapes. Remarquons que le problème de deuxième étape est un pur problème d'affectation. Ainsi, son polyèdre des contraintes est entier. Résoudre le problème de deuxième étape en nombres entiers revient donc à résoudre sa relaxation linéaire. Le problème de deuxième étape se simplifiant à un problème de programmation linéaire, les auteurs ont choisi la décomposition de Benders classique pour résoudre leur programme stochastique à deux étapes.

Incertitude. Afin de quantifier l'incertitude, les auteurs ont analysé des données réelles sur les retards au repoussage (en anglais, *push-back*) (pour les départs) et sur les déviations par rapport aux heures prévues d'atterrissage (pour les arrivées). Même si les lois de probabilités identifiées étaient continues (loi lognormale et bêta, respectivement), et afin de limiter le nombre total de scénarios, ces lois ont été discrétisées en trois points : l'espérance et plus et moins l'écart-type.

Résultats. Suite à leur étude numérique, les auteurs ont montré que le processus d'ordonnancement dynamique (reposant dans sa première phase sur un programme stochastique à deux étapes), qu'il proposent, est préférable à la politique FCFS et à un ordonnancement déterministe, dans le cas d'un trafic dense (où la demande dépasse la capacité des pistes). Par contre, pour un trafic dense et varié en terme de classe de turbulence, les temps de résolution du modèle stochastique ne sont pas adaptées au temps réel : la simulation d'un scénario de deux heures a pris environ 160 minutes.

Thèse de doctorat de Sölveling [65]

La thèse de de doctorat de Sölveling [65], soutenue en 2012, formalise la version stochastique du problème d'ordonnancement d'avions, révise le modèle à deux étapes déjà publié [67], jugé simplificateur, et présente une nouvelle méthode de résolution : le *branch-and-bound stochastique*. L'auteur a défini le problème d'ordonnancement stochastique à la

piste (*Stochastic Runway Scheduling Problem - SRSP*) comme le problème stochastique formé des deux étapes suivantes :

— Première étape : la séquence des classes de turbulence est déterminée
— Seconde étape : les avions sont affectés aux positions dans cette séquence et leurs heures cibles sont calculées

Rappelons que le problème stochastique à deux étapes proposé dans [67] n'incluait pas (dans sa deuxième étape) le calcul des heures cibles d'atterrissage et de décollage des avions. Ces heures cibles étaient calculées par un programme linéaire mixte indépendant, en une deuxième phase, dans le cadre d'un algorithme par horizon roulant.

Variantes et formulations mathématiques. Plus précisément, dans sa thèse, Sölveling définit deux variantes du problème : *restreinte* et *complète*. D'une part, la variante *restreinte* compte tous les éléments du problème stochastique d'ordonnancement des avions tout en s'alignant parfaitement avec les problèmes classiques d'ordonnancement. D'autre part, la variante dite *complète* capture certains détails supplémentaires, augmentant son réalisme et sa complexité, tels que la possibilité d'avancer un avion par rapport à son heure prévue, minimiser une fonction de coût non-linéaire, etc. L'auteur a proposé deux formulations mathématiques pour modéliser la variante restreinte. Les deux formulations possèdent le même objectif pondéré : minimiser la longueur de la séquence (donnée par la somme des minima de séparation entre les positions successives) et le coût total du retard (calculé sur tous les avions).

— La première formulation est basée sur un modèle de flots dans un réseau (*network-flow based formulation*). Elle contient des variables supplémentaires modélisant les flots sur les arcs.
— La deuxième formulation reprend le modèle de [67], appelée formulation basée sur les créneaux (*slot formulation*).

Ces deux formulations ont été ensuite étendues à la version complète du problème d'ordonnancement. Remarquons tout de même que la fonction de coût non-linéaire, étant convexe, a été transformée en une fonction linéaire par morceaux (toujours convexe). Afin d'améliorer ces formulations pour la version complète, des inégalités valides ont été introduites.

Méthode de résolution. Contrairement à l'article [67], Sölveling a abandonné, dans [65], la résolution par décomposition de Benders. En effet, cette méthode n'était plus adaptée puisque la deuxième étape ne correspond plus à un problème de programmation linéaire, l'auteur a donc opté pour la relaxation lagrangienne où les contraintes, dite de *non-anticipativité*, sont relaxées. La méthode de sous-gradient a été employée avec plusieurs améliorations pour la mise à jour des bornes au fil des itérations. Le nombre total de scénarios étant très grand, les fonctions de coût étaient estimées par échantillonnage Monte Carlo. Remarquons que la quantification de l'incertitude est la même que dans [67] où les lois ont été discrétisées en trois points. Les nombres de scénarios générés par échantillon sont 8 et 12, alors que le nombre maximal des échantillons était 100.

Résultats. L'étude numérique menée par Sölveling dans sa thèse [65] visait principalement à comparer les deux formulations (*network* et *slot formulation*) pour chaque variante (restreinte et complète) dans les deux cas : déterministe et stochastique. Les for-

mulations dans le cas déterministe ont été résolues par branch-and-bound. Les conclusions pour les deux cas déterministe et stochastique sont similaires :

- la formulation basée sur les flots (*network formulation*) est préférée pour résoudre :
  - la variante restreinte (les instances déterministes avec 10 avions peuvent être résolues quasiment en temps réel) ;
  - la variante complète quand les instances sont complexes en terme de densité et de composition du trafic ;
- la formulation en créneaux (*slot formulation*) donne de meilleurs temps de calcul pour les instances plus faciles de la variante complète.

La variante complète étant celle d'intérêt, l'auteur s'y est concentré pour étudier l'efficacité de son implémentation en pratique, et ce en limitant le temps de résolution. Il a été remarqué que les résultats obtenus en 20 minutes sont suffisamment proches de ceux obtenus en deux heures, suggérant l'intérêt prometteur de l'implémentation en pratique de son algorithme avec résolution tronquée.

En plus du problème stochastique à deux étapes résolu par relaxation lagrangienne et échantillonnage Monte Carlo, Sölveling a proposé dans sa thèse une deuxième méthode pour trouver la séquence des classes de turbulence : le *branch-and-bound stochastique*.

Branch-and-bound stochastique. Puisée dans la littérature de l'ordonnancement, le branch-and-bound stochastique est une adaptation de l'algorithme classique de branch-and-bound avec prise en compte de l'incertitude, basée sur la méthode Monte Carlo. L'espace des solutions est partitionné et des bornes sur la valeur optimale dans chaque nœud de l'arbre de recherche sont calculées. Ce calcul n'étant pas déterministe (calcul d'estimations statistiques sur un ensemble de scénarios), l'élagage classique n'est pas possible. À la place de l'élagage, la taille de l'échantillon par nœud est changée dynamiquement pour orienter la recherche dans l'arbre du branch-and-bound. Typiquement, les zones les plus prometteuses voient la taille de leur échantillon augmenter.

Le branch-and-bound stochastique permet de trouver la séquence des classes de turbulence de sillage à la piste. Ensuite, un programme linéaire est résolu pour trouver les heures cibles des avions individuels. Notons que l'étude numérique a été effectuée pour les atterrissages et les décollages sur un doublet de pistes hétérogènes mais dépendantes. En plus, une variable de décision supplémentaire était nécessaire pour chaque avion en atterrissage : l'heure de la traversée de la piste de décollage pour rejoindre le parking. Les incertitudes sur les heures prévues des avions ont été supposées suivre des lois triangulaires. Par rapport à un modèle déterministe, les gains enregistrés en termes de *makespan* et de retard total sont de 4% et 26% respectivement. Plusieurs améliorations à cet algorithme ont été présentées permettant de trouver des solutions de très bonne qualité d'une instance de 14 avions en 10 minutes gagnant ainsi 30% à 70% en temps de calcul.

Article de Sölveling et Clarke [66]

Dans un article plus récent [66], Sölveling et Clarke ont réutilisé le branch-and-bound stochastique sur le problème d'ordonnancement des avions mais en redéfinissant les deux étapes de décision comme suit :

- Première étape : trouver la séquence des avions individuels ;

— Seconde étape : trouver les heures cibles de façon à respecter les séparations et les fenêtres de temps.

Les auteurs ne se sont plus intéressés à la séquence des classes de turbulence en première étape comme en [65] mais se sont directement penchés sur la séquence des avions. Le problème de première étape ainsi défini est difficile, tandis que celui de deuxième étape est trivial. L'incertitude est modélisée comme dans [65].

Les résultats numériques suggèrent une meilleure performance par rapport à [65] : une instance de 14 avions est résolue en une minute. Le gain en *makespan* est de 5 à 7% par rapport à un planificateur déterministe. Finalement, une étude de sensibilité suggère que l'intérêt de formuler le problème d'ordonnancement des avions sous incertitude (et de le résoudre par l'algorithme du branch-and-bound stochastique) augmente avec l'incertitude.

2.2.2 Autres travaux

Certains autres articles à orientation opérationnelle, principalement avec l'implication de la NASA, [13, 14, 74, 75] ont tenté d'optimiser l'utilisation de l'espace aérien (TMA) et de l'infrastructure aéroportuaire tout en considérant des heures prévues aléatoires. La démarche et les résultats étant plus complets dans [75] que dans [74], nous choisissons de résumer dans la suite l'article le plus récent.

Article de Xue et Zelinski [75]

Dans [75], les auteurs se sont intéressés aux opérations d'atterrissage, de décollage et de roulage à l'aéroport international de Los Angeles (LAX), possédant quatre pistes. Le problème est formulé sur la région de contrôle (TMA) et sur la surface aéroportuaire, modélisées par un graphe formé des points d'entrée (les IAF pour les arrivées et les portes d'embarquement pour les départs), de repères intermédiaires ainsi que des routes de montée et de descente. Pour chaque avion, quatre décisions sont prises : retard, vitesse, route et piste. Les incertitudes sont prises en compte au niveau des points d'entrée du graphe (IAF et porte d'embarquement). Les heures prévues aux IAF suivent une distribution normale de moyenne nulle et d'écart-type 30 secondes. Pour les départs, les heures prévues pour le repoussage sont également des variables aléatoires normales, mais de moyenne 30 secondes et d'un écart-type linéairement croissant en fonction du temps d'anticipation (plus l'heure prévue est lointaine, plus l'incertitude est grande).

Les auteurs ont employé un algorithme génétique multi-objectif (*Non-dominated Sorting Genetic Algorithm - NSGA*) intégré à un horizon roulant pour à la fois minimiser le retard total et le nombre des interventions d'un pseudo-contrôle aérien. Le compteur des interventions de contrôle est incrémenté chaque fois qu'il y a violation de la séparation. Selon l'étude numérique présentée, le planificateur stochastique proposé permet de réduire les retards de 28 à 40%, comparé à un planificateur déterministe et ce pour un même nombre d'interventions des contrôleurs aériens. Les longueurs de la fenêtre d'optimisation de l'horizon roulant suggérées vont de 2 à 8 minutes.

Article de Bosson et al. [13]

Les auteurs ont étudié le même problème intégré d'optimisation que [75], où les opérations d'atterrissage, de décollage et de roulage à l'aéroport international de Los

Angeles (LAX) sont à ordonnancer. Les auteurs ont réutilisé le même graphe que celui dans [75] pour modéliser l'espace terminal et la surface aéroportuaire. Ils ont formulé un programme linéaire à variables mixtes, formé de trois étapes séquentielles de décision. La première étape est déterministe tandis que les deux autres sont altérées par l'incertitude, selon une structure imbriquée : pour un scénario de révélation de l'incertitude en deuxième étape, il y a un ensemble complet de scénarios de révélation de l'incertitude en troisième étape. Une telle structure rappelle celle d'un arbre de scénario utilisé dans un programme stochastique à trois étapes. Les données affectées par l'incertitude sont les suivantes. Pour les arrivées, ce sont les heures prévues de passage à l'IAF ainsi que les heures prévues d'arrivée à la porte de débarquement après le roulage. Pour les départs, l'incertitude est prise en compte au niveau du repoussage et du dernier point de la trajectoire de montée dans l'espace aérien considéré. Leurs résultats sont comparés à la politique FCFS et semblent prometteurs pour une application en temps réel.

Bosson et Sun [14] reprennent les mêmes idées de [13] en modélisant uniquement la surface aéroportuaire comme un graphe, alors que la zone terminale n'est plus considérée. Ainsi, pour les vols en arrivée, le point d'entrée du graphe est la piste d'atterrissage, et les points de sortie sont les portes de débarquement. Pour les vols au départ, les points d'entrée sont les portes d'embarquement, et le point de sortie est la piste de décollage. Leurs résultats sont comparables à ceux dans [75].

Ordonnancement robuste des atterrissages

Outre que l'approche stochastique, d'autres auteurs [37, 38, 42] ont opté pour des approches d'optimisation robuste. Dans de telles approches, les données incertaines sont supposées varier dans un ensemble connu de valeurs possibles, appelé ensemble d'incertitude (contrairement à la programmation stochastique où les données incertaines suivent des lois de probabilité connues à l'avance). L'optimisation robuste prend donc en compte toutes les valeurs possibles des données incertaines, permettant ainsi de se protéger contre le pire cas. Toutefois, pour des ensembles d'incertitude très grands, une solution robuste peut présenter l'inconvénient d'être trop conservatrice. Dans la suite, nous présentons trois travaux ayant appliqué le paradigme de la programmation robuste à des variantes du problème d'ordonnancement des atterrissages.

Article de Kapolke et al. [42]

Définition du problème. Kapolke et al. [42] considèrent la phase *pré-tactique*, à quelques heures en amont des opérations sur la piste. Les auteurs formulent le problème d'optimisation consistant à affecter les avions à des créneaux d'atterrissage afin de minimiser les déviations par rapport aux heures prévues (autrement dit, maximiser la ponctualité), sous les contraintes de fenêtres de temps et en prenant en compte les séparations minimales au seuil de piste. Une hypothèse majeure est faite : un même créneau d'atterrissage peut être affecté à plusieurs avions. Ce nombre d'avions est calculé en fonction de la longueur du créneau et des séparations de turbulence de sillage nécessaires entre les avions concernés. La détermination avec précision des heures cibles n'est pas traitée, car une telle décision relève de la phase tactique (typiquement 30 minutes avant l'heure prévue d'atterrissage, selon les auteurs) non modélisée dans l'article.

Prise en compte de l'incertitude. Suite à la formulation du problème déterministe, les auteurs proposent d'incorporer l'incertitude sur les heures au plus tôt et au plus tard d'atterrissage, selon deux paradigmes : la programmation robuste (via des ensembles d'incertitude) et la programmation stochastique (via des distributions de probabilité). Au total, trois formulations prenant en compte l'incertitude sont proposées. Selon les auteurs, l'approche par ensembles d'incertitude permet de se protéger contre le pire cas, alors que la prise en compte des valeurs moyennes uniquement est moins conservatrice. Un compromis entre les deux approches est une troisième approche qualifiée de "robustesse restaurable" (*recoverable robustness*). Cette dernière approche, considérée comme une version relâchée de l'approche robuste, calcule, en plus de la solution du problème, une action de réparation de cette solution (à une solution réalisable) pour chaque scénario de l'incertitude (si non réalisable). Le coût de ces actions de réparation est également intégré dans la fonction objectif. Vu la difficulté que présente le modèle mathématique obtenu par cette approche, les auteurs se sont limités à calculer une action de réparation à la solution "strictement" robuste (obtenue pour le pire cas). Remarquons que l'approche par programmation stochastique, adoptée par les auteurs, réduit les données stochastiques à leur valeurs moyennes, et par conséquent ne traite que le scénario moyen. Rappelons que résoudre le problème correspondant au scénario moyen correspond à une approche purement déterministe, ignorant la stochasticité des données incertaines. Typiquement, en programmation stochastique à deux étapes, la valeur de la solution stochastique permet de quantifier le gain en valeur de la fonction objectif entre l'approche stochastique et une approche déterministe basée sur le scénario moyen. Quant aux méthodes de résolution, comme chaque modélisation de l'incertitude donne lieu à une formulation différente de programmation linéaire en variables mixtes, ces différentes formulations sont résolues par un solveur générique.

Les résultats numériques sont en concordance avec les attentes des auteurs. L'approche déterministe est la moins stable face à l'incertitude, même si elle affiche le retard le plus faible, en moyenne sur tous les avions, comparée aux autres approches. D'autre part, l'approche d'optimisation "robuste restaurable" propose le meilleur compromis entre la stabilité et l'efficacité.

Article de Heidt et al. [38]

Heidt et al. [38] ont traité le problème d'ordonnancement robuste à la piste. Les auteurs ont proposé un modèle où le temps est discrétisé (*time-indexed model*) pour le cas déterministe. L'avantage d'un tel modèle est de pouvoir représenter des coûts quadratiques de déviation par rapports aux heures prévues, sans introduire des non-linéarités. Il reste donc facile à résoudre. Ce modèle a été ensuite augmenté pour prendre en compte l'incertitude. Deux méthodes de l'optimisation robuste ont été employées : la robustesse "stricte" (*strict robustness*), et la robustesse "légère" (*light robustness*). La robustesse classique stricte protège contre toutes les valeurs de l'ensemble d'incertitude, mais dégrade la valeur de la fonction objectif, à cause de son conservatisme. Par opposition, la robustesse légère limite la dégradation de la valeur optimale à un degré prédéterminé par le décideur, au prix de fournir une solution potentiellement non réalisable pour certaines valeurs de l'incertitude. Quatre algorithmes d'ordonnancement, dont un déterministe et trois robustes, ont été testés et comparés, pour des niveaux d'incertitude et des densité de trafic différents. Les auteurs ont recommandé l'algorithme basé sur le concept de la robustesse "légère", pour un trafic de densité moyenne, avec un niveau d'incertitude élevé, alors que l'algorithme utilisant la robustesse stricte serait préférable pour un trafic plus

dense, toujours en présence d'une incertitude importante.

Thèse de Heidt [37]

Heidt a consacré sa thèse, soutenue en 2017, aux modèles de programmation robuste pour la planification de trafic aérien. Dans une première partie, le cas déterministe est étudié et une formulation le temps est discrétisé est proposée. Dans une deuxième partie, l'auteur a cherché à définir la robustesse / résilience dans le contexte de la gestion du trafic aérien. Il en a déduit que le paradigme de l'optimisation robuste est adapté pour traiter les problématiques dans ce contexte. Des approches dites robuste stricte ont été étudiées. Vu leur conservatisme, des approches plus avancées ont été proposées, telles que la robustesse "restaurable" et la robustesse "légère". Finalement, un environnement de simulation pour la planification du trafic aérien en phase tactique avec prise en compte de l'incertitude a été développé.

Chapitre 3

Démonstration de faisabilité et étude numérique – *A proof of concept and a numerical study*

Note for English readers. This chapter corresponds to the article entitled "*Extended aircraft arrival management under uncertainty : A computational study*" published in the *Journal of Air Transportation* [46].

Résumé du chapitre

Ce chapitre reprend l'article "*Extended aircraft arrival management under uncertainty : A computational study*" publié dans la revue *Journal of Air Transportation* [46]. Il est donc rédigé en anglais.

L'article se veut une démonstration de faisabilité du concept de l'ordonnancement étendu des arrivées d'avions sous incertitude en s'appuyant sur le paradigme de la programmation stochastique à deux étapes. L'ordonnancement "étendu" des arrivées d'avions, connu sous le nom de *Extended Arrival Management*, a pour objectif d'optimiser l'utilisation de la piste en commençant l'ordonnancement des arrivées d'avions quelques heures avant leur atterrissage, tout en minimisant les actions de dernière minute.

Après une brève revue de littérature, nous présentons le problème d'ordonnancement des arrivées d'avions sous incertitude, où les avions sont captés à 2–3 heures de l'aéroport de destination. L'article se base sur une formulation du problème comme un problème de programmation stochastique à deux étapes. En première étape, étant données des heures planifiées de passage par un même point d'approche initiale, les avions captés sont ordonnancées sur ce point d'approche : une séquence cible et des heures cibles sont alors cherchées afin d'optimiser un critère de première étape, tout en satisfaisant des contraintes opérationnelles. Plusieurs sources d'incertitudes (météo, etc) entraînent un décalage entre les heures cibles et les heures effectives de passage par ce point d'approche. Dans le cadre de cette thèse, ces déviations sont considérées comme des variables aléatoires de lois connues. En deuxième étape, les avions sont supposés être suffisamment proches du point de début d'approche (IAF), permettant ainsi de connaître sans incertitude l'heure effective de passage par ce point. La décision subséquente en deuxième étape consiste à donner des heures cibles d'atterrissage aux avions considérés, selon la séquence cible prédéterminée en première étape, tout en satisfaisant un ensemble de contraintes

opérationnelles. L'objectif de deuxième étape est de minimiser la somme des déviations entre les heures cibles d'atterrissage et les heures non contraintes d'atterrissage, correspondant aux heures d'atterrissage dans un espace aérien terminal décongestionné. Notons que, pour un avion donné, une telle déviation correspond au temps à faire gagner (*time to gain*) ou à faire perdre (*time to lose*) à l'avion dans l'espace terminal ou aux zones d'attente limitrophes (*holding patterns*). Utilisant un modèle de programmation stochastique à deux étapes résolu en boîte noire avec CPLEX, nous étudions quelques caractéristiques des instances du problème (largeur des fenêtres de temps et amplitude de l'incertitude) et certains paramètres d'optimisation (pondération de l'objectif de première étape et nombre de scénarios de deuxième étape). L'article démontre l'intérêt de prendre en compte l'incertitude dans l'ordonnancement étendu des arrivées d'avions. Il illustre également l'avantage de l'ordonnancement étendu des arrivées d'avions où l'attente circulaire proche de la zone terminale est transformée en attente linéaire.

Table of content

Abstract

The arrival Manager operational horizon, in Europe, is foreseen to be extended up to 500 nautical miles around destination airports. In this context, arrivals need to be sequenced and scheduled a few hours before landing, when uncertainty is still significant. A computational study, based on a two-stage stochastic program, is presented and discussed to address the arrival sequencing and scheduling problem under uncertainty. This preliminary study focuses on a single Initial Approach Fix and a single runway. Different problem characteristics, optimization parameters as well as fast solution methods for real-time implementation are analyzed in order to evaluate the viability of our approach. Paris-Charles-De-Gaulle airport is taken as a case study. A simulation-based validation experiment shows that our approach can decrease the number of expected conflicts near

the terminal area by up to 70%. Moreover, in a high-density traffic situation, the total time-to-lose inside the terminal area can be decreased by more than 71%, while the expected landing rate can be increased by 7.7%, compared to the first-come first-served policy. This computational study demonstrates that sequencing and scheduling arrivals under uncertainty, a few hours before landing, can successfully diminish the need for holding stacks by relying more on upstream linear holding.

3.1 Introduction

Air traffic world-wide growth puts more and more pressure on major airports to better use their infrastructure in order to meet the required levels of safety and efficiency. Since the early 90's in the USA and Europe, air traffic controllers (ATCOs) around major airports have been using decision support tools that compute "optimal" sequences and schedules of landings at the available runways. In Europe, such tools are called Arrival Managers (AMANs). AMAN typically captures inbound aircraft at a distance of 100-200 nautical miles (NM) around the destination airport (30-45 minutes before landing). In the last few years, extending AMAN operational horizon up to 500 NM (2 to 3 hours before landing) has been identified as one key measure to limit delays and to enhance punctuality and eco-efficiency [70]. In fact, starting the sequencing and scheduling process earlier will help aircraft fly more fuel-efficient trajectories and avoid congestion in the terminal area. The foreseen European tool is often referred to as Extended AMAN (E-AMAN). This new decision support tool is expected to diminish the need for *holding patterns* near the terminal area, a stressful air traffic control (ATC) technique for both controllers and pilots in top of being extremely eco-inefficient. However, when the sequencing-and-scheduling horizons are extended, uncertainties about predicted times of arrival get large and cannot be overlooked. A possible technique to hedge against uncertainty is to re-optimize as soon as input data is updated. However, under large uncertainty, re-optimizing a deterministic model each time input data is updated, will deliver unstable solutions, making their implementation impractical. Instead of frequently re-optimizing deterministic models, uncertainties can be embedded within the optimization model in order to obtain more stable solutions. In this paper, we present a computational study based on a two-stage stochastic program addressing the problem of arrival sequencing and scheduling under uncertainty, two to three hours look-ahead time. Numerical tests on realistic instances from Paris-Charles-De-Gaulle airport (CDG) are presented and discussed. In the following, the related literature is briefly reviewed before giving the paper outline.

3.1.1 Literature review

The problem of sequencing and scheduling arrivals on a given destination airport has been studied for several decades [22, 8]. 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 runway throughput and/or minimizing aircraft delay. Operational constraints, mainly minimum separations called *final approach separations*, have to be satisfied between aircraft near the runway threshold. Variants of this problem may consider a single or multiple runways [5, 33, 51]. Also, when more operations (departures, runway crossings, etc) are included, the problem is often called the Aircraft Scheduling/Sequencing problem (ASP). The case in which predicted operations times are known with certainty, called the *deterministic* case, has been thoroughly studied

in the literature [5, 3, 9], while the case under uncertainty has less often been addressed. So far in the related literature, three main approaches to optimization under uncertainty were applied to ASP : probabilistic [55, 50], stochastic [67, 66, 14] and robust [38, 37, 42] approaches. Pioneer studies such as [55, 50] mainly enriched deterministic models by probability constraints and/or by a probability objective-value function. Stochastic programming models, including two-stage and multi-stage models, were proposed in [67, 66, 14]. Finally, [38, 37, 42] proposed and studied several robust programming models for the runway scheduling problem. Apart from Kapolke et al. [42], all of the aforementioned studies concentrated on variants of ASP involving a short planning horizon of around 45 minutes. Kapolke et al. [42] addressed the pre-tactical aircraft landing problem under uncertainty, where aircraft are captured a few hours before landing.

3.1.2 Contribution and paper outline

In the context of the present paper, we seek to optimally sequence and schedule aircraft over the same Initial Approach Fix (IAF), two to three hours before landing. We propose a computational study based on a two-stage stochastic programming model. In the first stage, an aircraft sequence and a schedule at the IAF are found so as to maximize the expected runway throughput under uncertain arrival times at the IAF. In the second stage, uncertainty is assumed to be revealed and the landing times are computed so as to minimize a time-deviation impact cost in the terminal area. Focusing on realistic instances from CDG airport, we study the compromise between flexibility and punctuality when scheduling at the IAF as well as the effect of uncertainty amplitude. We examine fast solution methods from the literature and compare them to a different proposed approach under a limited computing-time budget. The aim of our computational study is to evaluate various model and optimization-method parameters as well as different resolution approaches, within the framework of two-stage stochastic programming, in order to design an efficient algorithm for real-time implementation. Also, since the first-come first-served (FCFS) policy is widely used in practice to schedule arrivals from the IAF to the runway threshold, we evaluate the benefit of our approach (sequencing and scheduling arrivals at the IAF with few hours look-ahead time) for the FCFS policy performance in the terminal area.

The remaining of this paper is organized as follows. First, the problem is described in Section 3.2. The solution method is explained in Section 3.3. The computational study is presented and discussed in Section 3.4. Finally, conclusions are drawn in Section 3.5.

3.2 Problem statement

We consider a set of n aircraft planning to land at a given destination airport in two to three hours look-ahead time. Let $\mathcal{A} = \{1, \ldots, n\}$ be the set of their indices. We make the following two operational assumptions. Firstly, all aircraft will fly to the same IAF before entering the airport terminal area. Secondly, all aircraft will land on the same runway of the considered airport. We are given a *predicted time at the IAF* for each aircraft. We seek to sequence and schedule these aircraft to the IAF so as to maximize the expected landing rate as well as to minimize a time-deviation impact cost in the terminal area. To that aim, we first introduce the continuous decision variable x_i as the *target time at the IAF* for aircraft $i \in \mathcal{A}$. Target times at the IAF must satisfy two types of constraints detailed below : minimum-time separation at the IAF, and time-windows constraints.

3.2.1 Minimum-time separation at the IAF

We assume that all aircraft pass over the IAF at the same altitude (*i.e.* the same flight level) so that only longitudinal separation between aircraft matters. Aircraft successively passing over the IAF have to be separated by a distance-based minimum separation. For modeling and optimization purposes, this minimum separation at the IAF is converted to time. Let us note $\underline{S}^I$ the time-based minimum separation at the IAF expressed in seconds. Assuming all aircraft speeds over the IAF are equal to 250 knots and a distance-based minimum separation at the IAF of 5 NM, $\underline{S}^I$ may then be set to 72 seconds [55].

3.2.2 Time windows at the IAF

Imposing a time window constraint at some point of an aircraft trajectory either reflects its physical limitations (mainly its lowest and highest eligible speeds) or acceptable deviations with respect to some reference time as may be expressed by airlines or in order to ensure some level of air traffic punctuality. In our context, a target time at the IAF for each aircraft has to be found within a predefined time window, noted TW^I, around the predicted arrival time at the IAF.

We assume that actual times at the IAF will randomly deviate from the target times following a normal distribution with mean μ and standard deviation σ. In order to define the aircraft sequence over the IAF, we then introduce binary decision variables δ_{ij} for each pair of aircraft (i, j) such that $i \neq j$:

$$\delta_{ij} = \begin{cases} 1 & \text{if aircraft } i \textbf{ directly precedes} \text{ aircraft } j \\ 0 & \text{otherwise} \end{cases}$$

From the ATC perspective, aircraft will ideally pass over the IAF in the same sequence that maximizes the landing rate so that ATCOs in the terminal area will only have to "compress" the sequence, without shifting any aircraft position inside the terminal area. This may be achieved by enforcing the target sequence over the IAF to be an optimal landing sequence. An optimal landing sequence is one that has a minimal length in terms of final-approach separations. Let us note S_{ij} the minimum time-based final approach separation in seconds between a leading aircraft i and a following aircraft j. Final-approach separations are defined in terms of distance in NM according to the wake-turbulence categories (Heavy (H), Medium (M) and Light (L)) set by the International Civil Aviation Organization (ICAO). Final-approach separations applicable in CDG airport and converted into seconds are given in Table 3.1.

Table 3.1 – Final-approach separations (seconds) according to ICAO wake-turbulence categories [28]

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

Given the binary decision variables introduced above, the landing sequence length may

be computed using the following expression :

$$\sum_{\substack{(i,j)\in\mathcal{A}\times\mathcal{A}\\ i\neq j}} \delta_{ij} S_{ij}$$

Besides anticipating the optimal landing sequence, making aircraft arrive as early as possible at the IAF is equivalent to minimizing their average flight time during the en-route and descent phases, prior to the IAF. Such an objective may help flights catch up with their upstream delay, if any, in an attempt to improve punctuality. For that reason, we may try to minimize the sum of target times at the IAF, $\sum_{i\in\mathcal{A}} x_i$. Overall, we seek to minimize the following objective function, where x denotes the vector whose ith component is x_i for $i \in \mathcal{A}$, δ stands for the matrix whose ij-entry is δ_{ij} for $(i,j) \in \mathcal{A}\times\mathcal{A}$ such that $i \neq j$ and λ is a user-defined weighting parameter :

$$f_1(x,\delta) = \sum_{\substack{(i,j)\in\mathcal{A}\times\mathcal{A}\\ i\neq j}} \delta_{ij} S_{ij} + \lambda \sum_{i\in\mathcal{A}} x_i \tag{3.1}$$

In order to account for the expected air traffic situation in the terminal area, we consider a hypothetical second-stage problem in which actual arrival times at the IAF are assumed to be known with certainty. The beginning of the second stage can be set to the entry time of the last considered aircraft to the en-route sector neighbouring the terminal area. Defined as such, the second-stage problem corresponds to a deterministic aircraft sequencing-and-scheduling problem with a short operational horizon. In this second-stage problem, we schedule aircraft to the runway threshold so as to minimize the total time-deviation impact cost during the approach phase in view of eliminating congestion in the terminal area and improving punctuality.

The landing sequence is enforced to be the same as the already-found target sequence over the IAF, although the actual sequence over the IAF may be different once uncertainty is revealed. This is due to the fact that deviations of actual times at the IAF with respect to the target times may change the actual sequence over the IAF with respect to the target sequence. Since the landing sequence is already found in the first stage, no sequencing variables are needed in the second stage. Hence, we introduce the continuous decision variable y_i as the *target landing time* of aircraft $i \in \mathcal{A}$. Similarly to the target arrival times at the IAF, target landing times should satisfy two types of constraints : minimum time-based final-approach separation and time-windows constraints for landing. We define U_i the *unconstrained landing time* of aircraft i corresponding to the landing time of aircraft i as if it were alone in the terminal area flying its preferred trajectory at its preferred speed. For each aircraft $i \in \mathcal{A}$, U_i is computed as the sum of its actual arrival time at the IAF and an *unconstrained flight time* from the IAF to the runway threshold. Remark that the unconstrained landing time, computed at the beginning of the second stage, can be seen as the latest up-to-date estimated landing time available at around 30-minute look-ahead time. This unconstrained landing time is defined so that aircraft i flies its preferred trajectory at its preferred speed in the terminal area until landing, thereby saving fuel, and landing with no extra delay incurred during the approach phase. Hence, $|y_i - U_i|$ quantifies the (unweighted) impact cost of time deviation from the unconstrained landing time of aircraft i during the approach phase. In this paper, we assume that deviating from the unconstrained landing time in either directions is equally undesirable. For a study with weighted time-deviation impact costs during the approach phase, see [45]. In the context

of AMAN, a time deviation from the unconstrained landing time of a given aircraft can be seen as the *time-to-gain* or the *time-to-lose* as computed by AMAN, and then displayed to the approach controller. The controller has, then, to apply adequate control actions in order to satisfy this time deviation. Remark that if all inbound flights are able to follow their preferred trajectories at their preferred speed in the terminal area, then such a traffic situation should generate a low workload for both terminal-area controllers and pilots. Such an ideal situation is necessarily, but not sufficiently, described by the equation $\sum_{i\in\mathcal{A}} |y_i - U_i| = 0$. Indeed, an aircraft may succeed to land at its unconstrained time (*i.e.*, $y_i = U_i$), after stretching its path, speeding up, and then slowing down, generating thereby a significant workload for both pilots and controllers.

We define a *second-stage scenario* s as a possible realization of actual arrival times at the IAF of all aircraft in $\mathcal{A}$. For example, consider a set of 3 aircraft with IAF target times 6:00:00 AM, 6:01:30 AM and 6:03:00 AM for aircraft 1, 2 and 3 respectively. One possible realization of their IAF actual times, *i.e.* one possible scenario, is 6:00:32 AM, 6:01:10 AM and 6:03:55 AM respectively. In this scenario, the first and the third aircraft arrive at the IAF later than their target times (by 32 and 55 seconds respectively), while the second aircraft arrives 20 seconds earlier than its target time. Assuming 11 minutes of unconstrained flight time from the IAF to the runway threshold for any of the three aircraft, their unconstrained landing times in this scenario are 6:11:32 AM, 6:12:10 AM and 6:14:55 AM respectively. Remark that unconstrained landing times and target landing times depend on the scenario s. Hence, unconstrained landing time and target landing time of aircraft i in scenario s will be noted U_i^s and y_i^s respectively. Therefore, given a scenario s, the second-stage problem reads :

$$\begin{aligned} Q(x,\delta,s) := \quad & \min_{y^s} \quad \sum_{i\in\mathcal{A}} |y_i^s - U_i^s| \\ & \textit{subject to} \quad \text{final-approach separation constraints} \\ & \qquad\qquad \text{and time-windows constraints for landing} \end{aligned} \tag{3.2}$$

Note that, in our second-stage problem, we do not adjust the first-stage solution but we make new decisions (target landing times) based on the revealed uncertainty.

Since we need to account for all possible scenarios in the second-stage, the objective-function of the entire two-stage stochastic programming model reads :

$$\min_{x,\delta} \quad f_1(x,\delta) + \mathbb{E}\left[Q(x,\delta,.)\right] \tag{3.3}$$

where $\mathbb{E}[.]$ is the expectation operator. In the sequel, we shall need the following notation. Let $v^\star$ be the optimal value of the two-stage stochastic problem. The output of the two-stage model is then an optimal sequence (described by δ_{ij} for all $(i,j) \in \mathcal{A}\times\mathcal{A}$ such that $i \neq j$) and optimal target times at the IAF (x_i for all $i \in \mathcal{A}$) that maximize the expected landing rate and minimize the expected second-stage cost function.

We remark that the above model can be extended to the case of multiple IAFs at the expense of adding extra binary variables and associated constraints for the newly considered IAFs. A similar extension can be envisaged for the case of multiple runways.

The solution method proposed to address the two-stage stochastic program is explained in the next section.

3.3 Solution method

Two-stage stochastic programming models are usually intractable if all possible scenarios are taken into account. Nevertheless, the Sample Average Approximation (SAA) method [63] offers a framework to approximate the solutions of such problems, through solving an approximate problem, called the SAA problem, instead of the original problem. In the approximate problem, the expectation term in the objective function is estimated through a sample average computed over a finite set of scenarios. Given a set $\mathcal{S}$ (called "*training set*") of $n_{\mathcal{S}}$ equiprobable scenarios, the objective function of the SAA problem reads :

$$\min_{x,\delta} \quad f_1(x,\delta) + \frac{1}{n_{\mathcal{S}}} \sum_{s\in\mathcal{S}} Q(x,\delta,s) \tag{3.4}$$

Let $\hat{v}(\mathcal{S})$ be the optimal value of the SAA problem. Since $\hat{v}(\mathcal{S})$ depends on the scenarios set $\mathcal{S}$, $\hat{v}(\mathcal{S})$ is, itself, a random variable. The SAA method [63] guarantees that for any given $n_{\mathcal{S}}$, $\mathbb{E}[\hat{v}(\mathcal{S})] \leq v^{\star}$, *i.e.* the optimal value of the SAA problem is negatively biased. Moreover, under mild conditions, as $n_{\mathcal{S}} \to \infty$, $\hat{v}(\mathcal{S})$ converges towards $v^{\star}$ with probability one. However, in practice, the required computing time grows rapidly with $n_{\mathcal{S}}$. One difficulty with the SAA method is to decide whether a given number of scenarios, $n_{\mathcal{S}}$, is large enough to correctly approximate the original problem, *i.e.* to ensure that the solution obtained is a satisfying approximation of an optimal solution of the original problem. $\hat{v}(\mathcal{S})$ is not necessarily a good-quality indicator with respect to the original problem due to the SAA bias and variance of the SAA optimal value. Hence, a post-optimization validation step is needed to evaluate the quality of any SAA solution. In our study, we rely on the so-called *out-of-sample validation*, that consists in re-evaluating an SAA solution, $\left(\hat{x},\hat{\delta}\right)$, with a *validation set* containing much more scenarios than the training set used to find $\left(\hat{x},\hat{\delta}\right)$. The validation set is believed to represent the complete set of all possible scenarios. An out-of-sample validation provides a new quality indicator for the considered solution, called the *validation score*. When the gap between the SAA optimal value $\hat{v}(\mathcal{S})$ and the validation score of $\left(\hat{x},\hat{\delta}\right)$ is small, the SAA problem can be considered as stable and the training set used may be considered as large enough. Accordingly, in our exploratory computational study detailed in Section 3.4, we investigate solving a sequence of SAA problems involving increasing numbers of scenarios ($n_{\mathcal{S}}$) under a limited, although large, solving time. As we mentioned earlier, SAA solutions depend on the random set $\mathcal{S}$ of scenarios used for optimization. Therefore, to illustrate better the average behavior of SAA problems with a given number of scenarios $n_{\mathcal{S}}$, we build $n_{\mathcal{R}}$ replicated SAA problems with $n_{\mathcal{S}}$ scenarios each. Let $\bar{v}(n_{\mathcal{S}}, n_{\mathcal{R}})$ be the average optimal value obtained over $n_{\mathcal{R}}$ such replications. It can be expressed as follows, where $\mathcal{S}_r$ is a scenario set such that $|\mathcal{S}_r| = n_{\mathcal{S}}$:

$$\bar{v}(n_{\mathcal{S}}, n_{\mathcal{R}}) = \frac{1}{n_{\mathcal{R}}} \sum_{r=1}^{n_{\mathcal{R}}} \hat{v}(\mathcal{S}_r) \tag{3.5}$$

Although a well-applied SAA method can guarantee good-quality solutions for the original problem, computing times are very often inappropriate for real-time implementation. Consequently, fast solution methods based on SAA were proposed in the literature related to aircraft scheduling using stochastic programming [67, 14].

Real-time solution methods

For real-time implementation, mainly two approaches can be built on the SAA method : replication-based and scenario-based approaches.

Replication-based approach

Since solving a single SAA problem with a large scenario set may be time-consuming, [67, 14] opt for solving several replications of an SAA problem with a limited number of scenarios. Upon optimization, they are left with a pool of near-optimal solutions to the original problem (as many solutions as replications). Distinct solutions are kept in a pool from which only one solution need to be selected at the end. On one hand, Bosson and Sun [14] simply select the solution with the minimum objective function value. Hence, no post-optimization validation is performed. On the other hand, Sölveling et al. [67] re-evaluate all distinct solutions from the pool using as a validation set, the complete set of scenarios of their original problem. Then, they select from the pool the solution with the best validation score. Let us note that increasing the number of replications may enlarge the solutions' pool. However, it does not guarantee better solutions.

Scenario-based approach

Unlike replication-based approach, in a scenario-based approach, only one SAA problem with a large enough number of scenarios is solved. Hence, no replications are made. The quality of the obtained solution can be estimated through out-of-sample validation. Let us note that increasing the number of scenarios often leads to better quality solutions.

3.4 Computational study

We rely on a two-stage stochastic program implemented in Julia programming language [10] and on CPLEX 12.6.3 solver. Results are obtained on a Linux platform with 8 x 2.66 GHz Xeon processors and 32 GB of RAM. As a study case, we select the arrivals that planned to enter the terminal area around CDG airport on May 15^{th}, 2015 from 6:00 AM to 6:30 AM and that landed on the north runway (27R). Two realistic instances, involving n =10 and 14 aircraft respectively, are extracted. To conduct our computational study, we consider different problem characteristics and optimization parameters. First, we try to find the best combination of these characteristics and parameters by intensive empirical experiments on the first instance (n =10 aircraft). Once the best combination is identified, we compare two real-time solution methods under a fixed solving-time budget. Using our best setting, we apply it to the second instance (n =14 aircraft). Finally, we evaluate the benefit of our approach (sequencing and scheduling arrivals at the IAF under uncertainty) for the expected performance of a FCFS policy in the terminal area. To that end, different performance indicators are computed through simulation-based experiments.

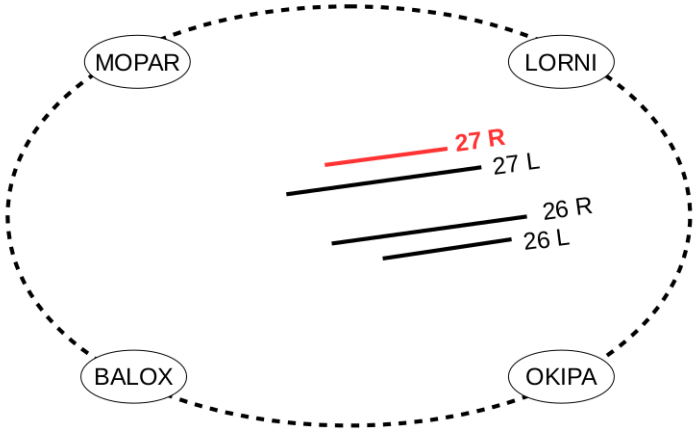
The two instances and the different problem characteristics are described in Subsection 3.4.1. The optimization parameters along with their tested values are presented in Subsection 3.4.2. The main results for the first instance are reported and discussed in Subsection 3.4.3. To select an effective real-time solution method, we compare the performance of the scenario-based and the replication-based approaches on the first instance in Subsection 3.4.4. Results for the second instance (n =14 aircraft) are presented in Subsection 3.4.5.

Finally, the benefit of our approach on the FCFS policy in the terminal area is discussed in Subsection 3.4.6.

3.4.1 Instances and problem characteristics

The terminal area around CDG has four IAF named : MOPAR, LORNI, OKIPA and BALOX. CDG has four runways : two for landings (27R and 26L) and two for departures (27L and 26R). A simplified scheme of CDG runways with the surrounding IAFs is displayed in Figure 3.1.

FIGURE 3.1 – CDG runways scheme with the four surrounding IAFs (not to scale)



On May 15th 2015, looking only at aircraft that finally landed on CDG runway 27R, 10 of these aircraft were planned to enter the terminal area between 6:00 AM and 6:20 AM, while 14 were planned to do so between 6:10 AM and 6:30 AM. All of these aircraft entered the terminal area from three different IAFs (MOPAR, LORNI and OKIPA). For the sake of simplification, we merge all these arrivals as if they were planned to pass over a single IAF. We are then left with two realistic instances satisfying our operational context (a single IAF and a single runway). Details of these instances are summarized in Table 3.2.

TABLE 3.2 – The two considered instances

		instance 1	instance 2
Planned time span at the IAF		6 :00 – 6 :20	6 :10 – 6 :30
Number of aircraft per original IAF	MOPAR	7	8
	LORNI	2	5
	OKIPA	1	1
Total number of aircraft		10	14
Wake-turbulence category mix		H : 70% M : 30%	H : 50% M : 50%

Next, two problem characteristics are explored : the width of time windows at the IAF and the uncertainty amplitude.

Time windows at the IAF

The width of the time windows at the IAF may reflect different desired levels of flexibility and punctuality. Wide time windows offer more flexibility to optimize the sequence and the schedule at the IAF, whereas narrow time windows yield more punctuality. From an operational perspective, if an aircraft with a ground speed of 450 kts increases its speed by 3%, it will only save 1 minute with respect to its planned time over a distance of 300 NM. Therefore, in our study, we limit the acceptable time advance with respect to the IAF planned times to 1 minute. However, different levels of acceptable delays can be defined. For example, in [3], delays are allowed to reach one hour, while time advances are limited to 1 minute. Following the XMAN concept (a first operational step towards E-AMAN implementation), 5 minutes of delay can be achieved in the en-route and descent phases using only speed reduction over 300 NM. This defines a narrow IAF time window. For wide IAF time windows, we assume 10 minutes of feasible additional delay using path stretching. To summarize, in our study, the tested time windows are : [-1 min, +5 min] (narrow) and [-1 min, +15 min] (wide).

Uncertainty

Following the literature ([75, 55]), deviations of the actual times with respect to the target times at the IAF are assumed to follow a normal distribution with mean 0 and some standard deviation σ. To assess the impact of uncertainty, we test two values for σ : 30 seconds (small) and 60 seconds (large).

Other problem characteristics relevant to the second stage (time windows for landing and unconstrained flight time from the IAF to the runway threshold) are kept constant. For every aircraft, we opt for [-1 min, +19 min] time windows for landing, and 11 minutes for the unconstrained flight time from the IAF to the runway threshold (regardless of the aircraft type). This time window is constructed based on the minimum flight time observed in our data (10 minutes) and a maximum flight time of 30 minutes, where the extra 20 minutes can be spent in a holding stack for example. Assuming one minute as the longest time advance within the terminal area, we set the unconstrained flight time to 11 minutes.

3.4.2 Optimization parameters

Two main optimization parameters are studied : the weighting parameter in the first-stage objective function, λ, and the number of second-stage scenarios, $n_{\mathcal{S}}$.

First-stage objective-function weight

In addition to minimizing the landing sequence length, different weights λ in the first-stage objective function are tested in order to evaluate the effect of minimizing the sum of target times at the IAF. Two weighting parameters values are defined ($\lambda = 0$, and $\lambda = 0.01$). With $\lambda = 0$, only the landing sequence length is minimized. With $\lambda = 0.01$, a compromise is considered between the landing sequence length and the sum of target times at the IAF, such that both quantities have comparable amplitudes.

Number of second-stage scenarios

Since we follow an exploratory approach that successively increases the number of scenarios, we solve SAA problems with $n_{\mathcal{S}} = 10$, 50, 100, 200, 500 and 1000.

3.4.3 Results for instance 1

Instance 1 is solved with all the 48 possible combinations (6 different numbers of scenarios, 2 IAF time-window widths, 2 uncertainty amplitudes, and 2 values for the weight λ). For each setting combination, $n_{\mathcal{R}} = 10$ replications are performed. Far each replication, the time limit in CPLEX solver is set to one hour. Although this time limit is inappropriate for real-time implementation, it was selected for exploration and study purposes as explained in Section 3.3.

The main results for instance 1 are shown on Tables 3.3 to 3.6. "CPU" stands for CPLEX solving time expressed in seconds, averaged on all the replications. When the time limit is reached for all replications, "Tilim" is indicated instead of the exact time value. "Status" tells whether CPLEX proved the optimality of the feasible solutions found. Note that feasible solutions are found for all replications and under all settings. When solutions for all replications are proved optimal by CPLEX then "Opt." is reported, while "r Opt." or "Feas." are reported if only r ($1 \leq r < n_{\mathcal{R}}$) or no solutions are proved optimal, respectively. "Gap" stands for the average (over all replications) percentage error of the best bound with respect to the value of the best feasible solutions returned by CPLEX. Here, $\bar{v}$ stands for the average objective value over the replications solved to optimality, noted $\bar{v}\,(n_{\mathcal{S}}, n_{\mathcal{R}})$ in Section 3.3. For each computed value of $\bar{v}\,(n_{\mathcal{S}}, n_{\mathcal{R}})$, a 95%-confidence interval is computed, thanks to the central limit theorem, its radius is noted "$I_{95\%}$". In our context, a 95%-confidence interval indicates that we are 95% sure that the true value of $\mathbb{E}\left[\hat{v}\,(\mathcal{S})\right]$ lies within this interval. "Validation" stands for the out-of-sample validation score, introduced in Section 3.3, averaged on all the replications. The validation set is made of 10,000 scenarios. Figures 3.2 to 3.5 plot $\bar{v}$, "Validation" and CPU as functions of $n_{\mathcal{S}}$ from Tables 3.3 and 3.5. Box-plots around $\bar{v}$ and validation scores are also shown in order to give some insight on the distribution of replication-specific values.

Effect of the number of scenarios, $n_{\mathcal{S}}$

As expected, the solving time increases rapidly with $n_{\mathcal{S}}$. In 18 out of 480 runs, CPLEX is not able to prove optimality within 1 hour. These runs correspond to tests on instance 1 with wide IAF time windows, under large uncertainty and $n_{\mathcal{S}} = 1000$ scenarios. Nevertheless, we observe that, in 6 out of 8 cases, we can solve replications with up to $n_{\mathcal{S}} = 500$ scenarios in less than 3 minutes, on average. In the two remaining cases (corresponding to instance 1 with wide IAF time windows, under large uncertainty), we can solve replications with up to $n_{\mathcal{S}} = 200$ scenarios in less than 4 minutes. We observe that as the number of scenarios increases, average objective-function values, $\bar{v}$, clearly increase while the spread of objective-function values around the average decreases. These two facts correctly illustrate the behavior of SAA problems' objective-function values when increasing the number of scenarios. On the other hand, as expected, average validation scores decrease with the number of scenarios. This reveals that increasing the number of scenarios leads to better-quality solutions. Remark that, for any given number of scenarios $n_{\mathcal{S}}$, the average validation score is greater than the average objective-function value. This demonstrates that the value of an optimal solution obtained with a limited number of scenarios is often underestimated.

TABLE 3.3 – Results for instance 1 with narrow IAF time windows and $\lambda = 0$

	TW^I	Narrow	
$n_{\mathcal{S}}$	σ	Small	Large
10	CPU (s)	0.3	0.5
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	934.8 ± 8.0	1028.3 ± 19.3
	Validation	950.8	1060.9
50	CPU (s)	1.3	2.2
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	938.9 ± 3.1	1034.9 ± 6.6
	Validation	944.2	1046.4
100	CPU (s)	3.2	6.4
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	940.4 ± 2.2	1040.3 ± 4.8
	Validation	943.8	1045.6
200	CPU (s)	9.6	18.5
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	940.3 ± 2.0	1039.6 ± 4.8
	Validation	942.6	1045.1
500	CPU (s)	75.1	144.3
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	941.6 ± 0.9	1042.8 ± 1.9
	Validation	942.0	1044.1
1000	CPU (s)	358.6	812.0
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	941.5 ± 0.8	1042.8 ± 1.8
	Validation	941.9	1043.7

For any given test setting, we do not observe any change of the landing sequence length when increasing the number of scenarios, while slight modifications of IAF target times occur. However, a significant increase is observed in terms of second-stage cost (as the number of scenarios increases). We conclude that, under our test settings, increasing the number of scenarios helps estimating more accurately the second-stage cost, and eventually adjusting IAF target times.

Effect of time-window width at the IAF

For both values of the weighting parameter λ, it is clear that the problem with narrow time windows is easier to solve to optimality than with wide time windows. With relatively wide time windows, too much flexibility is left to the solver to find an optimal solution. More precisely, the maximum number of positions to which aircraft can be shifted grows with the width of the time windows. The additional sequences offered by wide time windows may achieve better values of the objective function, as it is the case when comparing Tables 3.3 and 3.5 (both with $\lambda = 0$) for example. In fact, the minimum sequence length (not shown in the tables) found with narrow time windows (for any values for the remaining test settings) is 887 seconds, while with wide time windows (for any values for the remaining test settings), optimal sequences have a length of 826 seconds. This can be explained by looking closer at the first three aircraft in instance 1, where the first and the third aircraft belong to the heavy turbulence category, while the

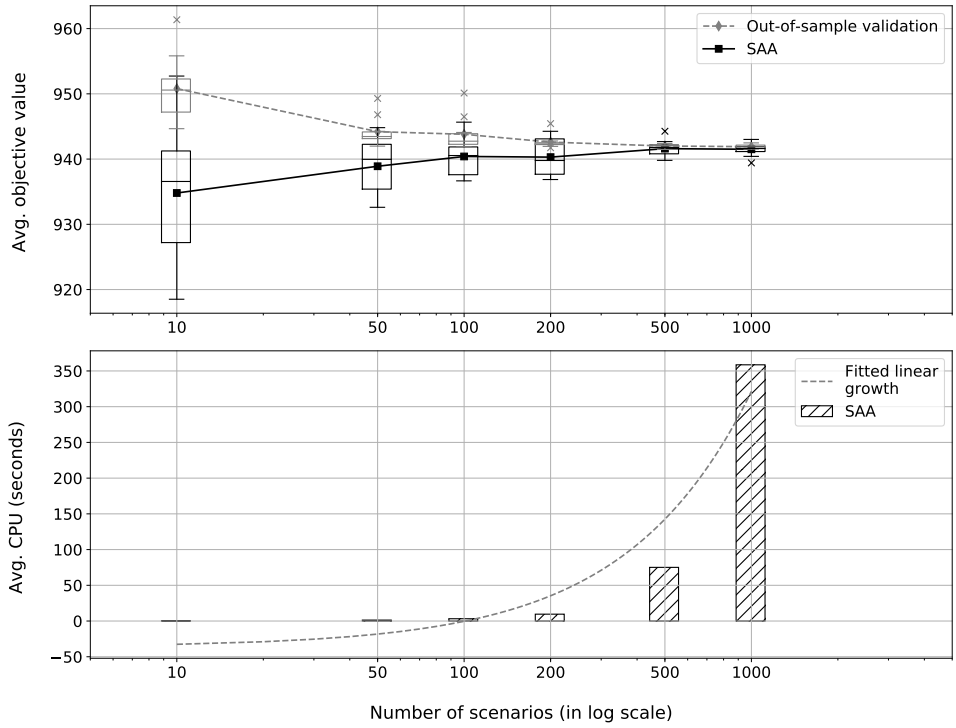


FIGURE 3.2 – Average objective-function values and validation scores (upper figure) and average CPU times (lower figure) for instance 1 with narrow IAF time windows, $\lambda = 0$ and small uncertainty.

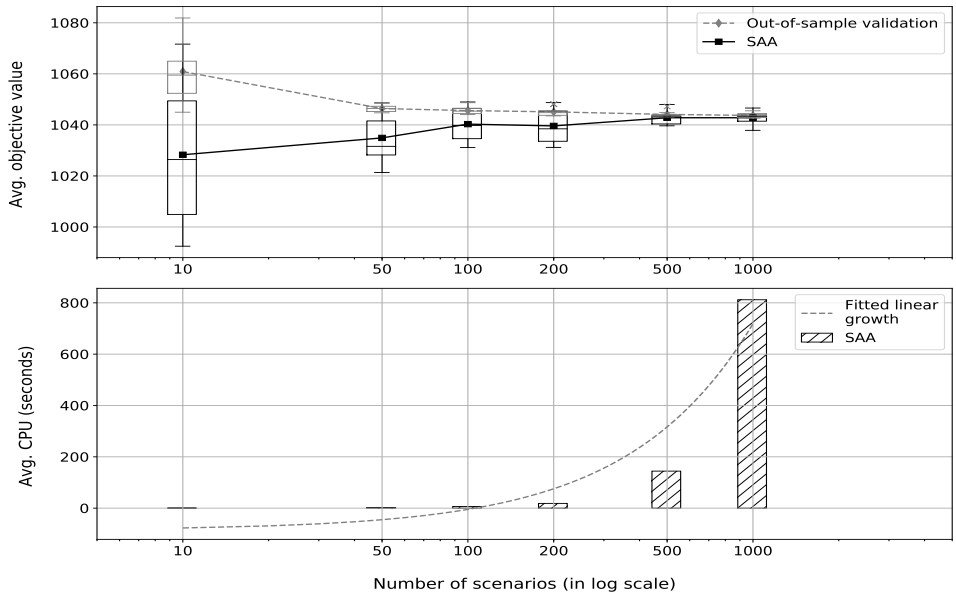


FIGURE 3.3 – Average objective-function values and validation scores (upper figure) and average CPU times (lower figure) for instance 1 with narrow IAF time windows, $\lambda = 0$ and **large** uncertainty.

second aircraft is a medium-turbulence jet. Due to narrow time windows, the first two aircraft cannot be shifted, which results in the partial sequence "H-M-H" whose length is 217 seconds. With wide time windows, the medium-turbulence-category aircraft can be shifted to the first position so that the partial sequence becomes "M-H-H", whose length

TABLE 3.4 – Results for instance 1 with narrow IAF time windows and $\lambda = 0.01$

	TW^I	Narrow	
$n_{\mathcal{S}}$	σ	Small	Large
10	CPU (s)	0.3	0.5
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	1730.9 ± 8.0	1823.5 ± 18.5
	Validation	1745.9	1852.7
50	CPU (s)	1.5	2.5
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	1733.5 ± 3.2	1829.6 ± 6.8
	Validation	1737.9	1841.1
100	CPU (s)	3.8	6.8
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	1735.2 ± 2.2	1835.2 ± 4.8
	Validation	1737.7	1840.5
200	CPU (s)	11.1	20.7
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	1734.8 ± 2.0	1834.4 ± 4.8
	Validation	1737.1	1839.9
500	CPU (s)	80.6	159.3
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	1736.2 ± 0.9	1837.5 ± 1.9
	Validation	1736.5	1838.6
1000	CPU (s)	372.3	837.4
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	1736.0 ± 0.7	1837.6 ± 1.8
	Validation	1736.4	1838.3

is only 156 seconds.

Better values of the objective function come at the expense of relatively longer solving times, especially under large uncertainty. For example, while with narrow time windows and under large uncertainty SAA problems with $n_{\mathcal{S}} = 500$ scenarios can be solved in less than 3 minutes on average, more than 17 minutes on average are needed to solve the same problems with wide time windows. On the other hand, in this example, using wide time windows improves the average objective-function value by 13%, and more precisely the expected landing sequence length is shortened by 61 seconds, compared to the case with narrow time windows.

In terms of second-stage cost, for $n_{\mathcal{S}} = 1000$ scenarios, we remark that, with wide IAF time windows, the average second-stage cost is decreased when compared to the test with narrow IAF time windows. In fact, wide IAF time windows allow more spaced IAF target times, which help absorbing the effect of uncertainty when revealed.

Finally, in our context, narrow time windows may be preferred to wide ones because they are likely to boost flight on-time performance and to reduce fuel consumption.

Effect of uncertainty amplitude

Throughout all the results, as uncertainty gets larger, more time is needed to solve the problem. Concerning runs with narrow time windows and different numbers of scenarios $n_{\mathcal{S}}$, the average CPU time may increase up to 2 times when the uncertainty standard

TABLE 3.5 – Results for instance 1 with wide IAF time windows and $\lambda = 0$

	TW^I	Wide	
$n_{\mathcal{S}}$	σ	Small	Large
10	CPU (s)	0.3	1.9
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	826.3 ± 0.7	876.8 ± 12.4
	Validation	840.2	929.7
50	CPU (s)	3.1	20.8
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	829.9 ± 1.1	894.2 ± 5.0
	Validation	834.3	907.8
100	CPU (s)	7.5	52.5
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	830.8 ± 0.8	897.7 ± 2.9
	Validation	833.7	906.9
200	CPU (s)	23.1	182.6
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	831.6 ± 0.6	899.4 ± 2.9
	Validation	833.4	906.3
500	CPU (s)	122.3	1023.2
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	832.3 ± 0.2	902.3 ± 1.4
	Validation	833.1	904.9
1000	CPU (s)	472.7	3557.9
	Status (Gap)	Opt. (0.0%)	2 Opt. (2.9%)
	$\bar{v} \pm I_{95\%}$	832.3 ± 0.1	902.2 ± 0.8
	Validation	833.0	904.7

deviation is doubled. The increase factor is more important when we switch from narrow to wide time windows. For example, with wide time windows and $n_{\mathcal{S}} = 500$, regardless of the value of λ, average CPU times increase by more than 8 times from small to large uncertainty.

Also, we observe that the gap between $\bar{v}$ and the validation score, called "validation" gap, decreases less rapidly under large uncertainty than under small uncertainty, as the number of scenarios increases. Consequently, we may conclude that as the uncertainty increases a larger number of scenarios is needed to correctly represent the original problem. Since computation times increase as uncertainty and the number of scenarios increase, real-time implementation of a stochastic programming approach based on scenario sampling may be very challenging.

Tested values of uncertainty standard deviation reveal no effect of uncertainty amplitude on the landing sequence length. However, larger uncertainty amplitudes are worthwhile to be tested. On the other hand, second-stage cost significantly increases as uncertainty increases. Figure 3.6 shows the effect of uncertainty amplitude and IAF time windows on the average second-stage cost, with $\lambda = 0$, $n_{\mathcal{S}} = 1000$ scenarios, and both narrow and large IAF time windows.

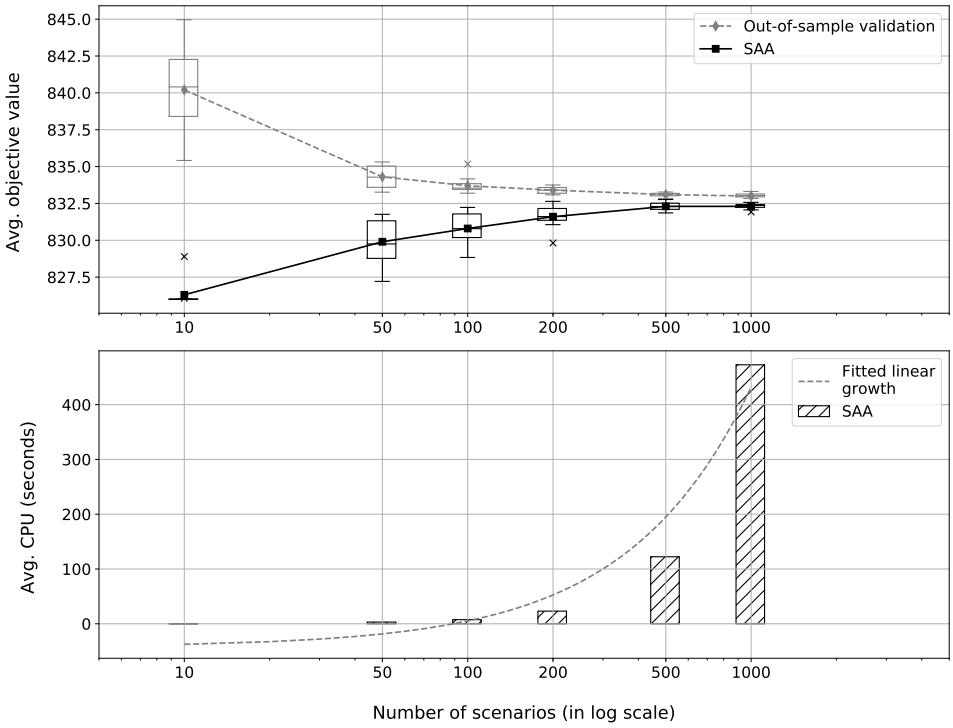


FIGURE 3.4 – Average objective-function values and validation scores (upper figure) and average CPU times (lower figure) for instance 1 with wide IAF time windows, $\lambda = 0$, small uncertainty.

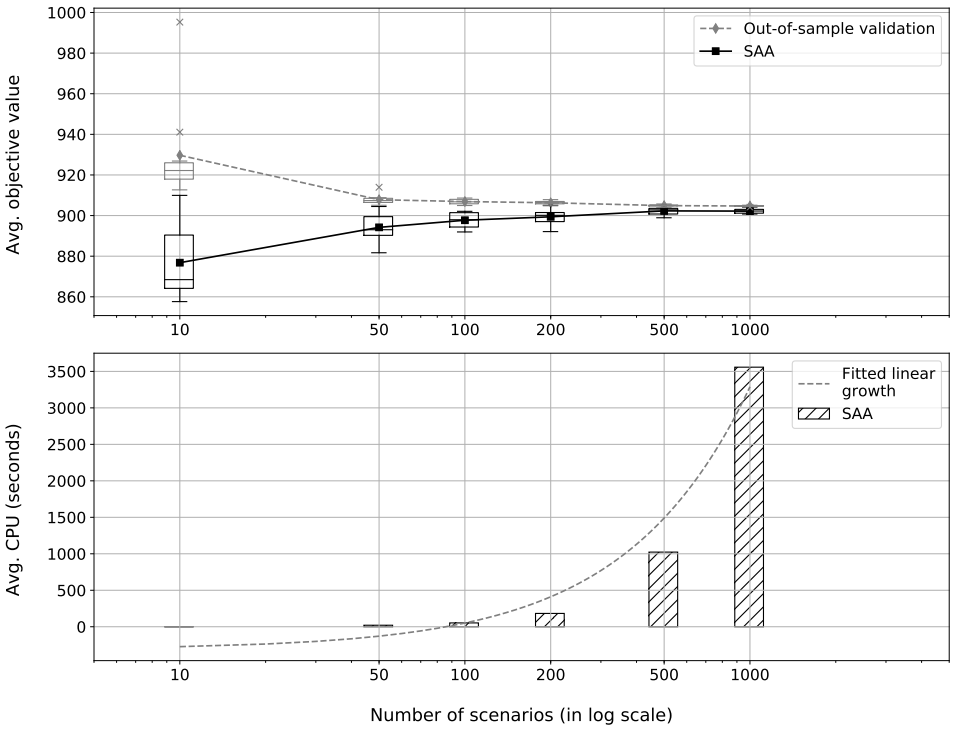


FIGURE 3.5 – Average objective-function values and validation scores (upper figure) and average CPU times (lower figure) for instance 1 with wide IAF time windows, $\lambda = 0$, large uncertainty.

TABLE 3.6 – Results for instance 1 with wide IAF time windows and $\lambda = 0.01$

	TW^I	Wide	
$n_{\mathcal{S}}$	σ	Small	Large
10	CPU (s)	0.7	2.6
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	1637.4 ± 2.4	1691.6 ± 12.1
	Validation	1657.6	1743.4
50	CPU (s)	5.0	23.9
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	1643.9 ± 1.3	1709.1 ± 5.2
	Validation	1648.9	1723.2
100	CPU (s)	13.8	65.3
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	1645.0 ± 1.1	1712.5 ± 2.8
	Validation	1647.9	1721.9
200	CPU (s)	33.9	217.7
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	1646.0 ± 0.9	1714.3 ± 2.9
	Validation	1647.8	1720.9
500	CPU (s)	164.9	1351.4
	Status (Gap)	Opt. (0.0%)	Opt. (0.0%)
	$\bar{v} \pm I_{95\%}$	1646.5 ± 0.3	1717.1 ± 1.5
	Validation	1647.3	1719.8
1000	CPU (s)	661.6	Tilim
	Status (Gap)	Opt. (0.0%)	Feas. (2.8%)
	$\bar{v} \pm I_{95\%}$	1646.7 ± 0.2	1717.2 ± 0.9
	Validation	1647.3	1719.5

Effect of minimizing the sum of target times at the IAF in the first stage

We observe that, when the weighted sum of target times at the IAF is also minimized in the first stage, average solving times slightly increase. The increase factor is greater with wide time windows than with narrow ones. For example, with wide time windows under small uncertainty and for $n_{\mathcal{S}} = 1000$ scenarios, the average solving time increases by 40% when increasing λ from 0 to 0.01. Figure 3.7 summarizes the effect of uncertainty amplitude, IAF time-window width, and λ on the average CPU time. In terms of objective-function values, large differences are obvious between results with $\lambda = 0$ and those with $\lambda = 0.01$.

As expected, since the objective-function value with $\lambda = 0.01$ is increased by the term $0.01 \times \sum_{i \in \mathcal{A}} x_i$ (called *weighted total completion time*), it is not directly comparable to the objective-function value with $\lambda = 0$. However, we still can extract and compare separately the values of the three criteria : weighted total completion time, landing sequence length and second-stage cost, displayed on Figure 3.8 for narrow time windows at the IAF and small uncertainty. Although, the weighted total completion time is not included in the objective function with $\lambda = 0$, it was recomputed separately after the optimization and plotted on Figure 3.8. Figure 3.8 reveals that minimizing the sum of target times at the IAF in the first stage in addition to the landing sequence length has no effect on the obtained landing sequence length, whereas it decreases the weighted total completion time, as expected. However, this decrease comes at the expense of increasing the second-

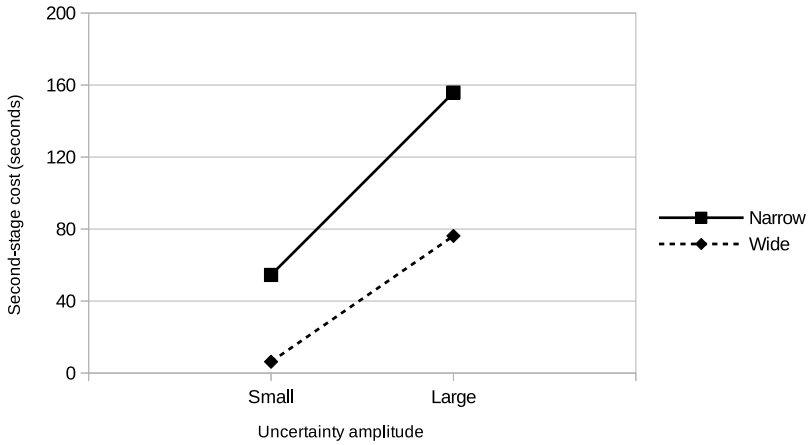


FIGURE 3.6 – Effect of uncertainty amplitude and IAF time-window width on the second-stage cost, for $\lambda = 0$ and $n_{\mathcal{S}} = 1000$.

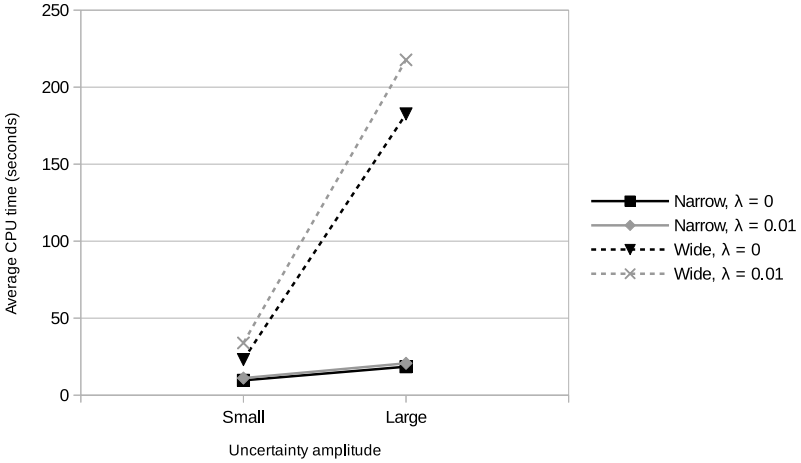


FIGURE 3.7 – Effect of uncertainty amplitude, IAF time-window width, and λ on the average CPU time for $n_{\mathcal{S}} = 200$.

stage cost. Therefore, to save computing time and decrease the second-stage terminal-area impact cost, we shall in the sequel focus only on minimizing the landing sequence length in the first stage ($\lambda = 0$).

3.4.4 Comparison of real-time solution methods : scenario-based versus replication-based approaches

Based on the numerical study reported in the previous subsection, we retain the following two settings : narrow time windows at the IAF and minimizing only the landing sequence length in the first stage ($\lambda = 0$). Uncertainty is expected to be smaller in a shorter time horizon, as shown in [67, 70]. This leads us to consider, in a real-time context, a small CPU time budget for optimization under small uncertainty, and a relatively larger time budget under large uncertainty. More precisely, we consider 1 and 5 minutes as time budgets for small and large uncertainty, respectively.

In this subsection, we compare scenario-based and replication-based approaches, in-

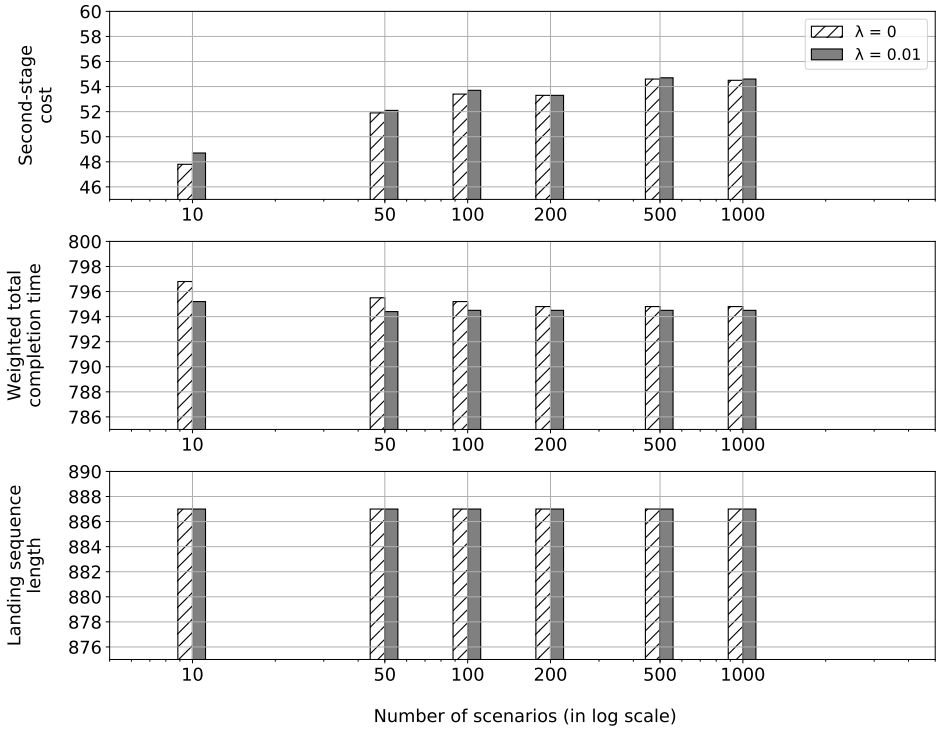


FIGURE 3.8 – Average landing sequence length, weighted total completion time and second-stage cost for instance 1, narrow IAF time windows, small uncertainty and $\lambda = 0$ and 0.01.

troduced in Section 3.3, to solve the SAA problem. Given the average CPLEX solving time reported in Subsection 3.4.3, we select $n_{\mathcal{S}} = 100$ scenarios for the replication-based approach under small uncertainty and $n_{\mathcal{S}} = 200$ scenarios under large uncertainty, while the number of replications is adjusted to fit the time budget. In the scenario-based approach, only one SAA problem is solved with a relatively large number of scenarios (400 and 600 scenarios under small and large uncertainty, respectively).

On the one hand, a scenario-based approach naturally returns a unique solution. On the other hand, at the end of the solving process using a replication-based approach, we are left with a pool of solutions. To select a unique solution from this pool, we may either directly choose the minimum-objective-function-value solution (called *min-Obj*) as in [14], or re-evaluate all distinct solutions from the pool on a validation set and choose the minimum-validation-score solution (called *min-Val*) as in [67]. Results of the replication-based approach for both small and large uncertainties are summarized in Tables 3.8 and

TABLE 3.7 – Scenario-based approach results : instance 1, narrow IAF time windows, $\lambda = 0$

σ	Small	Large
$n_{\mathcal{S}}$	400	600
$n_{\mathcal{R}}$	1	1
CPU (s)	47.7 (+ 12.8)	228.2 (+ 13.3)
$v^\star$	940.7	1040.3
Validation	941.7	1043.2

3.9. Results of the scenario-based approach for both small and large uncertainty are summarized in Table 3.7. In these Tables, "CPU" stands for the solving time from CPLEX expressed in seconds, while the added time between brackets stands for the total validation time (for one solution from the scenario-based approach and for all solutions from the replication-based approach). For each retained solution, $v^\star$ represents the objective-function value, while "Validation" refers to its out-of-sample validation score, computed over $10,000$ scenarios. Figures 3.9 and 3.10 plot validation scores and computing times (solving and validation) for replication-based solutions (*min-Obj* and *min-Val*) in terms of the number of replications, under small and large uncertainty respectively. The performance of the scenario-based solution is shown in black thick line.

Regardless the uncertainty amplitude and the number of replications, the *min-Obj* solution clearly performs worse than the *min-Val* and the scenario-based solutions, as expected. Under small uncertainty, the scenario-based approach can be applied with a large number of scenarios ($n_{\mathcal{S}} = 400$) within the time budget of 1 minute. The *min-Val* solution from the replication-based approach for $n_{\mathcal{R}} = 20$ replications outperforms the scenario-based solution. However, it can only be obtained with around 5 minutes computing time. Consequently, if the time budget is limited to 1 minute, the scenario-based approach is recommended. Similar observations can be made for the case under large uncertainty.

As a conclusion, regardless the uncertainty amplitude, a fast solution method should

TABLE 3.8 – Replication-based approach results : instance 1, narrow IAF time windows, $\lambda = 0$, small uncertainty

$n_{\mathcal{R}}$	solution	*min-Obj*	*min-Val*
5	CPU (s)	15.8 (+ 62.8)	
	$v^\star$	936.7	937.3
	Validation	942.6	942.3
10	CPU (s)	31.7 (+ 127.3)	
	$v^\star$	936.7	938.1
	Validation	942.6	941.9
20	CPU (s)	62.8 (+ 247.5)	
	$v^\star$	931.3	938.3
	Validation	942.5	941.5

TABLE 3.9 – Replication-based approach results : instance 1, narrow IAF time windows, $\lambda = 0$, large uncertainty

$n_{\mathcal{R}}$	solution	*min-Obj*	*min-Val*
5	CPU (s)	91.0 (+ 63.7)	
	$v^\star$	1032.7	1036.3
	Validation	1043.7	1043.6
10	CPU (s)	185.2 (+ 128.3)	
	$v^\star$	1031.2	1031.2
	Validation	1043.5	1043.5
15	CPU (s)	281.8 (+ 194.4)	
	$v^\star$	1029.1	1044.6
	Validation	1045.2	1043.1

FIGURE 3.9 – Solution characteristics from replication-based approach under small uncertainty

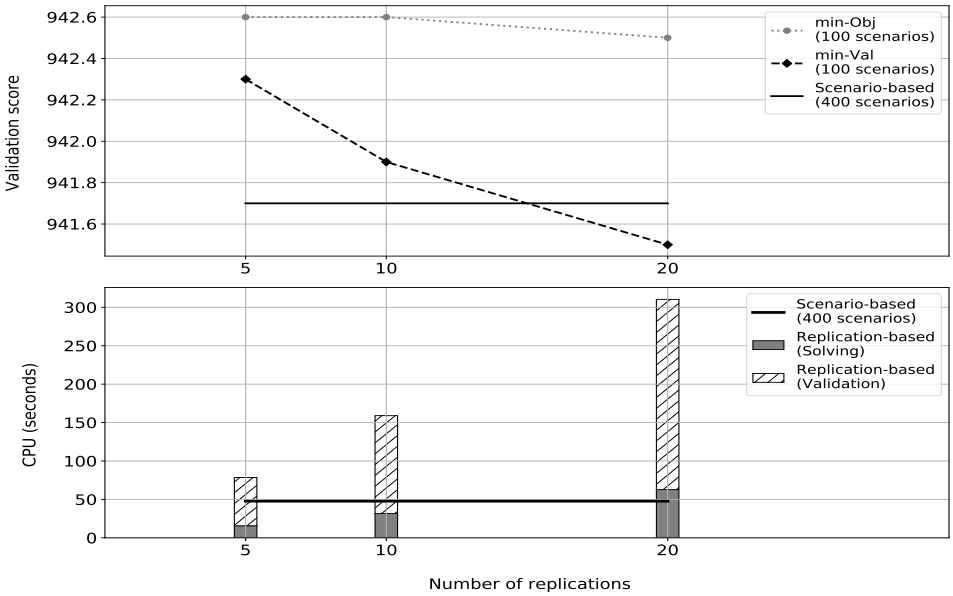
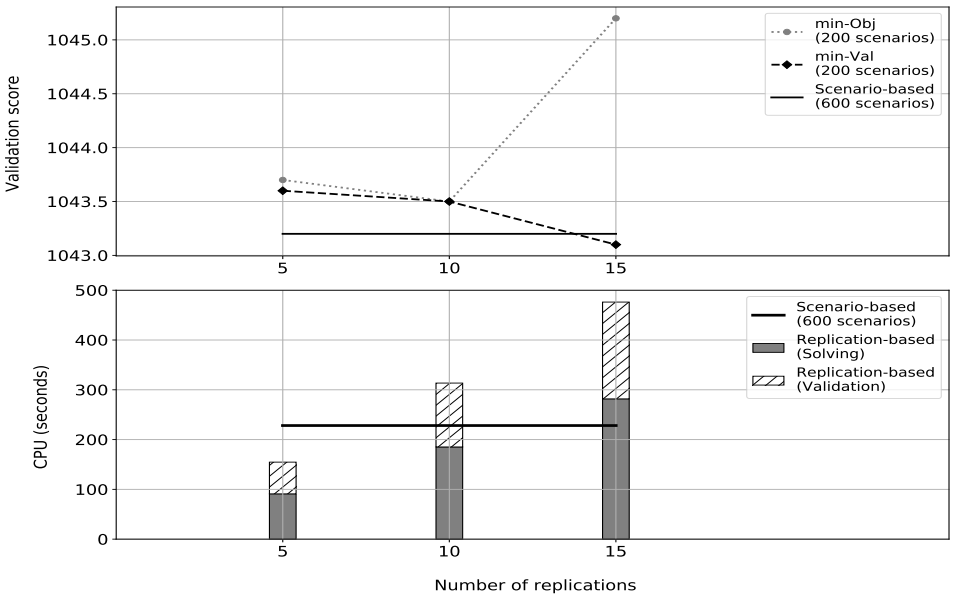


FIGURE 3.10 – Solution characteristics from replication-based approach under large uncertainty



be built on a scenario-based approach using a large enough number of scenarios, subject to the time budget. If a fast out-of-sample validation procedure is available, then any replication-based approach should follow the validation-score criterion to select a solution from the solution pool, while the minimum-objective-value-function criterion should be avoided.

3.4.5 Results for instance 2

As narrow time windows at the IAF and $\lambda = 0$ were identified as a suitable setting for instance 1 (n =10 aircraft), we keep them for our tests on instance 2 (n =14 aircraft). For each uncertainty amplitude, small and large, we solve a single SAA problem with 10 minutes as a time limit for CPLEX. We use 100 scenarios for small uncertainty and 200 scenarios for large uncertainty. Out-of-sample validation is performed using 10,000 scenarios. Results under small and large uncertainties, given in Table 3.10, show that instance 2 is much harder to solve than instance 1. Although CPLEX is unable to prove optimality within 10 minutes for 200 scenarios under large uncertainty, validation scores are not dramatically larger than the objective-function values. Nevertheless, an efficient solving algorithm is clearly needed to handle large numbers of aircraft.

TABLE 3.10 – Scenario-based approach results : instance 2, narrow IAF time windows, and $\lambda = 0$

σ	Small	Large
$n_{\mathcal{S}}$	100	200
CPU (s)	444.1	Tilim
Status (Gap)	Opt. (0.0%)	Feas. (19.9%)
$v^\star$	1292.3	1513.2
Validation	1297.6	1519.6

3.4.6 Effect on FCFS policy performance in the terminal area

In this subsection, we consider a complete sequencing-and-scheduling process of aircraft arrivals from the time they are captured (two to three hours before landing) to the time they land. We consider two sequencing-and-scheduling points : the IAF, and the runway threshold. Firstly, captured aircraft are sequenced and scheduled for the IAF according to some policy, called the *pre-IAF sequencing-and-scheduling policy*. Due to uncertainties, actual aircraft IAF times are different from the IAF target times. Subsequently, when aircraft actually arrive at the IAF, some other policy is applied in order to sequence and schedule approaching aircraft to the runway threshold. Here, we assume that ATCOs in the terminal area sequence aircraft for landing in the same order in which they actually pass over the IAF. This air traffic control policy will be referred to as *FCFS policy in the terminal area*. We aim at evaluating the effect of our retained solutions from fast solution methods, reported in Table 3.7, on the expected performance of the FCFS policy in the terminal area. For that purpose, we compare the expected performance of the FCFS policy in the terminal area in different upstream situations i.e, when different pre-IAF sequencing-and-scheduling policies are applied. The baseline upstream situation consists in scheduling aircraft at the IAF in a FCFS fashion according to their planned times at the IAF. The minimum time separation enforced over the IAF is simply $\underline{S}^I$ (72 seconds). This baseline pre-IAF sequencing-and-scheduling policy is called the *pre-IAF FCFS policy*. Enforcing a minimum separation between aircraft that is larger than operational requirements is a common technique to hedge against uncertainty [55]. When the pre-IAF FCFS policy is applied with an enlarged minimum separation over the IAF, we call it a pre-IAF *buffered* FCFS policy. Since the minimum separation over the IAF is commonly expressed in NM, we consider two values for the distance buffer, 1 NM and 2

NM, resulting in time buffers of 14 and 28 seconds respectively. The two resulting pre-IAF sequencing-and-scheduling policies are called pre-IAF 1-buffered FCFS policy and pre-IAF 2-buffered FCFS policy. These last two policies define the second and the third upstream situations respectively. The fourth upstream situation relies on our stochastic-optimization approach to pre-sequence and pre-schedule aircraft over the IAF. The four pre-IAF policies will be noted "FCFS-0", "FCFS-1", "FCFS-2", and "StochOpt" respectively. For each upstream situation, 10,000 scenarios are simulated and various performance measures of the FCFS policy in the terminal area, described below, are computed.

Performance measures :

- Average number of conflicts at the IAF, noted "IAF conflicts", computed as the average number of separation violations over the IAF between any pair of aircraft
- Average landing rate per hour, noted "landing rate"
- Average last landing time, noted "last landing"
- Average total time-to-lose in the terminal area, noted "TMA total time-to-lose", the time-to-lose for a single aircraft in the terminal area being computed as the (positive) deviation from the aircraft target landing time with respect to its unconstrained landing time
- Average maximum time-to-lose in the terminal area, noted "TMA max time-to-lose"

Tables 3.11 and 3.12 report values of these performance measures respectively for instance 1 and a compressed version of instance 1, under the four upstream situations. The modified version of instance 1 was obtained by compressing the planned IAF schedule of instance 1 by a factor two. Hence, the time span over the IAF is contracted from 6:00 - 6:20 AM to 6:00 - 6:10 AM, while the number of aircraft, $n = 10$, is conserved. We computed our solutions for the compressed version of instance 1 using $n_{\mathcal{S}} = 100$ scenarios for both small and large uncertainties. Solving times are respectively 59.1 and 71.6 seconds. The target sequence and IAF target times, for each pre-IAF policy, are illustrated in Figures 3.11 and 3.12 for the original instance 1 and its compressed version, respectively. In each of the two figures, "StochOpt-30" and "StochOpt-60" refer to our stochastic programming policy under small and large uncertainty respectively. IAF target times of a given aircraft that has the same position in the sequence across the different pre-IAF policies are linked with a continuous line, while dashed lines are used for aircraft whose position changes at least under one pre-IAF policy.

In moderate-density traffic, as in instance 1 (Table 3.11), solutions from our stochastic-optimization approach decrease the expected number of conflicts over the IAF, *e.g*, down to -70% under small uncertainty, and the time-to-lose within the terminal area, *e.g*, down to -86% in terms of total time-to-lose in the terminal area under small uncertainty, compared to the pre-IAF FCFS policy. On the other hand, average landing rates and last landing times using our approach are slightly worse than the pre-IAF FCFS policy. This may be due to the fact that our stochastically-optimized schedules at the IAF are too sparse (see Figure 3.11). With regard to the two pre-IAF buffered FCFS policies with 1 NM or 2 NM, they can be an interesting compromise in moderate traffic since they perform close to our approach while impacting less the landing rate and the average last landing time.

In high-density traffic, as in the compressed instance 1 (Table 3.12), our stochastic sequencing-and-scheduling policy outperforms the pre-IAF FCFS policy almost in all

TABLE 3.11 – FCFS performance in the terminal area : instance 1 with narrow IAF time windows

uncertainty	pre-IAF policy	IAF conflicts	landing rate	last landing	TMA time-to-lose total	TMA time-to-lose max
Small	FCFS - 0	3.1	**26.9**	**06 :33 :29**	8 min 23 s	2 min 35 s
	FCFS - 1	2.2	26.8	06 :33 :35	6 min 01 s	2 min 01 s
	FCFS - 2	1.5	26.6	06 :33 :43	4 min 09 s	1 min 35 s
	StochOpt	**0.9**	25.0	06 :34 :13	**1 min 08 s**	**0 min 41 s**
Large	FCFS - 0	3.6	**26.9**	**06 :33 :42**	9 min 02 s	2 min 48 s
	FCFS - 1	3.0	26.6	06 :33 :55	7 min 25 s	2 min 24 s
	FCFS - 2	2.5	26.3	06 :34 :11	6 min 11 s	2 min 07 s
	StochOpt	**1.8**	24.7	06 :34 :39	**2 min 47 s**	**1 min 22 s**

TABLE 3.12 – FCFS performance in the terminal area : compressed instance 1 with narrow IAF time windows

uncertainty	pre-IAF policy	IAF conflicts	landing rate	last landing	TMA time-to-lose total	TMA time-to-lose max
Small	FCFS - 0	3.9	34.8	06 :28 :24	16 min 09 s	4 min 20 s
	FCFS - 1	2.7	34.6	06 :28 :32	10 min 24 s	2 min 51 s
	FCFS - 2	**1.8**	33.9	06 :28 :52	5 min 35 s	1 min 44 s
	StochOpt	2.9	**37.5**	**06 :26 :10**	**5 min 10 s**	**1 min 24 s**
Large	FCFS - 0	5.0	34.6	06 :28 :42	18 min 05 s	4 min 27 s
	FCFS - 1	3.9	34.1	06 :28 :57	12 min 51 s	3 min 14 s
	FCFS - 2	3.2	33.2	06 :29 :25	8 min 31 s	2 min 23 s
	StochOpt	**2.8**	**37.5**	**06 :26 :10**	**5 min 12 s**	**1 min 24 s**

measures. In terms of expected number of IAF conflicts, under large uncertainty, 44% less separation violations are likely to occur on average compared to the unbuffered pre-IAF FCFS policy. The time-to-lose inside the terminal area dramatically decreases as well, *e.g*, down to -71% in terms of total time-to-lose in the terminal area under large uncertainty. Also, the average last landing time is earlier by more than 2 minutes, which increases the average landing rate, *e.g*, by 7.7% under large uncertainty. In high-density traffic, pre-IAF buffered FCFS policies may efficiently decrease the expected number of IAF conflicts and the time-to-lose inside the terminal area. However, average landing rates slightly decline, most likely because of too sparse IAF target times (due to the enlarged IAF separation requirements).

To study further the effect of uncertainty, we test our approach with a standard deviation of $\sigma = 200$ seconds. Remark that under this "very large" uncertainty, more than 93% of random IAF time deviations fall within ± 5 minutes and more than 99% within ± 10 minutes. Results in moderate-density traffic (instance 1) confirm the trend observed under small and large uncertainties. In high-density traffic (compressed instance 1), average landing rates are maintained compared to the unbuffered pre-IAF FCFS policy, while significant improvements are made on all of the remaining performance measures.

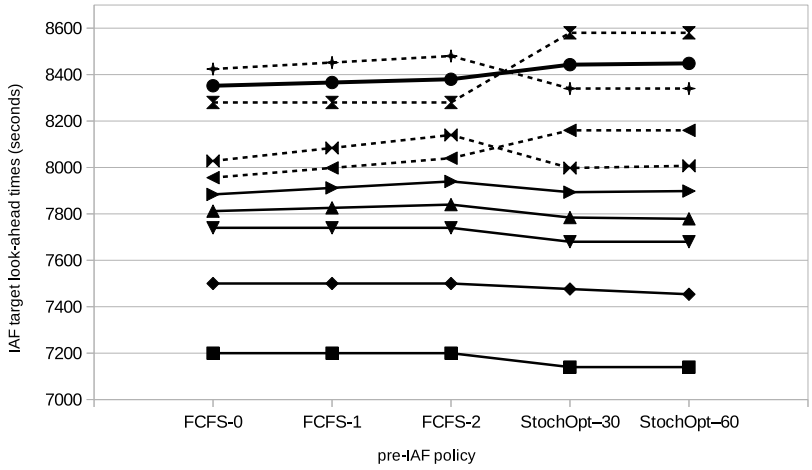


FIGURE 3.11 – IAF target times and sequences for instance 1 with different pre-IAF policies.

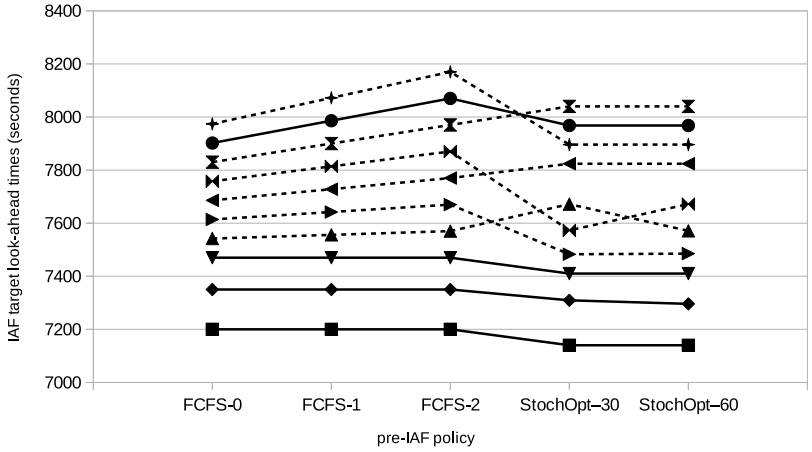


FIGURE 3.12 – IAF target times and sequences for the compressed version of instance 1 with different pre-IAF policies.

Finally, recall that a single tour in a holding stack is usually flown in 4 minutes. Measures of time-to-lose in the terminal area, especially in high-density traffic, show that our stochastic-optimization approach may avoid resorting to such holding stacks. Under a FCFS policy in the terminal area, we may conclude that our optimization approach over the IAF successfully transforms a *circular holding* (holding patterns at the entry of the terminal area) into a linear holding applied when aircraft are still a few hours away from landing, while increasing the landing rate, as expected from an efficient E-AMAN.

3.5 Conclusions

In this paper, we have presented a computational study on the problem of sequencing and scheduling arrivals over a single IAF under uncertain times at the IAF. Such a problem is relevant due to the foreseen extension of AMAN operational horizon up to a few hours before landing. We relied on a two-stage stochastic programming approach exploiting CPLEX solver. The SAA method was used to make the problem tractable and to find

satisfying approximate solutions. The effect of different problem characteristics (narrow vs. wide time windows at the IAF, small vs. large uncertainty) as well as different optimization parameters (first-stage objective function, number of second-stage scenarios) were analyzed in order to evaluate the viability of our approach. Scenario-based and replication-based approaches were tested and compared as potential solution methods for real-time implementation. Realistic instances involving 10 and 14 arrivals on CDG were used as a case study. For 10 aircraft with narrow time windows at the IAF, our approach returns good-quality solutions in a short solving time (less than 1 minute for small uncertainty and less than 5 minutes for large uncertainty). Simulation-based validation experiments show that our retained solutions can decrease the number of expected conflicts over the IAF by more than 70% and the total time-to-lose inside the terminal area by 86% over a FCFS policy, at the expense of a slight decrease of the landing rate in moderate-density traffic under small uncertainty. In high-density traffic, besides alleviating traffic complexity near the terminal area, our solutions enhance the expected landing rate by more than 7% under large uncertainty.

Chapitre 4

Modélisation par programmation stochastique à deux étapes – *Two-stage stochastic programming modeling*

Note for English readers. This chapter corresponds to the article entitled "*Two-stage stochastic mixed-integer programming with chance constraints for extended aircraft arrival management*", accepted for publication in *Transportation Science* (on March 9, 2020).

Résumé du chapitre

Ce chapitre reprend l'article "*Two-stage stochastic mixed-integer programming with chance constraints for extended aircraft arrival management*" accepté pour publication dans la revue *Transportation Science* (le 9 mars 2020). Il est rédigé en anglais.

Dans ce travail, nous modélisons le problème d'ordonnancement des arrivées d'avions avec horizon opérationnel étendu, comme un problème d'optimisation stochastique à deux étapes.

Dans la première étape, un ensemble d'avions à 2–3 heures d'un même aéroport de destination est considéré. Il s'agit d'ordonnancer ces avions sur un même IAF en vue de minimiser la longueur de la séquence cible d'atterrissage, calculée comme la somme des minima de séparation au seuil de piste entre les paires d'avions successifs. Les décisions de la première étape sont la séquence cible et les heures cibles de passage à l'IAF. Les contraintes opérationnelles sont le minimum de séparation pour toute paire d'avions à l'IAF, estimée à 72 secondes pour toute paire d'avions, et les fenêtres de temps à l'IAF. L'incertitude concerne les heures *effectives* d'arrivée des avions à l'IAF, supposées dévier des heures cibles par des quantités aléatoires suivant des lois de probabilités connues.

La deuxième étape commence au moment où l'ensemble des avions est suffisamment proche de l'IAF, de sorte que toutes les heures effectives de passage à l'IAF puissent être connues/estimées avec certitude. Dans la deuxième étape, après révélation de l'incertitude, il est question d'ordonnancer les avions, très proches à l'IAF, au seuil d'une même piste d'atterrissage. Les décisions sont des heures cibles d'atterrissage, tandis que la séquence *cible* d'atterrissage n'est pas re-calculée car elle a été déjà déterminée en première étape.

L'objectif de la deuxième étape est de minimiser la somme des coûts de déviation de l'heure cible d'atterrissage de chaque avion par rapport à son heure *non-contrainte*. Un tel coût est supposé convexe linéaire par morceaux, pour chaque avion. L'heure non-contrainte d'atterrissage correspond à la situation où l'avion est considéré tout seul dans la zone terminale, pouvant voler de l'IAF jusqu'au seuil de piste en suivant sa trajectoire optimale, et sans intervention supplémentaire des contôleurs aériens. Notons que cette heure non-contrainte n'est révélée qu'après connaissance de l'heure effective de passage de l'avion à l'IAF. Toute déviation par rapport à l'heure non-contrainte d'atterrissage inflige un coût, soit de retard, soit d'avance et peut correspondre au point de vue des contrôleurs aériens (une augmentation relative de la charge de travail) ou des compagnies aériennes (consommation de carburant, etc). Les contraintes opérationnelles de la deuxième étape sont les séparations minimales au seuil de piste (dépendant de la paire d'avions concernés) et les fenêtres de temps à l'atterrissage.

Ce problème d'ordonnancement des arrivées d'avions sous incertitude est formulé comme un programme stochastique à deux étapes, où le problème de première étape est linéaire à variables mixtes et enrichi par des contraintes en probabilités, et où le problème de deuxième étape est un programme linéaire. Les contraintes en probabilités en première étape servent à mitiger le risque de violation de la séparation minimale entre les heures effectives à l'IAF, suite à la révélation de l'incertitude. Ce risque est limité à un niveau de tolérance prédéterminé par le décideur pour toute paire d'avions.

Grâce à l'hypothèse d'indépendance et de distribution normale identique pour les variables aléatoires, les contraintes en probabilités peuvent être linéarisées en des contraintes de séparation avec un minimum de séparation dépendant du niveau de tolérance. Pour des niveaux de tolérance élevés, ces contraintes linéarisées peuvent remplacer les contraintes classiques de séparation à l'IAF. Les variables aléatoires étant de support infini, la méthode d'approximation par moyenne empirique (présentée dans le chapitre 1, Section 1.3.3) est utilisée. La formulation déterministe d'approximation, en résultant, s'apprête bien à une résolution par décomposition de Benders. Cette méthode de résolution est présentée dans le chapitre 1, Section 1.4. L'article propose une reformulation de Benders *partiellement agrégée*, et esquisse certaines techniques d'accélération.

Une étude numérique sur des instances réalistes de l'aéroport Paris Charles-De-Gaulle met en valeur l'intérêt de la programmation stochastique et des contraintes en probabilités, face à une approche déterministe annulant les déviations par rapport aux heures cibles à l'IAF. Une comparaison entre différentes méthodes de résolution se basant sur la décomposition de Benders automatique de CPLEX montre l'intérêt d'agréger partiellement les sous-problèmes de Benders.

Parmi les perspectives de cette étude, nous envisageons de traiter le cas opérationnel plus réaliste de plusieurs IAF. Aussi, une implémentation de la décomposition de Benders exploitant la structure du sous-problème pourrait donner des résultats encore plus performants, en comparaison à la décomposition de Benders automatique de CPLEX.

Table of content

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.

4.1 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 22 and, later, Bennell et al. 8). 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 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 4.1, and are expressed as inter-aircraft distances in nautical miles (NM ; 1 NM = 1.852 m) as in Table 4.2. 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 4.1 – 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 4.2 – 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. 21

On the operational side, since the early 90's in the USA and Europe, air traffic controllers (ATCOs), 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 ATCOs' input criteria [31, 58, 72, 36]. 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 [23, 70]. 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 ATCOs' 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 [70]. 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 [55, 70]. 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. 70. 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 ATCOs who 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 [22, 5, 3, 8, 30], while uncertainty has less often been taken into account. Pioneer studies of the ALP considering uncertainty were conducted by [55, 17], and [49] who basically added probabilistic considerations to the deterministic ALP. Stochastic optimization models, including two-stage and multi-stage models, were applied by [67, 65, 66], and [14] to address a variant of the ALP under uncertainty that considers departures and surface operations on the airport. Recently, [38, 42], and [37] 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 [42] investigates the *pre-tactical ALP* which starts several hours before the planned landing times. In Kapolke et al. 42, 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 [11] : "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 73, Laporte and Louveaux 48). 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 [11, 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 4.2. In Section 4.3, we propose a two-stage stochastic model with recourse. Solution methods are proposed in Section 4.4. Results of numerical experiments are discussed in Section 4.5. Section 4.6 presents some conclusions and future research tracks.

4.2 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 4.3 shows final-approach separations converted to seconds, as used in CDG. Detail of such a conversion may be found in de Neufville et al. 21. 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.

TABLE 4.3 – 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

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 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, ATCOs 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 [8], this may also help fulfill fairness requirements among aircraft, similarly to the more classical *constraint position shifting (CPS)* approach (see Balakrishnan and Chandran 3 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 [8], "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 ATCOs 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 [46], 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.

4.3 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. 5), 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 4.2, 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 [9], 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* :

$$\min_{\delta,\, x} \quad \sum_{\substack{(i,j)\in\mathcal{A}^+\times\mathcal{A}^+ \\ i\neq j}} \delta_{ij} S_{ij} + \lambda\, \mathbb{E}_{\boldsymbol{\omega}}[Q(\delta, x, \omega)] \tag{4.1}$$

$$\text{s.t.} \quad \sum_{\substack{j\in\mathcal{A}^+ \\ j\neq i}} \delta_{ji} = 1 \qquad i \in \mathcal{A}^+ \tag{4.2}$$

$$\sum_{\substack{j\in\mathcal{A}^+ \\ j\neq i}} \delta_{ij} = 1 \qquad i \in \mathcal{A}^+ \tag{4.3}$$

$$x_j \geq x_i + \underline{S}^I - M^I_{ij}(1-\delta_{ij}) \qquad (i,j) \in \mathcal{A}\times\mathcal{A},\quad i\neq j \tag{4.4}$$

$$\mathbb{P}(x_j + \omega_j \geq x_i + \omega_i + \underline{S}^I - M^{I\alpha}_{ij}(1-\delta_{ij})) \geq \alpha \qquad (i,j) \in \mathcal{A}\times\mathcal{A},\ i\neq j \tag{4.5}$$

$$E^I_i \leq x_i \leq L^I_i \qquad i \in \mathcal{A} \tag{4.6}$$

$$\delta_{ij} \in \{0,1\} \qquad (i,j) \in \mathcal{A}^+\times\mathcal{A}^+,\quad i\neq j \tag{4.7}$$

where :

$$Q(\delta, x, \omega) = \min_{y} \quad f(x, \omega, y) \tag{4.8}$$

$$\text{s.t.} \quad y_j \geq y_i + S_{ij} - M_{ij}(1-\delta_{ij}) \qquad (i,j) \in \mathcal{A}\times\mathcal{A},\quad i\neq j \tag{4.9}$$

$$\underline{V_i} \leq y_i - (x_i + \omega_i) \leq \overline{V_i} \qquad i \in \mathcal{A} \tag{4.10}$$

The objective function (4.1) 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 (4.2) to (4.7). Given a scenario ω, the cost of the second-stage (so-called *recourse*) problem, $Q(\delta, x, \omega)$, is defined by (4.8) to (4.10). Big-M constants appearing in constraints (4.4), (4.5) and (4.9) will be further commented below.

First-stage model

Constraints (4.2), (4.3) and (4.4) are directly inspired from the classical ATSP formulation. Constraints (4.2) and (4.3) ensure that all aircraft in $\mathcal{A}^+$ are sequenced, which corresponds to visiting all cities in an ATSP. Constraints (4.4) 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 (4.5) 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 (4.5) can be expressed in a deterministic form analogous to the big-M separation constraints (4.4). This will be detailed in Subsection 4.3.2. 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 56). Constraints (4.6) are time-window constraints on target times over the IAF. Constraints (4.7) stipulate the binary nature of the δ_{ij} variables.

Without the probability constraints (4.5), the first-stage problem defined by the first-stage objective function and constraints (4.2) to (4.4), (4.6) and (4.7), 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 (4.4) play the role of MTZ constraints [57]. 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 (4.4) 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 (4.8) 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 4.3.1. Constraints (4.9) ensure final approach minimal time separation. Minimal and maximal flight times are enforced by constraints (4.10). 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 (4.9) can be computed as the lowest upper bound to $(y_i - y_j + S_{ij})$. Using constraints (4.9) and (4.10) and bound constraints (4.6) 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.

4.3.1 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

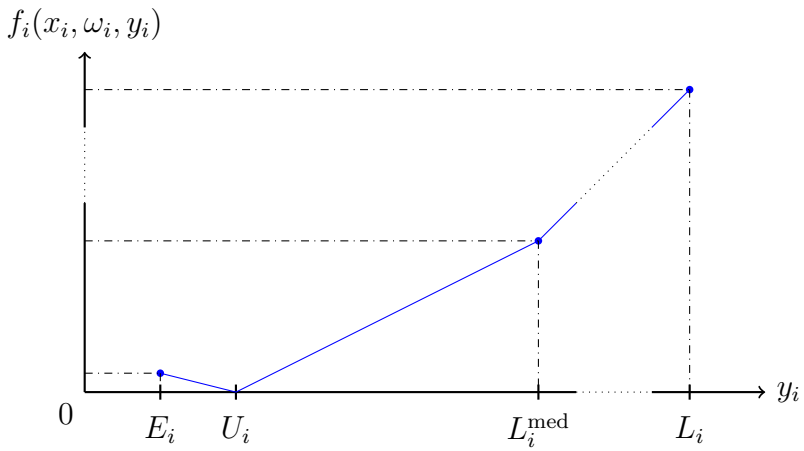


FIGURE 4.1 – Time-deviation impact cost function f_i of aircraft $i \in \mathcal{A}$.

cost function f_i for an aircraft $i \in \mathcal{A}$ (Figure 4.1) :

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

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 (4.8) 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 :

$$\min_{\substack{y, z^-, z^+ \\ z^{++}}} \sum_{i \in \mathcal{A}} \left(c_1 z_i^- + c_2 z_i^+ + c_3 z_i^{++}\right) \tag{4.11}$$

$$y_i - U_i = z_i^+ + z_i^{++} - z_i^- \qquad i \in \mathcal{A} \tag{4.12}$$

$$z_i^+ \leq L_i^{\text{med}} \qquad i \in \mathcal{A} \tag{4.13}$$

$$z_i^-, z_i^+, z_i^{++} \geq 0 \qquad i \in \mathcal{A} \tag{4.14}$$

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

The aim of this Subsection is twofold. Firstly, we show that the chance constraints (4.5) can be reformulated as deterministic linear constraints analogous to the big-M separation constraints (4.4), 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 :*

$$\mathbb{P}\left(x_j + \omega_j \geq x_i + \omega_i + \underline{S}^I - M_{ij}^{I\alpha}(1 - \delta_{ij})\right) \geq \alpha \tag{4.5$_{ij}$}$$

$$x_j \geq x_i + S^I(\alpha) - M_{ij}^{I\alpha}(1 - \delta_{ij}) \tag{4.17$_{ij}$}$$

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* (4.5_{ij}) *is equivalent to the deterministic constraint* (4.17_{ij}).

Démonstration. Proof of Lemma 1

Consider a couple $(i, j) \in \mathcal{A} \times \mathcal{A}$ such that $i \neq j$. Then, the constraint (4.5_{ij}) can be re-written as :

$$\mathbb{P}\left(\omega_i - \omega_j \leq x_j - x_i - \underline{S}^I + M_{ij}^{I\alpha}(1 - \delta_{ij})\right) \geq \alpha \tag{4.15}$$

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 18, Miller and Wagner 56).

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 (4.15) is equivalent to :

$$F_{\boldsymbol{\gamma}}\left(x_j - x_i - \underline{S}^I + M_{ij}^{I\alpha}(1 - \delta_{ij})\right) \geq \alpha \tag{4.16}$$

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, (4.16) 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 (4.5_{ij}) and (4.17_{ij}) for all couples $(i, j) \in \mathcal{A} \times \mathcal{A}$ such that $i \neq j$.

Proposition 1. *The chance constraints* (4.5) *in the true model can be replaced by the following deterministic linear constraints :*

$$x_j \geq x_i + S^I(\alpha) - M_{ij}^{I\alpha}(1 - \delta_{ij}) \qquad (i, j) \in \mathcal{A} \times \mathcal{A}, \quad i \neq j \tag{4.17}$$

Remark. *The best expression (smallest while sufficiently large) for the big-M constants* $M_{ij}^{I\alpha}$ *in* (4.17) *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 (4.17) can replace the original IAF separation constraints (4.4) 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 :*

$$x_j \geq x_i + \underline{S}^I - M_{ij}^I(1-\delta_{ij}) \qquad (4.4_{ij})$$
$$x_j \geq x_i + S^I(\alpha) - M_{ij}^{I\alpha}(1-\delta_{ij}) \qquad (4.17_{ij})$$

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* (4.4_{ij}) *and* (4.17_{ij}) *are redundant.*
2. *When* $\delta_{ij} = 1$, *we have the following relations between the constraints* (4.4_{ij}) *and* (4.17_{ij}) *:*
   (a) *constraint* (4.17_{ij}) *dominates constraint* (4.4_{ij}) *if* $\alpha > \mathbb{P}(\gamma \leq 0)$
   (b) *constraint* (4.4_{ij}) *dominates constraint* (4.17_{ij}) *if* $\alpha < \mathbb{P}(\gamma \leq 0)$
   (c) *constraint* (4.4_{ij}) *is equivalent to constraint* (4.17_{ij}) *if* $\alpha = \mathbb{P}(\gamma \leq 0)$
   where $\boldsymbol{\gamma} \stackrel{def}{=} \boldsymbol{\omega_i} - \boldsymbol{\omega_j}$.

Démonstration. 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 (4.4_{ij}) and (4.17_{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 (4.4_{ij}) and (4.17_{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* (4.17) *can substitute for constraints* (4.4) *and* (4.5) *in the true model.*

Démonstration. Proof of Proposition 2 Using Proposition 1, constraints (4.17) can substitute for constraints (4.5) 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 4.3.1), and under Assumptions 1 and 2, due to Proposition 2, the true model (introduced in the beginning of Section 4.3) simplifies as follows :

$$\begin{aligned} \min_{\delta,\, x} \quad & \sum_{\substack{(i,j)\in\mathcal{A}^+\times\mathcal{A}^+ \\ i\neq j}} \delta_{ij} S_{ij} + \lambda\, \mathbb{E}_{\boldsymbol{\omega}}[Q(\delta, x, \omega)] \\ \text{s.t.} \quad & (4.2), (4.3), (4.17), (4.6), (4.7) \\ \text{where :} & \\ Q(\delta, x, \omega) = \quad & (4.11) \\ \text{s.t.} \quad & (4.12), (4.13), (4.14), (4.9), (4.10) \end{aligned} \tag{True model}$$

4.4 Solution methods

The two-stage stochastic program introduced in Section 4.3 presents two main challenges. The first challenge is to deal with the probability constraints in the first stage. In Subsection 4.3.2, 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. 29) 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 [11] 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.

4.4.1 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 (4.1) can be replaced by :

$$\min_{\delta,\, x} \quad \sum_{\substack{(i,j)\in\mathcal{A}^+\times\mathcal{A}^+ \\ i\neq j}} \delta_{ij} S_{ij} + \lambda \sum_{s\in\mathcal{S}} \frac{1}{n_{\mathcal{S}}} Q\left(\delta, x, \omega^s\right) \tag{4.18}$$

Replacing (4.1) by (4.18) 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 [63]. Using the linearized second-stage objective function proposed in Subsection 4.3.1, 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 (4.18)) as follows :

$$Q\left(\delta, x, \omega^s\right) = \min_{\substack{y^s,\, z^{s-} \\ z^{s+},\, z^{s++}}} \sum_{i\in\mathcal{A}} \left(c_1 z_i^{s-} + c_2 z_i^{s+} + c_3 z_i^{s++}\right) \tag{4.19}$$

$$-y_i^s + z_i^{s++} + z_i^{s+} - z_i^{s-} = -x_i - \omega_i^s - \hat{V}_i \qquad i\in\mathcal{A} \quad (\beta_i^s) \tag{4.20}$$

$$-y_i^s \geq -x_i - \omega_i^s - \overline{V_i} \qquad i\in\mathcal{A} \quad (\sigma_i^s) \tag{4.21}$$

$$y_i^s \geq x_i + \omega_i^s + \underline{V_i} \qquad i\in\mathcal{A} \quad (\rho_i^s) \tag{4.22}$$

$$y_j^s - y_i^s \geq S_{ij} - M_{ij}^s(1-\delta_{ij}) \qquad (i,j)\in\mathcal{A}\times\mathcal{A}, i\neq j \quad (\pi_{ij}^s) \tag{4.23}$$

$$-z_i^{s+} \geq -x_i - \omega_i^s - V_i^{\text{med}} \qquad i\in\mathcal{A} \quad (\mu_i^s) \tag{4.24}$$

$$z_i^{s++},\ z_i^{s+},\ z_i^{s-} \geq 0 \qquad i\in\mathcal{A} \tag{4.25}$$

where dual variables corresponding to constraints (4.20) to (4.24) 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 :

$$\begin{aligned} \min_{\substack{\delta,\, x \\ y^s,\, z^{s-} \\ z^{s+},\, z^{s++}}} \quad & \sum_{\substack{(i,j)\in\mathcal{A}^+\times\mathcal{A}^+ \\ i\neq j}} \delta_{ij} S_{ij} + \lambda \sum_{s\in\mathcal{S}} \frac{1}{n_{\mathcal{S}}} \sum_{i\in\mathcal{A}} \left(c_1 z_i^{s-} + c_2 z_i^{s+} + c_3 z_i^{s++}\right) \\ \text{s.t.} \quad & (4.2), (4.3), (4.17), (4.6), (4.7) \\ & (4.20), (4.21), (4.22), (4.23), (4.24), (4.25) \end{aligned} \tag{Determ. Eq.}$$

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 4.4.2.

4.4.2 Benders reformulations

Using Benders decomposition [11, 62], 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. 62), the initial Benders master problem, in the partially-aggregated-cut version, is therefore :

$$\min_{\delta,\, x,\, \nu} \quad \sum_{\substack{(i,j)\in\mathcal{A}^+\times\mathcal{A}^+ \\ i\neq j}} \delta_{ij} S_{ij} + \lambda \sum_{c\in\mathcal{C}} \nu^c \tag{4.26}$$

$$\text{s.t.} \quad \text{first-stage constraints :} \quad (4.2), (4.3), (4.17), (4.6), (4.7)$$

$$\nu^c \geq 0 \qquad c \in \mathcal{C} \tag{4.27}$$

where $\nu = \left(\nu^1, \nu^2, \dots, \nu^K\right)^T$. Constraints (4.27) 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 (4.28) and (4.29) respectively :

$$0 \geq \sum_{s \in c} \left[\frac{1}{n_{\mathcal{S}}} \sum_{i \in \mathcal{A}} \left[\left(-x_i - \omega_i^s - \hat{V}_i \right) \beta_i^{sr} + \left(-x_i - \omega_i^s - \overline{V_i} \right) \sigma_i^{sr} + \left(x_i + \omega_i^s + \underline{V_i} \right) \rho_i^{sr} \right. \right.$$
$$\left. \left. + \left(-x_i - \omega_i^s - V_i^{\text{med}} \right) \mu_i^{sr} \right] + \frac{1}{n_{\mathcal{S}}} \sum_{\substack{(i,j) \in \mathcal{A} \times \mathcal{A} \\ i \neq j}} \left(S_{ij} - M_{ij}^s (1 - \delta_{ij}) \right) \pi_{ij}^{sr} \right]$$
$$r \in \mathcal{R}^c, \; c \in \mathcal{C} \tag{4.28}$$

$$\nu^c \geq \sum_{s \in c} \left[\frac{1}{n_{\mathcal{S}}} \sum_{i \in \mathcal{A}} \left[\left(-x_i - \omega_i^s - \hat{V}_i \right) \beta_i^{st} + \left(-x_i - \omega_i^s - \overline{V_i} \right) \sigma_i^{st} + \left(x_i + \omega_i^s + \underline{V_i} \right) \rho_i^{st} \right. \right.$$
$$\left. \left. + \left(-x_i - \omega_i^s - V_i^{\text{med}} \right) \mu_i^{st} \right] + \frac{1}{n_{\mathcal{S}}} \sum_{\substack{(i,j) \in \mathcal{A} \times \mathcal{A} \\ i \neq j}} \left(S_{ij} - M_{ij}^s (1 - \delta_{ij}) \right) \pi_{ij}^{st} \right]$$
$$t \in \mathcal{T}^c, \; c \in \mathcal{C} \tag{4.29}$$

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 (4.20) to (4.24) 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 4.4 summarizes the model sizes according to the different formulations. Table 4.5 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 4.5, denoted "5-aggregated-cut Benders", involves $K = \dfrac{n_{\mathcal{S}}}{5}$ clusters.

4.4.3 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 the expected performance (in terms of computation time) or even to outperform

TABLE 4.4 – 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 4.5 – 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

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 [62].

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 (39, 12). In the following, we present these two acceleration techniques.

Two-phase solution scheme

[54] 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 [27], 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 (44), 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 [25] and applied to Benders decomposition to solve efficiently facility location problems in [26, 27]. In [26], the "in-out" strategy is applied to generate initial cuts before the Branch-and-Benders-Cut. Similarly, in [27], 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., [47, 12]). 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 [40], 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 4.4.2.

4.5 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 [11]. Secondly, we compare the different solution methods presented in Section 4.4, in terms of computational time. The remaining of this section is organized as follows.

Instances and parameter values are presented in Subsection 4.5.1. The methodology used to determine an appropriate number of scenarios, as well as our main computational results are presented in Subsection 4.5.2. In Subsection 4.5.3, 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 4.5.4. 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.

4.5.1 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 4.2. 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 4.6. 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 4.6 – 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.

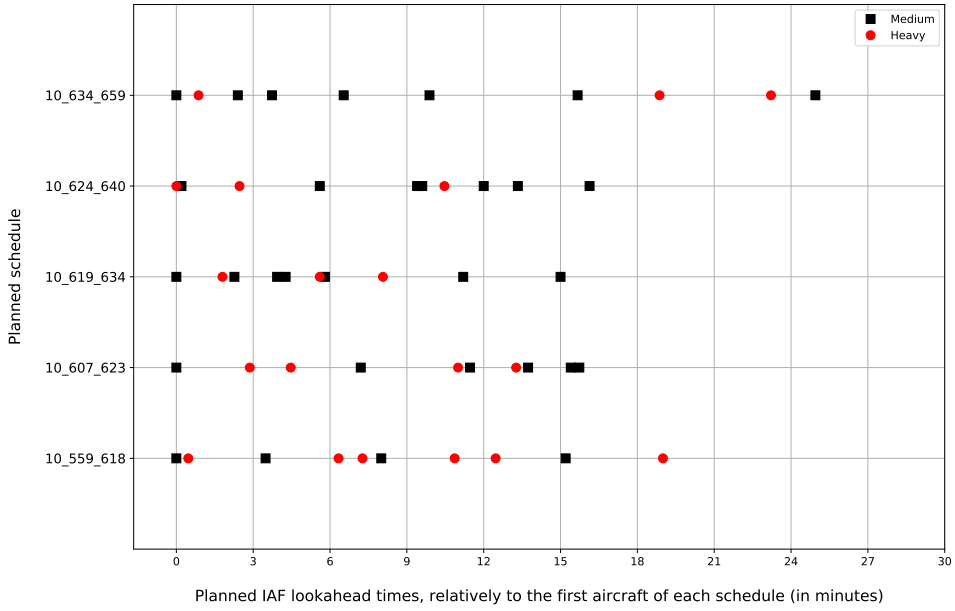


FIGURE 4.2 – Visualization of the five planned schedules.

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 [46]. 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 4.3.2), 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%)

TABLE 4.7 – Rounded buffered separation $S^I(\alpha)$ (in seconds) for uncertainty $\sigma = 30$ sec

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

corresponds 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 4.2. The buffered IAF separation $S^I(\alpha)$ (defined in Subsection 4.3.2) depends on the value of the protection level α, as shown in Table 4.7.

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 4.3.1. 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 4.5.3 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 4.8.

4.5.2 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 4.8 – 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 [43]. 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 4.9 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 4.9, 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. 46, 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 [63].

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 [63].

TABLE 4.9 – 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 4.9.

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.

4.5.3 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 :

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

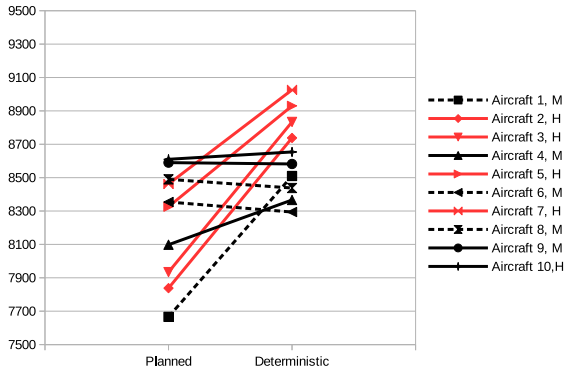
Note that, since we solve a minimization problem, and since the stochastic solution is expected to be better than its deterministic counterpart (i.e., $v^\star_{\text{SP}} \leq v^\star_{\text{EP}}$), we expect the relative VSS to be non-negative.

Relative VSS for each instance for different values of protection level α are reported in Table 4.11. 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 4.7. 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.

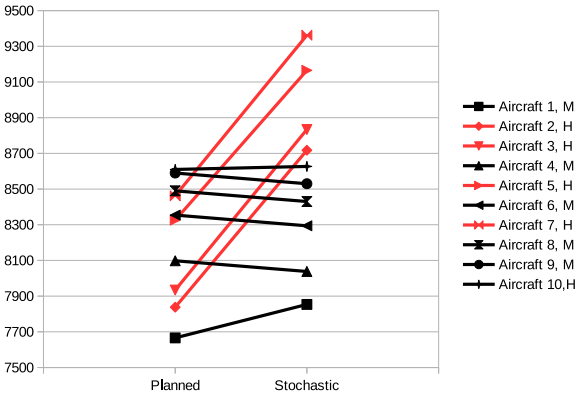
We conclude that the benefit of solving a two-stage stochastic program is more prominent in the situation of high-density air traffic, and when aircraft and controllers have many degrees of freedom for re-scheduling. Also, hedging against uncertainty using buffered separations may lead to a better performance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

In order to study further the features of the stochastic solutions with the highest VSS, we display in Figures 4.3 and 4.4 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 4.3a. 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



(a) Planned vs Deterministic



(b) Planned vs Stochastic

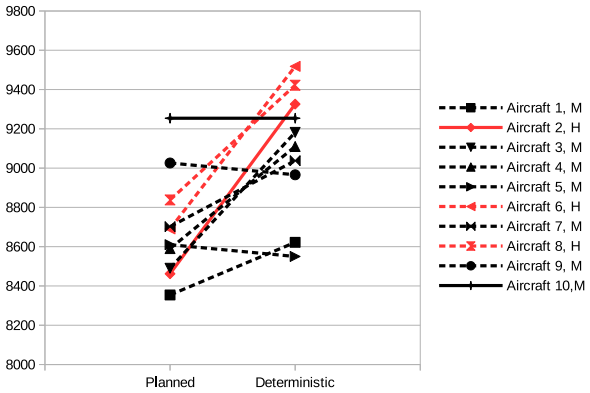
FIGURE 4.3 – Planned, deterministic, and stochastic schedule for instance 10_607_623_W and $\alpha = 50\%$.

position shifts among aircraft from the same turbulence category than their 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. 46.

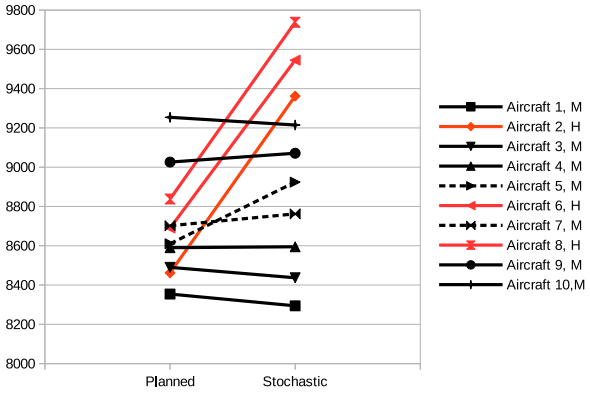
From an operational viewpoint, we conclude that an efficient solution to our two-stage stochastic problem may be obtained first by building a sequence using the rule "lighter aircraft first" and where the relative positions of aircraft from the same wake-turbulence category are conserved. Remark that a similar optimality property is proved in the context of a deterministic version of the aircraft landing problem (see Briskorn and Stolletz [15, 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 4.10. We remark that :



(a) Planned vs Deterministic



(b) Planned vs Stochastic

FIGURE 4.4 – Planned, deterministic, and stochastic schedule for instance 10_619_634_W and $\alpha = 50\%$.

— more scenarios are needed to satisfy the required stability condition ($n_S^* = 500$ 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 4.5 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 $n_S = 500$, the computation time for 607_623_10_W with $\alpha = 50\%$ and $\lambda = 1.0$ is 466.99 seconds (see the appendix), which is three times shorter than with $\lambda = 4.0$.

TABLE 4.10 – 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 4.11 – 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)

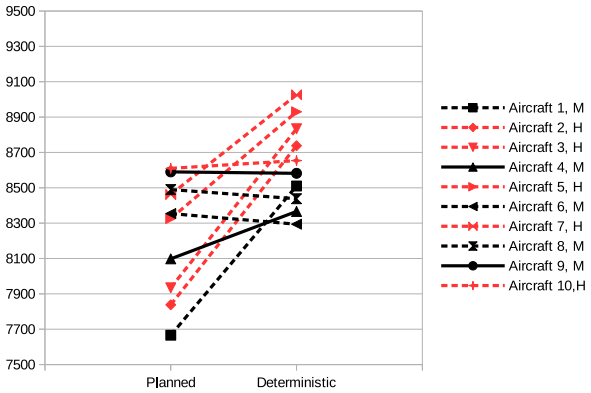
For each instance, the stochastic solution corresponds to the solution with the highest validation score obtained when solving 10 replications of the SAA-problem formulated using the appropriate number of scenarios $n^*_{\mathcal{S}}$ from Table 4.9.

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.

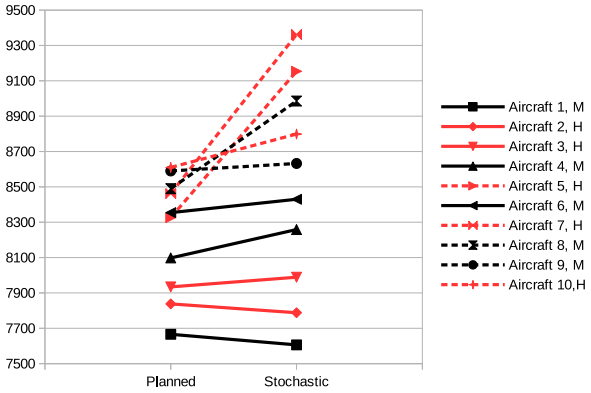
4.5.4 Solution-method performance comparison

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



(a) Planned vs Deterministic



(b) Planned vs Stochastic

FIGURE 4.5 – Planned, deterministic, and stochastic schedule for instance 10_607_623_W, $\alpha = 50\%$ and $\lambda = 4.0$.

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 4.12, 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.

4.6 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 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-

TABLE 4.12 – 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 $k = 1$.

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 [24] 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.

4.7 Appendix : Extensive results

Tables 4.13 to 4.22 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 4.13 – 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 4.14 – 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 4.15 – 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 4.16 – 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 4.17 – 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 4.18 – 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 4.19 – 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 4.20 – 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 4.21 – 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 4.22 – 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

Chapitre 5

Extension au cas de plusieurs points de début d'approche – *Extension to the multiple-IAF case*

Note for English readers. This chapter corresponds to a working article entitled "*Two-stage stochastic programming models for the extended aircraft arrival management problem with multiple pre-scheduling points*".

Résumé du chapitre

Ce chapitre correspond à un article en préparation intitulé "*Two-stage stochastic programming models for the extended aircraft arrival management problem with multiple pre-scheduling points*". Il est rédigé en anglais.

Dans les chapitres précédents, nous avons considéré un environnement opérationnel constitué d'une piste d'atterrissage précédée par un seul point de début d'approche (IAF). Pour cet environnement, le problème d'ordonnancement étendu des arrivées d'avions a été formulé comme un programme stochastique à deux étapes. Dans ce chapitre, nous considérons un environnement opérationnel plus réaliste où plusieurs IAFs précèdent la piste d'atterrissage. Ceci correspond à l'organisation des flux d'arrivée d'avions sur l'aéroport de Paris Charles-de-Gaulle, où chacune des deux pistes d'atterrissage est principalement alimentée par deux IAFs différents. Un tel environnement traduit bien le fait que les pistes forment un goulot d'étranglement du système aéroportuaire.

Nous adaptons la formulation de programmation stochastique à deux étapes proposée dans le chapitre précédent, afin de prendre en compte plusieurs IAFs en première étape. Nous considérons deux variantes du problème d'ordonnancement étendu des arrivées d'avions avec plusieurs IAFs. Dans la première variante, l'affectation d'un IAF pour chaque avion est considérée comme une décision de première étape. Dans la deuxième variante, le choix de l'IAF est supposé fixé, et connu à l'avance. Pour chacune des deux variantes, nous proposons un modèle de programmation stochastique à deux étapes, inspiré de celui du chapitre précédent. Toutefois, attirons l'attention du lecteur que la définition des variables binaires de séquencement au seuil de piste (et aux IAFs), dans ce chapitre, suit la pratique commune dans la théorie de l'ordonnancement, où ces variables n'expriment pas forcément une succession *directe* entre les avions/tâches, contrairement à leur définition dans les deux chapitres précédents.

Une étude numérique exploratoire sur une instance de 15 avions, en utilisant le modèle de la deuxième variante, montre que la solution stochastique est capable de diminuer l'heure du dernier atterrissage en espérance, comparée à la solution du problème du scénario moyen. Également, la tendance croissante de la valeur de la solution stochastique avec l'amplitude de l'incertitude confirme le choix de la programmation stochastique pour faire face aux incertitudes élevées.

Table of content

Abstract

Extended aircraft arrival management under uncertainty has been previously studied in [45] using a two-stage stochastic programming model in the case of a single initial approach fix (IAF) and a single runway. In this paper, we propose an extension of that model to the more realistic case where multiple IAFs are considered. Two problem variants are modeled according to the degree of freedom on IAF assignment to aircraft. In the first variant, IAF are to be assigned to aircraft, as an additional first-stage decision. In the second variant, IAF assignment is fixed and considered as a problem input. Preliminary numerical results on a realistic instance from Paris-Charles-De-Gaulle airport show that the stochastic solution relatively improves the expected makespan compared to its deterministic counterpart.

5.1 Introduction

The Arrival Manager (AMAN) has been a crucial decision-support system for European air traffic controllers (ATCOs) to sequence and schedule, safely and efficiently, aircraft arrivals on destination airports. AMAN's current operational horizon is around 100–200 NM from the destination airport, *i.e.* 30–45 minutes before landing. In the near future, AMAN is foreseen to be upgraded in order to capture aircraft at a distance up to

500 NM, *i.e.* 2–3 hours before landing [70]. Extending AMAN's horizon is expected to allow ATCOs to start sequencing and scheduling earlier, when aircraft are still in their cruise phase, which promotes more eco-efficient aircraft trajectories and hopefully improves airport capacity and reduces delays. However, at this extended horizon, uncertainty related to predicted landing times and expected approach times is significant.

Closely related to this issue, the problem of extended aircraft arrival management under uncertainty has been introduced by Khassiba et al. [46] as the optimization problem consisting in pre-scheduling aircraft arrivals, 2–3 hours before their planned landing times, on a reference air-traffic-route point in the terminal area, called the initial approach fix (IAF), so as to prepare for more efficient inbound air traffic handling, through the terminal area up until landing. The operational setup in [46] corresponds to a set of aircraft arrivals planning to cross the same IAF and to land on the same runway. Khassiba et al. [45] formulate this very problem using a two-stage stochastic mixed-integer programming model with chance constraints. The first-stage optimization problem determines an aircraft sequence and target times over the IAF, so as to minimize the "landing sequence length", expressed as the sum of *final-approach separations* between all pairs of successive aircraft in the landing sequence. Since final-approach separations depend on the pair of considered aircraft, different landing sequences yield different utilization times of the runway. For that reason, minimizing the landing sequence length is used in [45] as a surrogate for maximizing the runway throughput.

During the first stage, *target* IAF times are found, while *actual* times over the IAF are assumed to deviate randomly from target times following known probability distributions. In the second stage, actual IAF times are assumed to be revealed, hence, landing times are determined in view of minimizing a time-deviation impact cost function. Remark that the *target* landing sequence is assumed to be the same as the target IAF sequence. Compared to their deterministic counterparts, the stochastic solutions obtained in [45, 46] are shown to be more robust to the uncertainty occurring within the 2–3 hours before landing.

Remark that both previous studies [45, 46] are restricted to the case of a single IAF (and a single landing runway), while in major airports, several IAFs usually feed the landing runway(s). For example, in Paris-Charles-De-Gaulle airport (CDG), the northern landing runway 27R is mainly fed by two IAFs, named MOPAR and LORNI, while the southern landing runway 26L is usually fed by two IAFs, named OKIPA and BALOX (see Figure 5.1 for an illustration). The second limitation in [45, 46] is that both studies focus on minimizing a time-deviation cost function in the second stage, where the time deviation is computed with respect to a reference landing time, called the *unconstrained* landing time. For a given aircraft, an unconstrained landing time corresponds to the situation where this aircraft flies its preferred trajectory at its preferred speeds in the terminal area from the IAF until landing on the runway, thereby saving fuel and landing with no extra delay incurred during the approach phase. In [46], the second-stage cost function is interpreted as the sum of the time to lose (or to gain) by aircraft, as displayed to air traffic controllers by the typical decision-support tool, AMAN. In [45], the sum of convex piecewise-linear cost functions is minimized in the second stage, which can be interpreted as an estimation of the ATCOs' additional workload required to sequence landings according to the problem solution, as if it were found by AMAN.

In this paper, we address the previous two limitations as follows. We consider a more realistic operational setup, where several IAFs feed a single landing runway. We propose two problem variants corresponding to this setup. In the first variant, the decision-maker assigns an IAF to each aircraft in the first stage, in addition to determining target IAF

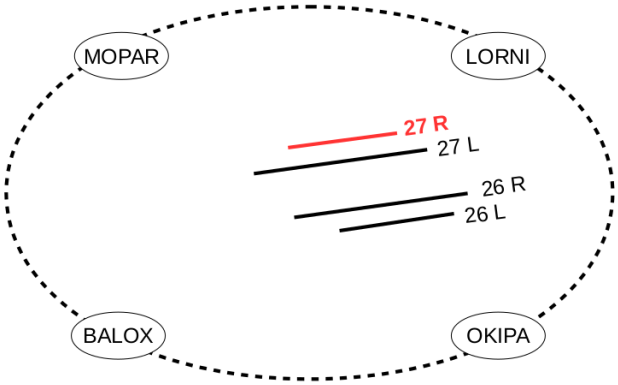


FIGURE 5.1 – Simplified scheme of IAFs surrounding CDG's runways (not to scale).

times, a target sequence on each IAF, and a target landing sequence. In the second variant, IAF assignment is considered as an input, *i.e.* each aircraft must cross a given IAF known in advance. The second-stage problem seeks to schedule aircraft, assumed to be close to the considered IAFs, in order to land on the single runway. We propose two candidate expressions for the second-stage cost function : the last landing time, to be minimized in view of improving the runway throughput, and the time-deviation cost, whose minimization is of interest from an airline viewpoint. In this paper, the reference time considered to compute the landing time deviation is the *planned* landing time (and *not* the *unconstrained* landing time as in [45, 46]), which is known in advance even before departure (*e.g.*, the planned time of arrival shown on a flight ticket or a boarding pass).

The remainder of this paper is organized as follows. In Section 5.2, the problem statement is given. Section 5.3 presents the two-stage stochastic programming model of the first variant. Section 5.4 derives the mathematical model of the second variant. A preliminary computational study is reported in Section 5.5. Concluding remarks and perspectives are given in Section 5.6.

5.2 Problem statement

We consider n aircraft planning to land on the same runway, while there are m available IAFs that any aircraft can fly through before landing. We assume that these aircraft are considered at 2–3 hours before their planned landing times. At this time horizon, we seek to schedule these aircraft on the available IAFs, so as to optimize the *expectation* of a performance indicator (*e.g.*, punctuality, landing rate, *etc*), assuming that actual arrival times to IAFs cannot be predicted with certainty. In this context, scheduling consists of the following decisions. Each aircraft has to be assigned to exactly one IAF, while all aircraft will land on a single runway. For each aircraft, a *target IAF time* is to be computed. Finally, the *target sequence* on each IAF has to be determined, as well as the *target landing sequence*. An operational setup with two IAFs and a single landing runway is illustrated in Figure 5.2.

Let $\mathcal{A} = \{1, 2, \ldots, n\}$ denote the set of aircraft indices, and $\mathcal{I} = \{1, 2, \ldots, m\}$ denote the set of IAF indices. Each aircraft $a \in \mathcal{A}$ has a *planned landing time* P_a^L known in advance, where "L" stands for the landing runway. For each IAF $i \in \mathcal{I}$ and each aircraft

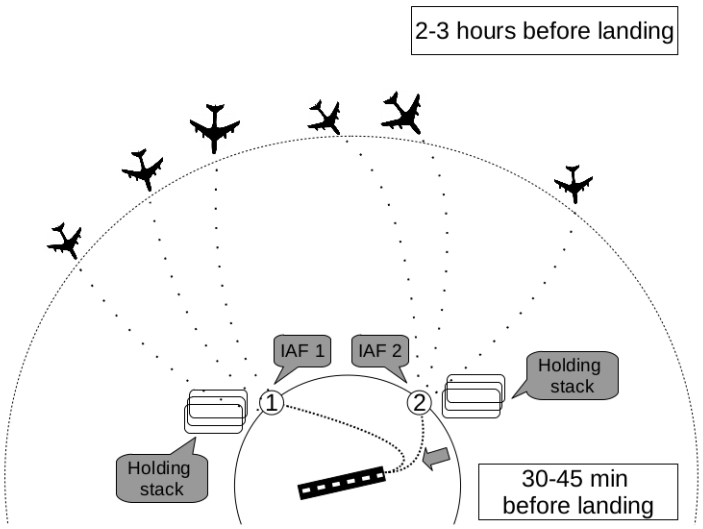


FIGURE 5.2 – Operational environment with two IAFs (not to scale).

$a \in \mathcal{A}$, we are provided with an *unimpeded flight time* $\hat{V}_a^i$, that corresponds to the time required for aircraft a to fly from IAF i to touch down on the landing runway, as if it were alone in the terminal area. Through a direct reverse planning, we can compute, for each aircraft $a \in \mathcal{A}$, *a planned IAF time* at each IAF $i \in \mathcal{I}$, denoted P_a^i, such that :

$$P_a^i = P_a^L - \hat{V}_a^i \tag{5.1}$$

During the en-route flight phase, with 2–3 hour lookahead time, it is possible to relatively expedite or delay aircraft by given amounts of time, mainly through speed change. Let us denote $\underline{d}_a^R$ and $\overline{d}_a^R$, respectively, the *maximal possible time saving*, and the *maximal possible delay* for each aircraft during the *en-route phase*. Hence, we may compute for every aircraft $a \in \mathcal{A}$, *an earliest*, and *a latest IAF time* at each IAF $i \in \mathcal{I}$, denoted E_a^i, and L_a^i respectively, as follows :

$$E_a^i = P_a^i - \underline{d}_a^R \tag{5.2}$$

$$L_a^i = P_a^i + \overline{d}_a^R \tag{5.3}$$

Let $[E_a^i,\ L_a^i]$ denote the IAF-i time window, where must lie the *IAF target time* for aircraft $a \in \mathcal{A}$, if it is assigned to IAF $i \in \mathcal{I}$. Hence, each aircraft has m IAF time windows, each corresponding to one IAF. For any pair of successive aircraft assigned to the same IAF, a minimal distance separation must be satisfied. In practice, this IAF separation is independent of the aircraft pair and is identical for all IAFs. For modeling and optimization purposes, we convert this minimal distance separation into a minimal time separation, that we denote $\underline{S}$. Typically, $\underline{S} = 72$ seconds.

When flying from 2–3 hours to 30–45 minutes before landing, flights are subject to unpredicted phenomena (*e.g.*, bad weather, en-route control actions, *etc*) because of which they may not be able to accurately satisfy their target times to reach the IAFs. We assume that their *actual IAF times* deviate from their target times by random amounts of time (an advance or a delay) that follow known probability distributions.

When aircraft are close to the terminal area, we assume that actual IAF times are known (or, at least, can be predicted) with certainty. In addition, we assume that there is a time point, when actual IAF times of all aircraft are revealed (*i.e.* they can be estimated

without error) before that any aircraft reaches its allocated IAF. This time point may correspond to the time when the "earliest" aircraft is very close to its IAF, while the "latest" aircraft is less than 30-minute far from its IAF, when ground-based trajectory prediction from AMAN is assumed very accurate. This time point corresponds to a second decision stage, where all aircraft, coming from the m different IAFs, are to be scheduled on the same landing runway. We assume this time point to occur in the short time horizon of 30–45 minutes before landing. At this decision stage, a *target landing time* is to be determined for each aircraft. We assume that the target landing sequence has already been determined at the larger time horizon of 2–3 hours.

In case of congestion in the terminal area, ATCOs may resort to *holding stacks* near IAFs, in order to further delay some aircraft. Holding stacks are air-route deviation structures allowing to delay aircraft by keeping them flying circularly in confined areas, usually neighboring IAFs. However, the delay that can be absorbed by holding stacks and other control techniques (such as path stretching) in the terminal area is limited to a given amount of time, noted $\overline{d}_a^T$ for an aircraft $a \in \mathcal{A}$. Also, there is room for expediting an aircraft $a \in \mathcal{A}$ (*e.g.*, through path shortening) within the terminal area, which may save some amount of time, not exceeding a given limit, denoted $\underline{d}_a^T$, with respect to the unimpeded flight time, $\hat{V}_a^i$. Given these maximal possible time saving and delay in the terminal area, a *minimal* and *a maximal flight times*, denoted $\underline{V}_a^i$ and $\overline{V}_a^i$ respectively, can be defined for every aircraft $a \in \mathcal{A}$ and every IAF $i \in \mathcal{I}$, as follows :

$$\underline{V}_a^i = \hat{V}_a^i - \underline{d}_a^T \tag{5.4}$$

$$\overline{V}_a^i = \hat{V}_a^i + \overline{d}_a^T \tag{5.5}$$

Moreover, for any pair of aircraft landing successively, a minimal distance separation must be satisfied during the final-approach phase. Final-approach separations depend on the wake-turbulence categories of the pair of considered aircraft, as defined by the International Civil Aviation Organization (ICAO) : Heavy (H), Medium (M), and Light (L). For modeling and optimization purposes, we convert these minimal distance separations into minimal time separations (see Table 5.1 for numerical values), that we denote S_{ab}, for an ordered pair of aircraft $(a, b) \in \mathcal{A} \times \mathcal{A}$, such that $a \neq b$.

Table 5.1 – Final-approach separations (seconds) according to wake-turbulence categories from the International Civil Aviation Organization (ICAO) [28].

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

In the following, we formulate the extended aircraft arrival management problem with multiple IAFs, using the framework of two-stage stochastic programming. The first-stage time frame starts from 2–3 hours before landing, when aircraft are to be scheduled on the available IAFs. The second-stage time frame starts from 30–45 minutes before landing, when each aircraft is close to its IAF. In the context of extended arrival management, the most important quantities to optimize, such as the sum of flight delays, or the landing rate, are the most meaningful when flights are completed, rather than when they are still

in progress. For that reason, in both variants, we assume that the second-stage objective function fully captures the desired viewpoint (*e.g.*, of ATCOs or airlines), while the first-stage problem has a null objective function.

We derive two variants from the extended aircraft arrival management problem with multiple IAFs, introduced above. Both variants follow the same two-stage partition, while they differ in terms of first-stage decisions. The first variant includes IAF assignment to aircraft as an additional first-stage decision, while in the second variant, no IAF assignment decision is to be made, since IAF are assumed to be pre-assigned to aircraft. The two-stage stochastic programming model of each variant is detailed in the following two sections.

5.3 First-variant model : IAF assignment as a first-stage decision

The general statement of the problem of extended aircraft arrival management problem with multiple IAFs, presented in Section 5.2, assumes that IAFs can be freely assigned to aircraft. We propose to formulate this general case with a two-stage stochastic programming model, inspired by the model proposed in [45]. The main components of the mathematical model of each stage (decision variables and operational constraints) are described in the following. Then, candidate expressions for the second-stage objective function are presented. Finally, the full two-stage stochastic programming model is recalled.

5.3.1 First-stage problem

In the first stage, we assign each aircraft $a \in \mathcal{A}$ to one IAF $i \in \mathcal{I}$ and we determine a *target IAF time* for each aircraft. Also, we find a *target sequence on each IAF*. We assume that the target landing sequence is also determined in the first stage, and must be coherent with the merging of target sequences from each IAF. For instance, two aircraft a and b scheduled to cross the IAF i successively, such that a is before b, must land in the same relative order. However, a third aircraft c scheduled to cross IAF $j \neq i$ can land in any position relatively to a and b (*i.e.* before a and b, or between a and b, or after a and b).

Let x_a be the *target IAF time* for aircraft $a \in \mathcal{A}$. Let ζ_{ai} be a binary variable that assigns aircraft $a \in \mathcal{A}$ to IAF $i \in \mathcal{I}$, as follows :

$$\zeta_{ai} = \begin{cases} 1 & \text{if aircraft } a \text{ is assigned to IAF } i \\ 0 & \text{otherwise} \end{cases}$$

To determine the target landing sequence of aircraft, we introduce a binary variables δ_{ab} for each pair of aircraft $(a, b) \in \mathcal{A} \times \mathcal{A}$, $a \neq b$:

$$\delta_{ab} = \begin{cases} 1 & \text{if aircraft } a \textbf{ lands before} \text{ aircraft } b \\ 0 & \text{otherwise} \end{cases}$$

Remark that, for a pair of aircraft (a, b) assigned to a same IAF, when the binary variable δ_{ab} is equal to 1, then a is required to cross the IAF before b. Hence, the sequencing variable δ_{ab} coherently expresses the relative order between a and b both on the shared

IAF and on the runway. Note that in the case that a and b are not assigned to a same IAF, then δ_{ab} is only meaningful for the landing sequence (of interest in the second-stage problem).

Very importantly, in comparison with the models proposed in [45, 46] where the sequencing binary variables mean *direct precedence* between a pair of aircraft like in the classical Traveling Salesman Problem's (TSP) model, here we define the sequencing variables as in scheduling theory, where direct precedence is *not* required. For instance, if $\delta_{ab} = 1$, then aircraft b must land *after* a, but *not necessarily directly* after, *i.e.*, there might be a number of aircraft landing after a and before b. Also, this holds true, regardless if the aircraft are assigned to a same IAF or not.

This definition of the sequencing variables δ_{ab} guarantees that the target landing sequence is a coherent merging of target sequences from each IAF. In other words, overtaking is not allowed between the IAFs and the runway, among aircraft assigned to a same IAF. For instance, two aircraft a and b scheduled to cross IAF i successively, such that a is before b, must land in the same relative order. However, a third aircraft c scheduled to cross IAF $j \neq i$ can land in any position relatively to a and b (*i.e.* before a and b, or between a and b, or after a and b).

If an aircraft $a \in \mathcal{A}$ is assigned to IAF $i \in \mathcal{I}$, then its target IAF time x_a must lie within an appropriate time window :

$$x_a \in \left[E_a^i\,,\, L_a^i\right] \qquad \text{if aircraft } a \in \mathcal{A} \text{ is to cross IAF } i \in \mathcal{I}. \tag{5.6}$$

Moreover, for safety reasons, target IAF times of a pair of successive aircraft crossing the same IAF $i \in \mathcal{I}$ must be separated by the minimal IAF time separation, $\underline{S}$. For all pairs of aircraft $(a, b) \in \mathcal{A} \times \mathcal{A}$, $a \neq b$, we require that :

$$x_b \geq x_a + \underline{S} \qquad \text{if aircraft } a \text{ and } b \text{ are assigned to a same IAF, and } a \text{ is followed by } b. \tag{5.7}$$

For a given aircraft $a \in \mathcal{A}$, its *actual IAF time* will deviate, with respect to its target IAF time x_a, by a random amount of time, denoted $\boldsymbol{\omega_a}$, assumed to follow a known probability distribution. Hence, the actual IAF time of an aircraft $a \in \mathcal{A}$ is $(x_a + \omega_a)$, where ω_a is a possible realization of the random variable $\boldsymbol{\omega_a}$.

In the second stage, all random variables are assumed to be revealed, giving rise to a second-stage optimization problem, that we formulate in the following subsection.

5.3.2 Second-stage problem

In the second stage, we seek to determine a *target landing time* for each aircraft $a \in \mathcal{A}$, while the target landing sequence is already found in the first stage. Let y_a denote the *target landing time* for aircraft $a \in \mathcal{A}$. Given that the actual time of aircraft $a \in \mathcal{A}$ to cross IAF $i \in \mathcal{I}$ is $(x_a + \omega_a)$, and that minimal and maximal flight times from IAF i to touch down are $\underline{V}_a^i$ and $\overline{V}_a^i$ respectively, then we can compute the following landing time window, $\left[E_a^L(\omega_a, i)\,,\, L_a^L(\omega_a, i)\right]$, that the target landing time y_a must satisfy :

$$y_a \in \left[E_a^L(\omega_a, i)\,,\, L_a^L(\omega_a, i)\right] \tag{5.8}$$

where :

$$E_a^L(\omega_a, i) = (x_a + \omega_a) + \underline{V}_a^i \tag{5.9}$$

$$L_a^L(\omega_a, i) = (x_a + \omega_a) + \overline{V}_a^i \tag{5.10}$$

Remark that $\boldsymbol{E_a^L}(\boldsymbol{\omega_a}, i)$ and $\boldsymbol{L_a^L}(\boldsymbol{\omega_a}, i)$ are themselves random variables, since they depend on the random time deviation, $\boldsymbol{\omega_a}$. Hence, landing time windows are only known with certainty in the second stage. Also, according to the value of the decision variable x_a and the realization ω_a, one expects that $P_a^L \in \left[E_a^L(\omega_a, i)\ ,\ L_a^L(\omega_a, i)\right]$, may *or* may *not* hold. Typically, in case of a long delay in the en-route phase, the planned landing time P_a^L may become infeasible for some realizations of $\boldsymbol{\omega_a}$, *i.e.* $P_a^L < E_a^L(\omega_a, i) < L_a^L(\omega_a, i)$.

Operational constraints related to final-approach separations that must be satisfied between any pair of successively landing aircraft $(a, b) \in \mathcal{A} \times \mathcal{A}$, $a \neq b$, can be expressed as follows :

$$y_b \geq y_a + S_{ab} \qquad \text{if aircraft } a \text{ lands before } b, \tag{5.11}$$

where S_{ab} is the minimum final-approach separation between aircraft $a \in \mathcal{A}$ and $b \in \mathcal{A}$.

In the following subsection, we propose two candidate expressions for the second-stage objective function.

5.3.3 Candidate expressions for the second-stage objective function :

The two main stakeholders considered in this paper are : air traffic controllers and airlines. On the one hand, air traffic controllers, in the terminal area, care mainly about ensuring safety and increasing runway throughput. Hence, a typical concern of ATCOs is to minimize the landing time of the last aircraft in the sequence. On the other hand, punctuality is of a great importance for airlines, since delay may induce high direct and indirect costs to them. For that reason, another second-stage objective function is to minimize delay costs with respect to planned landing times.

In the following, we formulate each of these two candidate second-stage objective functions.

Minimizing the landing time of the last aircraft : A performance objective from an ATCOs' viewpoint is to maximize the landing rate, or equivalently, to minimize the landing time of the last aircraft in the sequence, for a given set of aircraft. According to this viewpoint, the second-stage objective function can be formulated using an auxiliary variable z, and a set of corresponding constraints, as follows :

$$\min_{y,z} \quad z \tag{5.12}$$

$$\text{s.t.} \quad z \geq y_a \qquad a \in \mathcal{A} \tag{5.13}$$

In scheduling theory, this is often referred to minimizing the *makespan*. However, since the second-stage problem is subject to uncertainty, we are content with minimizing the *expected makespan*, in the context of our two-stage stochastic optimization problem.

Minimizing deviation costs with respect to planned landing times : Unlike in [45] where the second-stage cost is interpreted as the workload of a controller applying AMAN sequencing recommendations, we propose in this paper to minimize the total deviation cost with respect to the planned landing times (P_a^L for $a \in \mathcal{A}$), which corresponds to airlines' viewpoint. European airline delay cost reference values report [19] defines, for each aircraft type, unitary delay costs increasing with the amount of delay.

Accordingly, we define a convex piecewise-linear cost function $f_a : y_a \mapsto f_a(y_a)$ for each aircraft. Note that unitary time advance costs can be extrapolated from data in [19], or simply set to zero. Figure 5.3 illustrates such a function with three breakpoints, and when $P_a^L \in \left[E_a^L(\omega_a, i)\ ,\ L_a^L(\omega_a, i)\right]$. The second-stage objective function is the total deviation cost to be minimized with respect to planned landing times for all aircraft :

$$\min_{y} \quad \sum_{a \in \mathcal{A}} f_a(y_a) \tag{5.14}$$

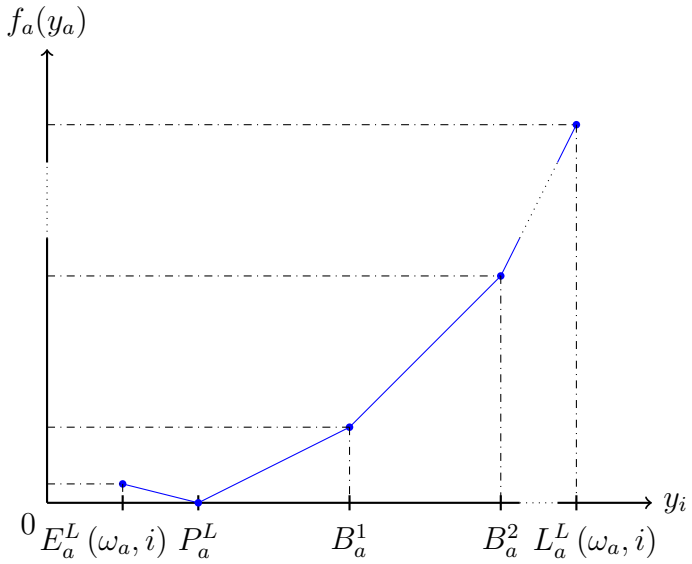


FIGURE 5.3 – Convex piecewise linear cost function with 3 breakpoints P_a^L, B_a^1, and B_a^2, and when $P_a^L \in \left[E_a^L(\omega_a, i)\ ,\ L_a^L(\omega_a, i)\right]$.

5.3.4 Full two-stage stochastic model for the first variant

We add to the first-stage model the following auxiliary binary decision variables ϕ_{ab}'s, defined for all pairs of aircraft $(a, b) \in \mathcal{A} \times \mathcal{A}$, $a \neq b$, as follows :

$$\phi_{ab} = \begin{cases} 1 & \text{if aircraft } a \text{ and } b \text{ are both assigned to a same IAF} \\ 0 & \text{otherwise} \end{cases}$$

Note that these binary variables are inspired by the multiple-runway formulation of the aircraft landing problem proposed in [5, Section 4].

The two-stage stochastic programming model of the extended aircraft arrival management problem under uncertainty with multiple IAFs, and where IAF assignment is

considered as a first-stage decision, reads :

$$\min_{\substack{\delta,\, x \\ \zeta,\, \phi}} \quad \mathbb{E}_{\boldsymbol{\omega}}[Q(\zeta, \delta, x, \omega)] \tag{5.15}$$

$$\text{s.t.} \quad \sum_{i \in \mathcal{I}} \zeta_{ai} = 1 \qquad a \in \mathcal{A} \tag{5.16}$$

$$\phi_{ab} = \phi_{ba} \qquad (a,b) \in \mathcal{A} \times \mathcal{A}, \quad a < b \tag{5.17}$$

$$\phi_{ab} \geq \zeta_{ai} + \zeta_{bi} - 1 \qquad i \in \mathcal{I}\,, \quad (a,b) \in \mathcal{A} \times \mathcal{A}, \quad a < b \tag{5.18}$$

$$\delta_{ab} + \delta_{ba} = 1 \qquad (a,b) \in \mathcal{A} \times \mathcal{A}, \quad a < b \tag{5.19}$$

$$x_b \geq x_a + \underline{S} - M_{ab}(2 - \phi_{ab} - \delta_{ab}) \qquad (a,b) \in \mathcal{A} \times \mathcal{A}, \quad a \neq b \tag{5.20}$$

$$\sum_{i \in \mathcal{I}} E_a^i \zeta_{ai} \leq x_a \leq \sum_{i \in \mathcal{I}} L_a^i \zeta_{ai} \qquad a \in \mathcal{A} \tag{5.21}$$

$$\zeta_{ai} \in \{0,1\} \qquad i \in \mathcal{I}, \quad a \in \mathcal{A} \tag{5.22}$$

$$\phi_{ab} \in \{0,1\} \qquad (a,b) \in \mathcal{A} \times \mathcal{A}, \quad a \neq b \tag{5.23}$$

$$\delta_{ab} \in \{0,1\} \qquad (a,b) \in \mathcal{A} \times \mathcal{A}, \quad a \neq b \tag{5.24}$$

where :

$$Q(\zeta, \delta, x, \omega) = \min_{y} \quad f\,(x\,, \omega\,, y) \tag{5.25}$$

$$\text{s.t.} \quad y_b \geq y_a + S_{ab} - M_{ab}^L(1 - \delta_{ab}) \qquad (a,b) \in \mathcal{A} \times \mathcal{A}, \quad a \neq b \tag{5.26}$$

$$\sum_{i \in \mathcal{I}} \underline{V_a}^{\,i} \zeta_{ai} \leq y_a - (x_a + \omega_a) \leq \sum_{i \in \mathcal{I}} \overline{V_a}^{\,i} \zeta_{ai} \qquad a \in \mathcal{A} \tag{5.27}$$

The objective function (5.15) involves minimizing exclusively the expectation of the second-stage objective function, while the first-stage objective function is assumed to be null. Constraints (5.16) ensure that each aircraft is assigned to exactly one IAF. Constraints (5.17) express the symmetry of the auxiliary variables ϕ_{ab}'s. Constraints (5.18) ensure the logical coherence of the decision variables ϕ_{ab}'s and the decision variables ζ_{ai}'s. Constraints (5.19) express the logical order between any pair of aircraft. Constraints (5.20) ensure separation between target IAF times of any pair of successive aircraft assigned to a same IAF, where the big-M constant can be set to :

$$M_{ab} = \max_{k \in \mathcal{I}} \left\{ L_a^k \right\} + \underline{S} - \min_{k \in \mathcal{I}} \left\{ E_b^k \right\} \tag{5.28}$$

The expression (5.28) can be obtained as follows. The big-M constant M_{ab} is required to be an upper bound of the expression $(x_a + \underline{S} - x_b)$. For a given feasible first-stage solution, we know that there is some IAF $i \in \mathcal{I}$, such that $x_a \leq L_a^i \leq \max_{k \in \mathcal{I}} \left\{ L_a^k \right\}$, and some IAF $j \in \mathcal{I}$ (not necessarily different from i), such that $-x_b \leq -E_b^j \leq -\min_{k \in \mathcal{I}} \left\{ E_b^k \right\}$. Since IAF assignment is not known in advance, then (5.28) is a tight upper bound of expression $(x_a + \underline{S} - x_b)$.

Constraints (5.21) enforce each target IAF time to lie within an appropriate time window that depends on the assigned IAF. Constraints (5.22), (5.23), and (5.24) stipulate the binary nature of decision variables ζ_{ai}'s, ϕ_{ab}'s and δ_{ab}'s.

The optimal value of the second-stage problem is noted $Q(\zeta, \delta, x, \omega)$. Objective function (5.25) refers to the second-stage objective function, where f may have any expression, *e.g.*, one of the two candidate expressions proposed in Subsection 5.3.3. Constraints (5.26) ensure final-approach separation between any pair of landing aircraft, where the big-M constant can be set to :

$$M_{ab}^{L} = \left(\omega_a + \max_{k\in\mathcal{I}}\left\{L_a^k + \overline{V}_a^k\right\}\right) + S_{ab} - \left(\omega_b + \min_{k\in\mathcal{I}}\left\{E_b^k + \underline{V}_b^k\right\}\right) \tag{5.29}$$

The expression (5.29) can be obtained as follows. As before, M_{ab}^{L} is required to be an upper bound of the expression $(y_a + S_{ab} - y_b)$. For a given feasible solution of the second-stage problem, we know that there is some IAF $i \in \mathcal{I}$, such that $y_a \leq x_a + \omega_a + \overline{V}_a^i \leq L_a^i + \omega_a + \overline{V}_a^i \leq \omega_a + \max_{k\in\mathcal{I}}\left\{L_a^k + \overline{V}_a^k\right\}$. A similar reasoning can be made to upper bound $(-y_b)$.

Constraints (5.27) enforce that the flight time, for a given aircraft, between its assigned IAF and the runway, lies within an appropriate flight time window.

5.4 Second-variant model : IAF assignment as a problem input

In the previous section, we presented a two-stage stochastic programming model for the general case of the extended aircraft arrival management problem, where IAF assignment is considered as a first-stage decision. Although it is reasonable to consider this degree of freedom, often, IAFs are pre-assigned to aircraft arrivals according to their geographical origin and their aircraft propulsion type (*e.g.*, see [55, Figure 1]). For example, in Paris-Charles-De-Gaulle airport (CDG), the two northern IAFs, MOPAR and LORNI, feed the landing runway 27R. North-western arrival flow usually crosses the north-western IAF, MOPAR, while the north-eastern IAF, LORNI, is pre-assigned to the north-eastern arrival flow.

From an operational viewpoint, changing the IAF, for a given aircraft, is usually due to the modification of the landing runway, since each runway is fed by a specific subset of IAFs. For instance, in CDG, consider an aircraft arriving from the north-west, planning to land on runway 27R, and to cross IAF MOPAR beforehand. If this aircraft is re-scheduled to land on the southern runway 26L (regardless the operational reason), then its IAF would be, very likely, updated to the south-western IAF, BALOX.

Since in our operational setup we consider a single runway, no landing runway modification is possible ; we may, then, consider the realistic case where IAF assignment is also fixed. Consequently, in this section, we consider the problem variant where IAFs are already pre-assigned to aircraft, 2–3 hours before landing. In this variant, the IAF assignment is considered as a problem input ; hence, the set of aircraft indices, $\mathcal{A}$, can be partitioned, as a preprocessing, into m subsets $\mathcal{A}^i$, for $i \in \mathcal{I}$, where each subset of aircraft indices is pre-assigned to a given IAF. Hence, $\mathcal{A} = \cup_{i=1}^{m}\mathcal{A}^i$ and $\mathcal{A}^i \cap \mathcal{A}^j = \emptyset$, $\forall\, i \neq j$. In the case of a fixed pre-assignment of IAFs, the two-stage stochastic programming model,

presented in Section 5.3, simplifies as follows :

$$\min_{\delta,\, x} \quad \mathbb{E}_{\boldsymbol{\omega}}[Q(\delta, x, \omega)] \tag{5.30}$$

$$\text{s.t.} \quad \delta_{ab} + \delta_{ba} = 1 \qquad (a,b) \in \mathcal{A} \times \mathcal{A}, \quad a < b \tag{5.31}$$

$$x_b \geq x_a + \underline{S} - M_{ab}(1 - \delta_{ab}) \qquad i \in \mathcal{I}\,, (a,b) \in \mathcal{A}^i \times \mathcal{A}^i, \quad a \neq b \tag{5.32}$$

$$E_a^i \leq x_a \leq L_a^i \qquad i \in \mathcal{I}\,, a \in \mathcal{A}^i \tag{5.33}$$

$$\delta_{ab} \in \{0,1\} \qquad (a,b) \in \mathcal{A} \times \mathcal{A}, \quad a \neq b \tag{5.34}$$

where :

$$Q(\delta, x, \omega) = \min_{y} \quad f\,(x\,, \omega\,, y) \tag{5.35}$$

$$\text{s.t.} \quad y_b \geq y_a + S_{ab} - M_{ab}^L(1 - \delta_{ab}) \qquad (a,b) \in \mathcal{A} \times \mathcal{A}, \quad a \neq b \tag{5.36}$$

$$\underline{V_a}^{\,i} \leq y_a - (x_a + \omega_a) \leq \overline{V_a}^{\,i} \qquad i \in \mathcal{I}\,, a \in \mathcal{A}^i \tag{5.37}$$

The main simplifications are as follows. When IAF assignment is fixed, there is naturally no need neither for the IAF assignment decision variables ζ_{ai}'s, nor for the decision variables ϕ_{ab}'s, identifying whether two aircraft are assigned to the same IAF or not. Also, with respect to the first-variant model, IAF separation constraints (5.20), IAF time-window constraints (5.21), and runway time-window constraints (5.27) are updated into constraints (5.32), (5.33), and (5.37) respectively. In the following, we comment in details the model of this second variant.

Objective function (5.30) and constraints (5.31) are similar to their counterparts in the first variant, (5.15) and (5.19) respectively. Note that first-stage decision variables are now limited to δ_{ab}'s and x_a's. The IAF separation constraints (5.32) are expressed only for pairs of aircraft crossing a same IAF, *i.e.* pairs of aircraft from the same subset $\mathcal{A}^i$, for $i \in \mathcal{I}$. For a pair $(a,b) \in \mathcal{A} \times \mathcal{A}$, $a \neq b$, the big-M constant in (5.32) can be set as follows :

$$M_{ab} = L_a^i + \underline{S} - E_b^i \tag{5.38}$$

For each aircraft, the IAF is known in advance, hence the IAF time window is known without ambiguity. Then, IAF time-window constraints (5.33) are formulated straightforwardly, using subsets $\mathcal{A}^i$, for $i \in \mathcal{I}$. Constraints (5.34), similarly to (5.24) in the first-variant model, stipulate the binary nature of the sequencing variables δ_{ab}'s.

For the second-stage problem, $Q(\delta, x, \omega)$ stands for its optimal value. Objective-function (5.35) and runway-separation constraints (5.36) are identical to their counterparts in the first-variant model, (5.25) and (5.26) respectively. However, for a pair of aircraft (a,b) such that aircraft a is pre-assigned to IAF i, and aircraft b to IAF j, the big-M constant can be expressed more concisely than in expression (5.29) as follows :

$$M_{ab}^L = \left(L_a^i + \omega_a + \overline{V}_a^{\,i}\right) + S_{ab} - \left(E_b^j + \omega_b + \underline{V_b^j}\right) \tag{5.39}$$

Finally, landing time-window constraints (5.37) simplify (compared to their counterparts (5.27), in the first variant) using the fact that the minimal and the maximal flight time for each aircraft, from the pre-assigned IAF to the runway, are known in advance.

5.5 Computational study

In the previous two sections, we presented two-stage stochastic programming models for two variants of the extended aircraft arrival management problem with multiple IAFs. The first variant, presented in Section 5.3, corresponds to the general case where IAF assignment is a first-stage decision, while the second variant, formulated in Section 5.4, deals with a realistic special case where IAF assignment is fixed in advance. In this section, we report a preliminary computational study based on the model of the second variant. We consider minimizing the landing time of the last aircraft in the sequence, as the second-stage objective function. The results are obtained with a Python 2.7 code calling IBM ILOG CPLEX 12.7.1, and running on a Linux platform with eight 2.66 GHz Xeon processors and 32 GB of RAM. Test data are given in Subsection 5.5.1. The solution method is explained Subsection 5.5.2. Results are presented and commented in Subsection 5.5.3.

5.5.1 Data

We use realistic data from CDG consisting of $n = 15$ aircraft planning to land on the northern runway 27R, and to cross the IAFs between 5 :59 AM and 6 :11 AM. Originally, 8 aircraft planned to cross IAF MOPAR, 5 aircraft to cross IAF LORNI, and 2 aircraft to cross IAF OKIPA. We updated the IAF of the last two aircraft to IAF LORNI, in order to build an instance with $m = 2$ IAFs, and a more balanced number of aircraft on each IAF.

Flight times from each IAF to the runway : Minimal, unconstrained, and maximal flight times from each IAF to the runway are assumed to be aircraft independent, and are shown in Table 5.2.

Maximal possible time saving and maximal possible delay in the en-route phase : We choose small and aircraft-independent maximal possible time saving and maximal possible delay ($\underline{d}_a^R = 60$ seconds, $\overline{d}_a^R = 300$ seconds, $\forall\, a \in \mathcal{A}$).

More details about the instance are reported in Table 5.3, where aircraft are sorted according to their earliest IAF time.

Minimal time separations : The minimal time separation on each IAF is $\underline{S} = 72$ seconds, like in [45, 46]. Final-approach time separations are as specified in Table 5.1.

Random IAF time deviations : We assume that IAF time deviations, $\boldsymbol{\omega_a}$, $a \in \mathcal{A}$, are randomly distributed following a same normal distribution with mean zero, and standard deviation σ. Test values for σ are 30, 60 and 90 seconds.

5.5.2 Solution method

Due to the continuous normal distribution of $\boldsymbol{\omega_a}$, $a \in \mathcal{A}$, a closed-form expression for $\mathbb{E}_{\boldsymbol{\omega}}\left[Q\left(\delta, x, \omega\right)\right]$ is difficult to find. Instead, for each test value of σ, we sample $n_{\mathcal{S}} = 100$ scenarios, where each scenario corresponds to the realization of the random vector $\boldsymbol{\omega} = (\boldsymbol{\omega_1}, \boldsymbol{\omega_2}, \ldots, \boldsymbol{\omega_n})$. Then, we formulate an *approximate* problem as follows. We create $n_{\mathcal{S}}$

IAF	RWY	IAF-to-RWY flight time		
		Min.	Unconst.	Max.
1 (MOPAR)	27R	720	780	1800
2 (LORNI)	27R	600	660	1800

TABLE 5.2 – Minimal, unconstrained, and maximal flight times (in seconds) from each IAF to runway 27R ($\underline{V}_a^i$, $\hat{V}_a^i$, and $\overline{V}_a^i$, for any $a \in \mathcal{A}$, respectively).

Id	AC type	WTC	IAF	IAF times			Planned landing
				Earliest, E_a^i	Planned, P_a^i	Latest, L_a^i	P_a^L
1	A320	M	2	7126	7186	7486	7846
2	A388	H	1	7140	7200	7500	7980
3	A319	M	2	7231	7291	7591	7951
4	B772	H	1	7316	7376	7676	8156
5	A343	H	1	7344	7404	7704	8184
6	A320	M	2	7366	7426	7726	8086
7	A333	H	1	7452	7512	7812	8292
8	B763	H	1	7500	7560	7860	8340
9	E190	M	2	7582	7642	7942	8302
10	B77W	H	1	7696	7756	8056	8536
11	A321	M	2	7710	7770	8070	8430
12	A333	H	1	7764	7824	8124	8604
13	A319	M	2	7778	7838	8138	8498
14	E170	M	1	7828	7888	8188	8668
15	B739	M	2	7838	7898	8198	8558

TABLE 5.3 – Instance details including for each aircraft : the identifier (Id), the aircraft type (AC type), the wake-turbulence category (WTC), the pre-assigned IAF (IAF), the IAF times ($E_a^i, P_a^i,$ and L_a^i), and the planned landing time (P_a^L).

copies of the second-stage problem, one copy per scenario. We approximate the expectation of the second-stage optimal value, $\mathbb{E}_{\boldsymbol{\omega}}\left[Q\left(\delta, x, \omega\right)\right]$, by a sample average over the $n_{\mathcal{S}}$ sampled scenarios, $\dfrac{1}{n_{\mathcal{S}}}\displaystyle\sum_{s=1}^{n_{\mathcal{S}}} Q\left(\delta, x, \omega^s\right)$, where $\omega^s = (\omega_1^s, \omega_2^s, \ldots, \omega_n^s)$ corresponds to the realization of the random vector $\boldsymbol{\omega}$ in scenario s. Finally, we solve the approximate problem as a large mixed-integer linear program using the automatic Benders decomposition provided in CPLEX. We annotate our model, for CPLEX, so that the first-stage decision variables (δ and x) are kept in the Benders master problem, and the second-stage decision variables are in $n_{\mathcal{S}}$ separate Benders subproblems (where decision variables y and z from the *same* second-stage *scenario* problem are kept in the *same Benders subproblem*).

We call the solution obtained (δ_{SP}, x_{SP}), the *stochastic solution*.

On the other hand, we formulate the so-called *expected value problem*, which corresponds to an approximate problem of our two-stage stochastic problem, where there is only one scenario, specifically the *mean* scenario, in the second stage. Remark that the mean scenario in our computational study corresponds to null IAF time deviations for all aircraft, *i.e.* every aircraft arriving at its pre-assigned IAFs exactly on its target time. The expected value problem corresponds to a purely deterministic approach to the operational

problem, that overlooks the uncertainty, by reducing the probability distributions to their mean values. We solve the expected value problem by Branch-and-cut using CPLEX, and we call the solution obtained (δ_{EV}, x_{EV}), the *deterministic solution*.

In order to compare the quality of both solutions (the stochastic and the deterministic ones), we sample $n_{\mathcal{S}_v} = 1000$ scenarios that we use as *reference scenario tree* as called in [43]. We formulate the *validation problem*, as an approximate problem of our two-stage stochastic problem, where the scenario set corresponds to the sampled reference scenario tree. We evaluate both solutions on this reference tree, and we call the obtained objective-function values, the *validation scores*. Let v_{SP} denote the validation score of the stochastic solution, (δ_{SP}, x_{SP}), and v_{EV} the validation score of the deterministic solution, (δ_{EV}, x_{EV}). Remark that v_{SP} expresses the expected makespan, if the stochastic solution is implemented, while v_{EV} expresses the expected makespan, if the deterministic solution is implemented.

The difference between the two validation scores, v_{SP} and v_{EV}, denoted VSS, is called the *value of the stochastic solution*, and expressed as follows :

$$\text{VSS} = v_{SP} - v_{EV} \tag{5.40}$$

The value of the stochastic solution quantifies the benefit, in terms of expected makespan, from taking into account the probability distribution of IAF deviation times, through two-stage stochastic programming. More specifically, VSS measures the expected saving (or loss) in runway utilization time for a given set of aircraft, when the solution applied to the extended aircraft arrival problem is the one provided by our two-stage stochastic model, and not by an uncertainty-unaware approach.

Our numerical results are presented in the next subsection.

5.5.3 Results

When solving the approximate stochastic problem, we choose to stop CPLEX with a relative optimality gap of 4%, in an attempt to limit the computation time while obtaining good-quality solution. Table 5.4 reports for each uncertainty amplitude ($\sigma = 30$, 60 and 90 seconds), the performance of the stochastic and the deterministic solutions. For the stochastic solution, each test case is repeated 5 times, each time with a different sample of $n_{\mathcal{S}} = 100$ scenarios. Hence, results reported in Table 5.4 for the stochastic solution are averaged over 5 repetitions / replications. Row "CPU (sec)" reports the computation time, in seconds, as reported by CPLEX. Row "Status (gap)" tells whether CPLEX found a solution and proved it is optimal, in which case "Optimal (0.0%)" is reported, otherwise, if a feasible solution was found but the relative optimality gap is not closed, then "Feasible (gap%)" is reported, with the value of the gap. Validation scores are also shown. The value of the stochastic solution "VSS" is displayed for each uncertainty amplitude.

The computation times suggest that the deterministic solutions are easier to obtain (CPU is around 2 seconds), as expected, since the problem size is smaller. As for the stochastic solution, the computation time to reach an optimality gap of 4% can be very long (more than 5 minutes). However, very interestingly, when the uncertainty increases, we observe that this computation time decrease down to 1 minute. Finally, the value of the stochastic solution proves that the expected makespan can be shortened by around 5 seconds in the case of small uncertainty, and by 23 seconds for the largest tested uncertainty amplitude. This suggests that there is some benefit, in terms of runway utilization, to solve the two-stage stochastic programming formulation rather than the deterministic

one where IAF deviation times, with respect to target times, are assumed to be null. Also, the increasing trend of VSS with the uncertainty amplitude proves that the two-stage stochastic programming is more beneficial as uncertainty increases.

		Stochastic Solution	Deterministic Solution
$\sigma = 30$	CPU (sec)	313.3	2.3
	Status (gap)	Feasible (3.8%)	Optimal (0.0%)
	Validation score	9007.0	9012.3
		VSS = - 5.3 sec	
$\sigma = 60$	CPU (sec)	115.7	2.3
	Status (gap)	Feasible (4.0%)	Optimal (0.0%)
	Validation score	9037.4	9048.4
		VSS = - 11.0 sec	
$\sigma = 90$	CPU (sec)	60.1	2.2
	Status (gap)	Feasible (4.0%)	Optimal (0.0%)
	Validation score	9065.1	9088.4
		VSS = - 23.3 sec	

TABLE 5.4 – Stochastic and deterministic solution comparison for different levels of uncertainty.

5.6 Conclusions and perspectives

A two-stage stochastic programming model for the extended aircraft arrival management problem has been proposed in the literature, by Khassiba et al. [45], for the case of a single IAF and a single runway. In this paper, we update their model to handle the case of several IAFs. Remark that the definition of the sequencing binary decision variables, in this paper, follows the common practice in scheduling theory, unlike their definition in [45, 46] that is inspired by the classical Traveling Salesman Problem's (TSP) model. We formulate two variants of the extended aircraft arrival management problem with multiple IAFs. The first variant considers new decision variables in the first stage, that assign one IAF to each considered aircraft. In the second variant, IAF assignment is assumed to be given as an input, which corresponds to a realistic operational setup, such in Paris-Charles-De-Gaulle airport. For each variant, we propose a two-stage stochastic programming model.

A preliminary computational study on a realistic instance of 15 aircraft, based on the model of the second variant, shows that the stochastic solution provides an expected makespan shorter than its deterministic counterpart by 23 seconds. Also, the increasing trend of VSS with the uncertainty amplitude proves that the two-stage stochastic programming is more beneficial as uncertainty increases. Future work will focus on the first-variant model to evaluate the benefit of a flexible IAF assignment. Also, different second-stage objective functions, *e.g.*, expressing the airlines viewpoint as suggested in Subsection 5.3.3, could be implemented and tested.

Chapitre 6

Conclusion

Les problématiques de la gestion du trafic aérien puisent leur importance dans l'écart grandissant entre la croissance du trafic aérien mondial et la capacité limitée des aéroports, et de l'espace aérien, justifiant le recours à l'optimisation mathématique afin de mieux profiter des ressources existantes, tout en satisfaisant les exigences de sécurité et d'efficacité, inhérentes au domaine du transport aérien. Le problème d'ordonnancement des atterrissages d'avions, formulé pour la première fois dans les années 1970 [22, 61], consiste à trouver, pour les avions se présentant à un aéroport, la séquence et les heures d'atterrissage optimisant un critère de performance (par exemple, maximiser le taux d'atterrissage, ou minimiser le retard total) tout en respectant des contraintes de sécurité. Même si ce problème d'optimisation est, à l'évidence, de nature dynamique et stochastique, seule l'étude du cas statique et déterministe, où toutes les données d'entrée sont complètes et connues avec certitude, a été approfondie dans la littérature. D'autre part, dans le cadre des projets, européen et américain, de modernisation des systèmes d'aide à la décision pour le séquencement des atterrissages d'avions, il a été proposé de prendre en considération les avions quelques heures en amont, afin de commencer l'ordonnancement plus tôt et diminuer la congestion de l'espace aérien autour des grands aéroports. De cette nouveauté opérationnelle, émerge une nouvelle variante du problème des atterrissages d'avions, où l'horizon opérationnel est étendu à quelques heures, et où les données sont inévitablement entachées d'incertitude : le problème d'ordonnancement étendu des arrivées d'avions sous incertitude.

Dans le cadre de cette thèse, ce problème d'optimisation est étudié selon le paradigme de la programmation stochastique à deux étapes. Ce paradigme distingue deux étapes de décision entre lesquelles des données incertaines, supposées suivre des lois de probabilité connues, sont révélées. Pour le problème d'intérêt, la structure en deux étapes de décision se décline comme suit. Rappelons, au préalable, que l'environnement opérationnel considéré principalement dans cette thèse est constitué d'une piste précédée par un seul point de début d'approche (IAF). Dans la première étape de décision, les avions sont considérés à 2–3 heures avant leur atterrissage prévu. Il est question de les ordonnancer à l'IAF, en déterminant une séquence et des heures cibles, respectant les contraintes de séparation et les fenêtres de temps à l'IAF. Une exigence supplémentaire est formulée : la séquence cible à l'IAF doit être identique à la séquence cible à l'atterrissage. Ainsi, la fonction objectif de la première étape est la longueur de la séquence cible à l'atterrissage (à minimiser), calculée comme la somme des séparations minimales au seuil de piste entre les paires d'avions successifs. L'incertitude correspond aux heures effectives de passage à

l'IAF qui dévient des heures cibles en suivant des lois de probabilité connues à l'avance.

Dans la deuxième étape, les avions sont supposées être si proches de l'IAF que leurs heures effectives de passage à l'IAF puissent être connues / prédites avec certitude. À cette étape, il s'agit de calculer des heures cibles d'atterrissage, selon la séquence cible prédéterminée en première étape. Les contraintes opérationnelles sont celles de la séparation au seuil de piste et des fenêtres de temps à l'atterrissage. La fonction objectif de deuxième étape peut correspondre à plusieurs points de vue, tels que celui des contrôleurs aériens (par exemple, la minimisation de la charge de travail liée à l'implémentation de la séquence et des heures cibles d'atterrissage) ou celui des compagnies aériennes (par exemple, la minimisation des coûts de déviation par rapport à l'heure d'atterrissage non contrainte).

La première contribution de cette thèse correspond à une étude numérique menée afin de démontrer la faisabilité du concept opérationnel d'ordonnancement étendu des arrivées d'avions sous incertitude. Les variables aléatoires étant considérées comme des variables indépendantes et identiquement distribuées selon une loi normale, la méthode d'approximation par moyenne empirique (SAA) est utilisée pour fournir une version approchée traitable du programme stochastique proposé. L'un des paramètres étudiés est le nombre suffisant de scénarios garantissant une bonne approximation du programme stochastique d'origine. Également, les effets de plusieurs caractéristiques des instances du problème, telles que la largeur des fenêtres de temps à l'IAF et l'amplitude de l'incertitude, sont évalués. Ainsi, il a été remarqué que des fenêtres de temps larges à l'IAF permettent d'obtenir des résultats de meilleure qualité, même si elles requièrent un plus grand nombre de scénarios, et un temps de calcul plus long, pour bien approcher le programme stochastique d'origine. Toutefois, pour une implémentation en contexte opérationnel où le budget de temps de calcul est limité, il est préférable de choisir des fenêtres de temps restreintes à l'IAF. Nous avons également observé, par simulation, que la solution stochastique domine ses homologues déterministes, basées sur la politique "premier arrivé, premier servi", en termes de nombre moyen de conflits à l'IAF, de taux moyen d'atterrissage, et de retard moyen à absorber dans la zone terminale.

La deuxième contribution consiste en la modélisation comme un programme stochastique à deux étapes avec des variables mixtes et des contraintes en probabilité en première étape. Ces contraintes en probabilité permettent de se protéger, à l'avance, contre la perte de la séparation à l'IAF, suite à la révélation de l'incertitude. L'hypothèse de variables aléatoires indépendantes et identiquement distribuées selon la loi normale permet de reformuler les contraintes en probabilité comme des contraintes de séparation à l'IAF, où le minimum de séparation dépend du niveau de tolérance, déterminé par le décideur, de la perte de séparation à l'IAF. Une fonction objectif convexe linéaire par morceaux générique est proposée pour le problème de deuxième étape. Elle peut représenter différents points de vue, selon l'acteur considéré (les contrôleurs aériens utilisant AMAN ou les compagnies aériennes). Cette fonction objectif est également linéarisée. Finalement, l'espérance de la valeur optimale de la deuxième étape, apparaissant dans la fonction objectif du programme stochastique complet, est approchée par une moyenne empirique sur un échantillon donné, dans le cadre de la méthode SAA. Le programme mathématique résultant des linéarisations sus-mentionnées se présente comme un programme linéaire mixte, potentiellement de grande taille, avec un partitionnement évident en un problème maître et des sous-problèmes, correspondant chacun à un scénario de deuxième étape.

Cette structure se prête bien à une résolution par décomposition de Benders. Une étude numérique, sur un ensemble représentatif d'instances, montre l'intérêt de l'approche par programmation stochastique enrichie par des contraintes en probabilité, grâce au calcul de la valeur de la solution stochastique. Finalement, plusieurs méthodes de résolution, exploitant l'algorithme de Branch-and-Cut et de la décomposition de Benders automatique du solveur CPLEX, ont été comparées. Il en est déduit qu'une décomposition de Benders avec une agrégation partielle des sous-problèmes peut surpasser les approches plus classiques, y compris la décomposition de Benders désagrégée.

En guise de troisième contribution, le modèle de programmation stochastique à deux étapes, préalablement proposé, est mis à jour afin de correspondre à l'environnement opérationnel plus réaliste constitué d'une piste d'atterrissage précédée par plusieurs IAF. Deux variantes sont modélisées, selon que le choix de l'IAF est fixé ou qu'il est considéré comme une variable de décision de première étape.

En perspectives de cette thèse, plusieurs axes de recherche sont envisagés. Tout d'abord, en termes de modélisation du problème opérationnel, il est important d'étudier le cas dynamique plus réaliste, où le flux d'arrivée des avions est continu. Une approche possible est d'implanter le modèle proposé pour le cas statique dans un algorithme d'horizon roulant. Également, le phénomène d'apparition inattendue d'avions (appelés *pop-up flights*) dans l'horizon opérationnel, pointé par les développeurs de AMAN comme un phénomène perturbateur de l'ordonnancement, est, à notre connaissance, non encore modélisé dans la littérature spécifique à l'ordonnancement des atterrissages. Toutefois, cette problématique s'inscrit dans la thématique plus générale de l'ordonnancement stochastique et dynamique de tâches pour laquelle des travaux de recherche sont déjà publiés.

En termes de modélisation de l'incertitude dans cette thèse, les variables aléatoires ont été considérées indépendantes et identiquement distribuées selon une loi normale. Il est plus réaliste de considérer une structure de dépendance, dans laquelle les avions traversant les mêmes secteurs aériens à des heures proches ont des retards corrélés. Toujours est-il que la quantification de l'incertitude ainsi que la structure de dépendance est l'une des difficultés à traiter.

Concernant la génération de scénarios, cette thèse a employé la technique d'échantillonnage Monte Carlo. Cependant, d'autres techniques, plus avancées, de génération de scénarios existent dans la littérature, et qui pourraient fournir des échantillons de taille plus réduite, tout en conservant la même représentativité de l'incertitude. L'emploi de telles techniques peut donc donner lieu à des problèmes d'approximation de plus petite taille et donc plus faciles à résoudre, sans pour autant perdre la qualité de l'approximation.

En termes de méthodes de résolution du programme stochastique à deux étapes proposé dans cette thèse, certes, la décomposition automatique de Benders, implémentée dans le solveur CPLEX, a montré une très bonne performance pour les instances considérées. Néanmoins, l'implémentation d'une décomposition de Benders avec des techniques d'accélération ad hoc pourrait donner lieu à un algorithme plus performant, ouvrant la voie à la résolution d'instances de plus grande taille. Nous citons, à titre d'exemple, la résolution des sous-problèmes de Benders par programmation dynamique, inspirée de

Faye [24], qui est en cours d'implémentation.

Finalement, dans cette thèse, le paradigme de choix pour la prise en compte de l'incertitude était la programmation stochastique à deux étapes. Cela ne réduit en rien le potentiel d'autres paradigmes d'optimisation sous incertitude, tels que la programmation robuste ou l'optimisation en ligne (*online*), pour traiter les problématiques de la gestion du trafic aérien.

Annexe A

Introduction à la gestion de trafic aérien

A.1 Gestion de trafic aérien

Le besoin de la gestion du trafic aérien émane de la croissance du secteur de l'aviation au cours du siècle dernier. Cette croissance a été accompagnée d'une augmentation des risques de sécurité et des soucis de performance et plus récemment de l'impact environnemental.

Pour répondre à ces enjeux, les premiers systèmes de gestion de trafic aérien ont vu le jour pendant la deuxième guerre mondiale. Ces systèmes s'organisent aujourd'hui autour de trois missions :

1. la gestion de l'espace aérien (*Air Space Management - ASM*)
2. la gestion et l'optimisation des flux et de la capacité (*Air Traffic Flow and Capacity Management - ATFCM*)
3. les services de la circulation aérienne (*Air Traffic Services - ATS*)

   Dans la suite, nous expliquons brièvement chacune de ces trois missions.

Gestion de l'espace aérien Un pré-requis à la gestion de trafic aérien est d'organiser l'espace aérien en volumes appelés *secteurs aériens*. Un secteur est un volume tridimensionnel défini par un contour, une altitude inférieure et une altitude supérieure. Chaque secteur appartient à une *classe* de l'espace aérien déterminant le rendu de service de contrôle dans ce secteur.

Gestion et optimisation des flux aériens Sur le plan stratégique, les organismes [1] responsables de la gestion de flux de trafic aérien ont pour mission de réguler les flux de trafic en fonction des capacités des secteurs et des aéroports afin d'assurer la sécurité et la fluidité du trafic. Cette régulation commence plusieurs mois et continue jusqu'à quelques jours avant la date prévue d'un vol.

1. En Europe, la gestion de flux de trafic aérien est centralisée et assurée par l'unité de gestion des capacités et des flux (*Capacity and Flow Management Unit - CFMU* rebaptisée *Network Manager Operations Center - NMOC*)

Services de la navigation aérienne Les services de la navigation aérienne sont délivrés sur le plan tactique, commençant plusieurs minutes avant le décollage, et se poursuivant jusqu'à l'atterrissage. Ces services sont divisés en trois groupes : service d'information de vol, service d'alerte et service de contrôle aérien (*Air Traffic Control - ATC*).

Service de contrôle aérien La mission du contrôle aérien consiste à détecter et éviter les abordages entre les aéronefs dans les espaces aériens contrôlés. Le contrôle de trafic aérien est confié à des organismes dédiés de la navigation aérienne fournissant différents niveaux du service de contrôle selon les classes d'espace aérien. Ces organismes sont principalement les centres de contrôles régionaux, les centres de contrôle en approche et les tours de contrôle. Un secteur bénéficiant du service de contrôle aérien est appelé *secteur contrôlé*.

Sur le plan tactique, la sectorisation de l'espace aérien n'est pas remise en question. Seuls le contrôle de trafic aérien et dans une moindre mesure la gestion de flux font partie du périmètre d'étude du problème d'ordonnancement d'avions en atterrissage et/ou en décollage.

Dans la suite, pour mieux illustrer la problématique, nous définissons les principales phases d'un vol. Nous décrivons, par la suite, les espaces aériens et les organismes de contrôle associés.

A.2 Phases de vol

Un vol se passe en cinq phases principales résumées dans la suite.

Décollage : cette phase commence quand l'avion est à la position de parking et est marquée principalement par le moment où il quitte la piste de décollage et commence sa montée. La procédure de décollage est détaillée dans l'annexe B.1. Pendant le décollage, le pilote communique avec la tour de contrôle jusqu'à quelques centaines de mètres d'altitude.

Montée : l'avion monte jusqu'à son niveau de vol [2] de croisière. Il suit généralement un itinéraire de départ prédéfini, propre à l'aéroport de départ, appelé *Standard Instrument Departure - SID* [3]. La fin de la montée est marquée par un point fictif désigné *Top-Of-Climb*. Au cours de la montée, le pilote communique d'abord avec les contrôleurs d'approche. Ensuite, la communication est transférée avec les contrôleurs de l'en-route.

En-route : Une fois arrivé à son niveau de vol de croisière, l'avion maintient son niveau de vol ainsi que sa vitesse. Dans le plan horizontal, il suit des routes aériennes, marquées par des balises fictives, appelées *waypoints*, indiquées initialement dans son plan de vol et éventuellement modifiées par les contrôleurs. Au cours de cette phase d'en-route, l'avion évolue d'un secteur aérien à un autre, et passe par conséquent de la responsabilité d'un centre de contrôle en route à un autre [4].

2. Altitude exprimée par rapport à un niveau standardisé de la mer.

3. Un aéroport possède généralement plusieurs SID, adaptés aux performances des avions et conçus de façon à éviter les obstacles environnants, à minimiser le bruit, etc.

4. Tant que les avions se trouvent proches ou dans un espace aérien continental et que les secteurs aériens traversés sont des secteurs contrôlés.

Le dernier point de cette phase de croisière est appelé *Top-Of-Descent*, marquant le début de la phase de descente.

Descente : À la fin de la phase de croisière, l'avion suit généralement un itinéraire normalisé d'arrivée, propre à l'aéroport de destination, appelé *Standard Terminal Arrival Route - STAR*[5]. L'avion dégrade son niveau de vol et sa vitesse jusqu'à atteindre l'IAF, qui est le point d'entrée à la TMA. À partir de l'IAF commence la procédure d'approche où l'avion est guidé par les contrôleurs d'approche afin de se préparer à l'atterrissage. La procédure d'approche est détaillée dans l'annexe B.2.

Atterrissage : Cette phase est marquée par le moment où l'avion rejoint le sol au niveau de la piste d'atterrissage. Une fois que l'avion est posé et que la piste est dégagée, l'atterrissage est considéré comme réussi. L'avion poursuit son mouvement au sol à travers les voies de circulation jusqu'à sa position de parking.

Pendant les différents phases de vol, la responsabilité de la sécurité d'un vol passe d'un organisme de contrôle à un autre selon les espaces aériens franchis, comme schématisé dans la figure A.1. Ces espaces sont présentés dans le prochain paragraphe.

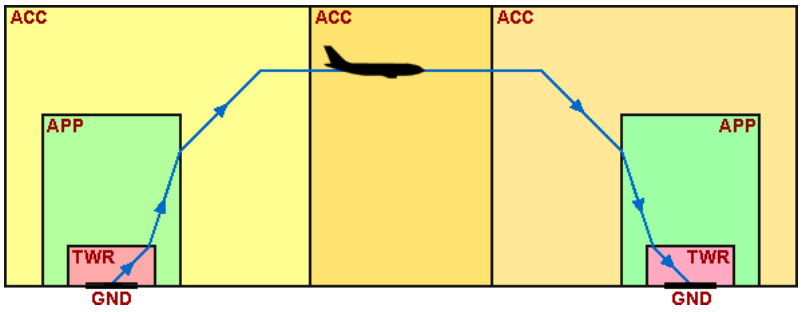


FIGURE A.1 – Organismes de contrôle selon les phases de vol

A.3 Espaces aériens et organismes de contrôle associés

Les principaux organismes de contrôle aérien sont la tour de contrôle, le centre de contrôle en approche et le centre de contrôle régional. Ils sont respectivement associés aux espaces aériens suivants : zone de contrôle, région de contrôle et région d'information de vol (*Flight Information Region - FIR*).

A.3.1 Zone de contrôle - CTR

La zone de contrôle (*ConTrolled aRea - CTR*) est l'espace aérien contrôlé s'étendant verticalement à partir de la surface jusqu'à une limite supérieure spécifiée (environ 500 m), englobant un ou plusieurs aéroports. Elle sert à protéger les trajectoires d'approche finale et de montée initiale. Une zone de contrôle est de forme cylindrique et avec quelques kilomètres de diamètres. L'organisme de contrôle associé est la tour de contrôle (*control ToWeR - TWR*). Elle a la responsabilité de donner les autorisations nécessaires, appelées *clairances*, d'emprunter les taxiways et d'utiliser les pistes en atterrissage et en décollage.

5. Un aéroport possède généralement plusieurs STARs. Ils sont conçus de manière stratégique.

A.3.2 Région de contrôle - TMA

L'espace aérien qui chapeaute une ou plusieurs zones de contrôle est appelé région de contrôle terminale (*Terminal Maneuvring Area - TMA*). Il est du ressort du contrôle en approche (**APP**) de garantir la sécurité des avions évoluant dans une TMA : avions en approche et avions en montée. Une TMA est cylindrique aussi avec un rayon d'environ cinquante miles nautiques. Quant à sa limite verticale supérieure, elle s'étend jusqu'à 6 km.

A.3.3 Région d'information de vol - FIR

Pendant la phase de vol en-route, tant que l'avion évolue dans des secteurs contrôlés de l'espace aérien inférieur [6], il bénéficie du service de contrôle délivré par les centres de contrôle régionaux (**CCR**) successifs par lesquelles il passe. En France, il existe 5 centres de contrôle régionaux appelés Centres Régionaux de Navigation Aérienne - CRNA.

A.3.4 TMA Etendue (*Extended TMA - E-TMA*)

Une TMA étendue est une TMA dont le diamètre a été augmenté au détriment de l'espace aérien en-route. Le diamètre d'une TMA étendue atteint une centaine de miles nautiques. Le contrôle des avions dans une TMA étendue permet de mieux anticiper le séquencement des avions à l'IAF.

6. de la surface jusqu'à 6 km environ. Au-delà, c'est l'espace aérien dit supérieur.

Annexe B

Procédures de décollage et d'atterrissage

B.1 Procédure de décollage

La procédure de décollage commence à partir de la position de parking de l'aéronef jusqu'au franchissement des 35 pieds à la verticale du seuil de piste. La progression d'un avion dans la procédure de décollage est sujette à l'attribution par les contrôleurs de la tour de contrôle d'autorisations, appelées clairances, à l'avion en question. Cette procédure se décompose en 4 phases :

Repoussage *push-back* : Après avoir effectué les opérations d'escale (débarquement / embarquement des passagers et des bagages, réapprovisionnement en carburant etc), un avion de ligne en position de parking demande la clairance au repoussage pour quitter son point de stationnement.

Roulage : Une fois que l'avion eut quitté son stationnement parking et que les moteurs eurent été allumés, le pilote demande la clairance de roulage, qui l'autorise à suivre des voies de circulation (*taxiways*) jusqu'au point d'arrêt à l'entrée d'une piste de décollage. Les voies de circulation, le point d'arrêt et la piste de décollage sont tous donnés par la tour de contrôle.

Décollage : Le contrôleur autorise l'avion en attente à l'entrée de la piste de s'aligner pour le décollage. L'avion s'aligne alors à la piste de décollage et met la poussée des avions à son maximum. L'avion accélère et à l'atteinte de la vitesse de rotation (V_R), le pilote lève le nez de l'avion. Porté par la force de l'air sous ses ailes, l'avion prend son envol et entame sa montée.

Montée initiale : Dès que l'avion dépasse la hauteur de 35 pieds à la vitesse de sécurité (V_2), le décollage est considéré comme réussi. L'avion poursuit sa montée généralement en suivant une SID indiquée par la tour de contrôle.

B.2 Procédure d'approche

La dernière portion de la phase d'en-route d'un vol finit par une STAR se terminant au point d'entrée de l'espace aérien en approche (TMA), appelé repère d'approche initiale (*Initial Approach Fix - IAF*). Au niveau de l'IAF, un aéronef peut être maintenu par les contrôleurs aériens dans des circuits d'attente appelés hippodromes d'attente ou "stacks"

dans le jargon des contrôleurs aériens.

Une heure d'approche prévue (*Expected Approach time - EAT*) est estimée pour chaque aéronef. Elle correspond à l'heure de passage de l'aéronef à l'IAF ou de sortie du "stack" s'il y était maintenu. Un temps estimé d'atterrissage (*Estimated Landing Time - ELT*) peut être calculé aussi en estimant la durée de l'approche à partir de l'heure d'approche prévue.

Dans le cas de trafic de faible densité, il n'y a pas besoin de gérer les circuits d'attente ni de recalculer les heures d'approche prévues.

Les contrôleurs peuvent recourir au guidage radar pour amener l'aéronef jusqu'à un point à partir duquel le pilote peut exécuter l'approche finale. Le guidage radar consiste à donner les instructions de cap (direction en degrés dans le plan horizontal), de vitesse et d'altitude nécessaire pour guider l'avion.

Souvent dans tous les cas, une approche est constituée de trois segments délimités par des repères spécifiques :

Approche initiale : de l'IAF à l'IF (*Intermediate Fix*)

Approche intermédiaire : de l'IF au FAF (*Final Approach Fix*).

Les étapes de l'approche initiale et intermédiaire servent à amener l'avion de la frontière de la TMA (précisément d'un IAF) jusqu'à s'aligner selon l'axe final de la piste d'atterrissage (au niveau du FAF). Pour cela, le contrôleur d'approche peut soit demander à l'avion de suivre une trajectoire d'approche standardisée soit le guider pour un contrôle plus fin de sa trajectoire.

Afin d'aider les avions à intercepter avec précision l'axe de la piste, un moyen aéroportuaire de radio-navigation, appelé système d'atterrissage aux instruments (*Instrument Landing System - ILS*), fournit un plan horizontal et un plan vertical dont l'intersection définit l'axe de descente pour rejoindre correctement la piste.

Pendant l'approche intermédiaire, l'aéronef maintient un niveau de vol et une vitesse stables.

Approche finale : du FAF au seuil de piste si l'avion finit son atterrissage et sinon du FAF à un point aérien spécifique appelé *Missed Approach Point - MAPT*.

La distance entre le FAF et le seuil de piste est spécifique à l'aéroport et à la piste. Elle est typiquement de l'ordre de quelques miles nautiques. Pendant l'approche finale, l'avion adopte une vitesse qui lui est particulière, appelée la vitesse d'approche, et descend selon les plans donnés par l'ILS jusqu'à une hauteur, dite de décision (Decision Hight - DH). À cette hauteur, le pilote doit décider de poursuivre ou non l'atterrissage. S'il voit le seuil de piste à cette hauteur, le pilote continue l'atterrissage en posant l'avion.

Par contre, si le pilote n'arrive pas à voir le seuil de piste à cette hauteur, il doit interrompre son atterrissage en remettant les gaz. Dans ce cas, l'approche est appelée approche interrompue (API). D'où un quatrième segment :

Approche interrompue : du MAPT à un circuit de remise de gaz.

Le pilote, guidé par le contrôleur, peut retenter l'atterrissage.

Quand le pilote réussit à poser l'avion, et aussitôt que ce dernier touche le sol, le pilote commence à freiner. La distance nécessaire pour le freinage dépend principalement de la masse de l'avion à l'atterrissage et de l'état de la piste (mouillée ou sèche). L'avion se dirige ensuite vers une bretelle désignée pour gagner une voie de circulation et quitter la piste. En fonction du type et de l'emplacement de la bretelle, une vitesse maximale

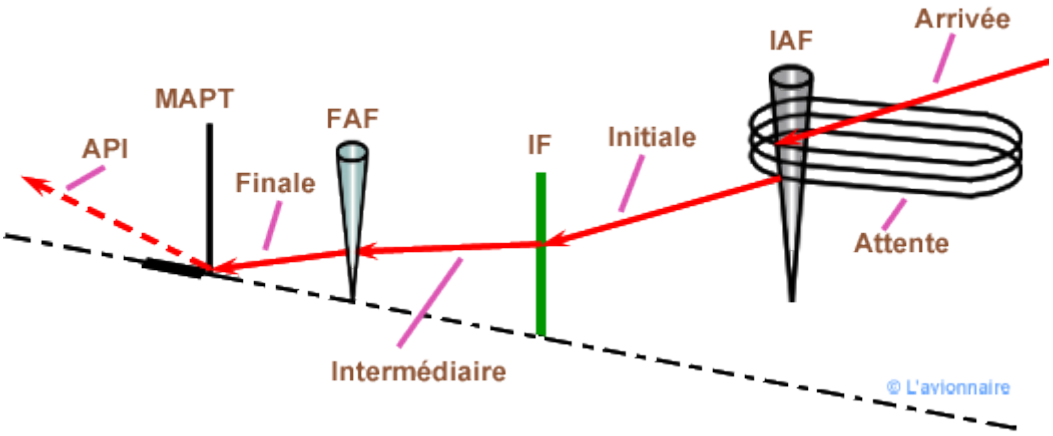


FIGURE B.1 – Segments et repères de la procédure d'approche

de dégagement de piste doit être respectée. Cette vitesse maximale ainsi que l'état de la piste et les caractéristiques de l'avion en question influent sur le temps d'occupation de la piste.

Quand l'avion dégage la piste, l'atterrissage est considéré comme réussi. Enfin, l'avion se dirige vers la porte de débarquement en suivant les taxiways.

Bibliographie

[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] Ioannis Anagnostakis and John-Paul Clarke. Runway operations planning : a two-stage solution methodology. In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*. IEEE, 2003.

[3] 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.

[4] Jonathan F. Bard. *Practical bilevel optimization : Algorithms and applications*, volume 30. Springer Science & Business Media, 2013.

[5] 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.

[6] Christopher Beck, Patrick Prosser, and Evgeny Selensky. Vehicle routing and job shop scheduling : What's the difference ? In *Proceedings of the 13th International Conference on Automated Planning and Scheduling*, pages 267–276, 2003.

[7] Jacques F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1) :238–252, 1962. doi : 10.1007/BF01386316.

[8] 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.

[9] 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.

[10] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B. Shah. Julia : A fresh approach to numerical computing. *SIAM Review*, 59(1) :65–98, 2017. doi : 10.1137/141000671.

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

[12] Pierre Bonami, Domenico Salvagnin, and Andrea Tramontani. Implementing automatic Benders decomposition in a modern MIP solver. In Daniel Bienstock and Giacomo Zambelli, editors, *Integer Programming and Combinatorial Optimization*, pages 78–90, Cham, 2020. Springer International Publishing. doi : 10.1007/978-3-030-45771-6_7.

[13] Christabelle Bosson, Min Xue, and Shannon Zelinski. Optimizing integrated arrival, departure and surface operations under uncertainty. In *Proceedings of the 10th USA/Europe ATM Research and Development Seminar*, 2015.

[14] 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.

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

[16] Gregory C. Carr, Heinz Erzberger, and Frank Neuman. Fast-time study of airline-influenced arrival sequencing and scheduling. *Journal of Guidance, Control, and Dynamics*, 23(3) :526–531, 2000.

[17] 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.

[18] 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.

[19] Andrew J. Cook and Graham Tanner. European airline delay cost reference values. Technical report, EUROCONTROL Performance Review Unit, 2015.

[20] Teodor G. Crainic, Mike Hewitt, Francesca Maggioni, and Walter Rei. Partial Benders decomposition strategies for two-stage stochastic integer programs. Technical Report 2016-37, CIRRELT, Université de Québec à Montréal, Canada, 2016.

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

[22] 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.

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

[24] 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.

[25] 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.

[26] 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`.

[27] 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.

[28] Michael J. Frankovich. *Air traffic flow management at airports : A unified optimization approach.* PhD thesis, Massachusetts Institute of Technology, 2012.

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

[30] 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.

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

[32] Emanuele Garone, Jean-François Determe, and Roberto Naldi. Generalized traveling salesman problem for carrier-vehicle systems. *Journal of Guidance, Control, and Dynamics*, 37(3) :766–774, 2014.

[33] Ahmed Ghoniem, Farbod Farhadi, and Mohammad Reihaneh. An accelerated branch-and-price algorithm for multiple-runway aircraft sequencing problems. *European Journal of Operational Research*, 246(1) :34–43, 2015. doi : 10.1016/j.ejor.2015.04.019.

[34] Ronald L. Graham, Eugene L. Lawler, Jan K. Lenstra, and Alexander H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling : A survey. *Annals of Discrete Mathematics*, 5 :287–326, 1979.

[35] Julien Guepet. *Optimisation de la gestion des avions dans un aéroport : affectation aux points de stationnement, routage au sol et ordonnancement à la piste.* PhD thesis, Institut Polytechnique de Grenoble, France, 2015.

[36] Nathalie Hasevoets and Paul Conroy. Arrival Manager - Implementation Guidelines and Lessons Learned. Technical report, EUROCONTROL, 2010. URL `https://www.skybrary.aero/bookshelf/books/2416.pdf`.

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

[38] 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.

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

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

[41] SESAR JU. Extended Arrival Manager (E-AMAN) Frequently Asked Questions, 2015. URL `http://www.sesarju.eu/sites/default/files/documents/wac2015/E-aman_factsheet_FINAL.pdf`.

[42] 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.

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

[44] 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.

[45] Ahmed Khassiba, Fabian Bastin, Sonia Cafieri, Bernard Gendron, and Marcel Mongeau. Extended aircraft arrival management under uncertainty : A chance-constrained two-stage mixed-integer programming model. Technical Report 2018-55, CIRRELT, Université de Montréal, Canada, 2018.

[46] 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.

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

[48] 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.

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

[50] Hanbong Lee and Hamsa Balakrishnan. A study of tradeoffs in scheduling terminal-area operations. *Proceedings of the IEEE*, 96(12) :2081–2095, 2008. doi : 10.1109/JPROC.2008.2006145.

[51] Alexander Lieder and Raik Stolletz. Scheduling aircraft take-offs and landings on interdependent and heterogeneous runways. *Transportation Research Part E*, 88 : 167–188, 2016. doi : 10.1016/j.tre.2016.01.015.

[52] Alexander Lieder, Dirk Briskorn, and Raik Stolletz. A dynamic programming approach for the aircraft landing problem with aircraft classes. *European Journal of Operational Research*, 243(1) :61–69, 2015.

[53] Thomas L. Magnanti and Richard T. Wong. Accelerating Benders decomposition : Algorithmic enhancement and model selection criteria. *Operations Research*, 29(3) : 464–484, 1981. doi : 10.2307/170108.

[54] 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.

[55] 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. doi : https://doi.org/10.2514/6.2005-7376.

[56] 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.

[57] 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.

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

[59] HARRIS ORTHOGON. Extended AMAN enabling sesar deployment, 2016. URL `http://www.harris-orthogon.com/wp-content/uploads/Harris-Orthogon_OSYRIS_E-AMAN_Product.pdf`.

[60] Michael L. Pinedo. *Scheduling : Theory, algorithms, and systems*. Springer Science & Business Media, 2012.

[61] Harilaos N. Psaraftis. A dynamic programming approach to the aircraft sequencing problem. Technical report, Massachusetts Institute of Technology, 1978.

[62] 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.

[63] 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.

[64] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on stochastic programming : Modeling and theory*. SIAM, 2009.

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

[66] 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.

[67] 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.

[68] Maarten Soomer and Geert-Jan Franx. Scheduling aircraft landings using airlines' preferences. *European Journal of Operational Research*, 190(1) :277–291, 2008.

[69] Laurel Stell. Prediction of top of descent location for idle-thrust descents. In *Proceedings of the 9th USA/Europe Air Traffic Management Research and Development Seminar*, 2011.

[70] 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.

[71] Richard M. Van Slyke and Roger Wets. L-Shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4) :638–663, 1969. doi : 10.1137/0117061.

[72] 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.

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

[74] Min Xue and Shannon Zelinski. Dynamic stochastic scheduler for integrated arrivals and departures. In *Proceedings of the 33rd Digital Avionics Systems Conference*. IEEE, 2014.

[75] Min Xue and Shannon Zelinski. A stochastic scheduler for integrated arrival, departure, and surface operations in Los Angeles. In *Proceedings of the 15th Aviation Technology, Integration, and Operations Conference*. AIAA, 2015. doi : 10.2514/6.2015-2273.