# Detecting and solving aircraft conflicts using bilevel programming

Martina Cerulli, Claudia d'Ambrosio, Leo Liberti, Mercedes Pelegrín

**Noname manuscript No.**
(will be inserted by the editor)

# Detecting and solving aircraft conflicts using bilevel programming

**Martina Cerulli, Claudia D'Ambrosio,
Leo Liberti, Mercedes Pelegrín**

the date of receipt and acceptance should be inserted later

**Abstract** We present two bilevel programming formulations for the aircraft deconfliction problem: one based on speed regulation in $k$ dimensions, the other on heading angle changes in 2 dimensions. We propose three reformulations of each problem based on KKT conditions and on two different duals of the lower-level subproblems. We also propose a cut generation algorithm to solve the bilevel formulations. Finally, we present computational results on a variety of instances.

**Keywords** mathematical programming · deconfliction · Air Traffic Management · cutting plane · bilevel programming

## 1 Introduction

Two aircraft are said to be in conflict if their relative distance is less than a given safety threshold. By *aircraft deconfliction* we mean the set of strategies for detecting and solving conflicts among flying aircraft. A growing effort is dedicated to automate its optimal management, which is currently still widely performed on the ground by (human) Air Traffic Controllers (ATC). The development of urban air mobility will also rely on such decision-making support tools. Looking at a certain restricted airspace on a radar screen, the human controllers give real-time instructions to the pilots, based essentially on the change of their trajectories.

This research was partly funded by the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement n. 764759 ETN "MINOA".
This publication was supported by the Chair "Integrated Urban Mobility", backed by L'X - École Polytechnique and La Fondation de l'École polytechnique and sponsored by Uber.

Martina Cerulli · Claudia D'Ambrosio · Leo Liberti · Mercedes Pelegrín
LIX CNRS, École Polytechnique, Institut Polytechnique de Paris, 91128 Palaiseau France
E-mail: {mcerulli,dambrosio,liberti,pelegringarcia}@lix.polytechnique.fr

Many strategies can be used to avoid conflicts. In this paper, we focus on heading angle and speed changes. While heading angle changes (HAC) are often used by ATC to prevent collisions, speed changes are almost never performed in practice because of the tight speed modification restrictions imposed by air travel regulations. There are several reasons for the strict bounds, which include aircraft dynamics, passengers' comfort and the real-time nature of the decision process needed to make this maneuver efficient. In 2004, however, the concept of *Subliminal Control* was introduced in the context of the European project ERASMUS [1]. Subliminal speed control consists in allowing minor speed adjustments that have to be small enough to remain imperceptible to ATC, thus reducing their workloads. During the ERASMUS project, the efficiency of this method was validated through human-in-the-loop experiments by Drogoul et al. [2], who proposed two speed modulation ranges: a weak one from -6% to +3% and a strong one from -12% to +6%. Nowadays, the European Union is working towards implementing these types of approaches through the Single European Sky ATM Research (SESAR) project [3].

When we consider aircraft moving in a three-dimensional space, the need for subliminal speed changes becomes less relevant: speed regulation (SR) is not realistically performed while changing altitude, but only when aircraft are flying within a fixed altitude layer. This does not apply to Unmanned Aerial Vehicles (UAVs), however, which have different dynamics. As discussed in [4], the minimum distance between UAVs can be guaranteed by modifying their 3D trajectories, including by SR methods.

In this paper, we present different formulations of the Aircraft Deconfliction Problem (ADP) based on both HAC and SR. We present different bilevel programming formulations for each case, with several corresponding reformulations. Our formulations can also apply to other problems, such as coordinating a fleet of robots or unmanned vehicles in a complex obstacle-ridden region [5]. This includes, among others, the scenario of automated material handling vehicles moving on fixed routes in warehouses or production plants, to transport raw materials or perform tasks in production processes [6].

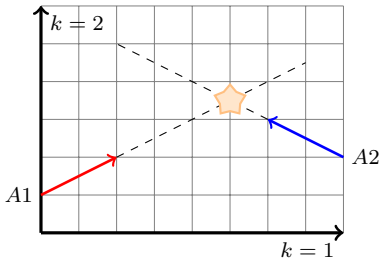


Fig. 1: Two conflicting aircraft in 2 dimensions

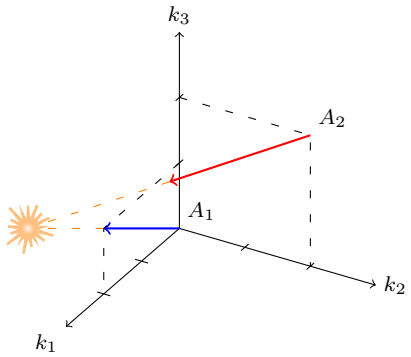


Fig. 2: Two conflicting aircraft in 3 dimensions

This paper considerably extends [7], where SR was applied to aircraft at the same flight level. This implies that the optimization takes place in the plane (see Figure 1). Here, we propose a more general approach by modeling the ADP through SR in a $k$-dimensional airspace (see Figure 2), an additional single-level reformulation based on the Wolfe dual, as well as a new bilevel formulation of the ADP based on the HAC strategy in the plane. Moreover, we extend the Cut Generation (CG) algorithm in [7] to the HAC based formulation, and provide a new analysis thereof.

The rest of the paper is organized as follows. We review the relevant literature in Section 2. In Section 3 we introduce our SR based ADP (SRADP) formulations: a natural formulation and a bilevel formulation, with three single-level reformulations. In Section 4 we introduce a bilevel formulation of the HAC based ADP (HACADP), and its reformulations. A CG solution algorithm for both SRADP and HACADP is presented in Section 5. In Section 6 we discuss computational results. Some concluding comments are given in Section 7.

## 2 Literature review

There exists a wide range of approaches for modeling and solving the ADP. Here, we give a non-exhaustive review on those related works proposing mathematical programming formulations based on SR, HAC, or both.

SR is one of the most common strategies for aircraft deconfliction by Mathematical Programming. In [8], speed is converted to travel time in order to minimize the total cost of all potential conflicts: the cost of a conflict depends on the time two aircraft spend travelling at a distance below the security threshold, since this time is proportional to the ATC monitoring and conflict solution effort. The decision variables of the proposed Mixed Integer Linear Program (MILP) are the arrival times at the different intersection points of the trajectories in the considered time horizon. An equity-oriented conflict resolution (ECR) model, based on these same variables, is introduced in [9]. It

proposes an innovative aircraft collision avoidance model promoting equitable solutions (airlines are equally affected by the trajectory adjustments). The ECR model combines three optimization stages, which can be formulated as MILP and attempt respectively to: maximize the number of solved conflicts; resolve conflicts in the fairest way; reduce the delay induced by the trajectory changes. A different kind of approach is proposed in [10,11], where Mixed Integer Nonlinear Programming (MINLP) formulations for the SRADP in the plane are considered. Specifically, [10] proposes a heuristic algorithm that decomposes the problem into smaller subproblems. The exact solutions of the subproblems are then combined to form a globally feasible but possibly sub-optimal solution of the original problem. A feasibility pump heuristic is proposed in [11]. This algorithm builds two sequences of points: one consisting of points that are feasible w.r.t. nonlinear constraints, and the other consisting of points satisfying the integrality conditions. The algorithm iterates until the two sequences converge to a feasible solution of the MINLP.

SR fails to solve frontal conflicts; moreover, it may not be sufficient to ensure safety if speed bounds are tight. Consequently, it is usually combined with other maneuvers, such as flight level reallocation. For instance, the authors of [12] present a MILP formulation where conflict situations are avoided by performing both speed and altitude changes over predefined routes. The objective is to minimize the expected fuel costs of the aircraft. Binary variables are used to assign flight levels, which indicate whether two aircraft fly at different altitudes, as well as allowing deconfliction of aircraft traversing the same flight level. A multi-objective MILP approach in a similar vein, based on both maneuver types, and aiming at an equitable distribution of the maneuvers over the aircraft, is proposed in [13]. In [14], two disjunctive formulations are proposed for the ADP based on speed and altitude changes. Their objective functions penalize the number of changes linearly or quadratically, giving rise to a MILP or a Mixed Integer Quadratic Program, respectively.

A part of the literature focuses on the geometric characterization of conflicts. These are then used in SR or HAC based models. This is, for example, the case of [15], where the geometric characteristics of aircraft trajectories are used in order to obtain closed-form expressions for single planar conflicts, based on SR and HAC alone, as well as closed-form expressions yielding minimum deviations from the original trajectories with combined SR and HAC. The authors of [16] present a geometric analysis of the conflicts leading to two MILPs: one for SR and another for HAC. The resulting separation constraints are linear on speeds and heading angles, respectively.

Other works interpret maneuvers as a combination of SR and HAC. In [17], SR and HAC are applied sequentially. First, a MINLP formulation minimizing HAC and subject to the safety separation condition is presented. Then, another MINLP is proposed which maximizes the number of collisions avoided by SR. These two models are solved sequentially. Then, using a two-step methodology, the solution of the SR MINLP is used as a pre-processing step for the HAC MINLP. SR and HAC are sometimes combined in the same formulation. In [18], the planar ADP is formulated as a nonconvex Quadratically Con-

strained Quadratic Program (QCQP) where the objective function minimizes the deviations from the original velocity vector. If the solution of the "natural" Semidefinite Programming relaxation of this QCQP has rank one, then the problem is solved; otherwise, a locally optimal and conflict-free solution with a certain crossing pattern can be obtained via a stochastic rank reduction procedure. A different approach, which also combines SR and HAC to find optimal aircraft maneuvers, is proposed in [19]. In this case, a formulation in complex numbers with disjunctive constraints is introduced; speed bounds are translated into nonconvex quadratic constraints by considering the Euclidean norm of the vectors of velocities; different relaxations of the resulting MINLP are then proposed, solved, and compared.

In this paper, we formulate the SRADP in $k$ dimensions and the HACADP in two dimensions via bilevel programming. To the best of our knowledge, this is the first time that a bilevel approach is used to model the ADP. Our objective is to minimize the changes w.r.t. the original flight plan while still satisfying the safety distance on aircraft pairs. The terminology and symbols, as well as the formulæ aircraft separation, are taken from [10,11,17].

## 3 Aircraft deconfliction via speed regulation

The goal of the approach presented in this section is to minimize the total speed changes needed to satisfy the minimum safety distance for each pair of aircraft in a given time horizon. An important assumption is that changes occur instantaneously and that the new speeds remain constant in the time horizon. Specifically, given a constant speed for every aircraft, our formulation decides new optimal constant speeds satisfying the safety constraints. The sets, parameters, and variables used in all the mathematical formulations in this section are listed below.

- **Sets:**
  - $A = \{1, \ldots, i, \ldots, n\}$ is the set of aircraft flying in a shared airspace;
  - $K = \{1, \ldots, k_{\max}\}$ is the set of dimension indices.
- **Parameters:**
  - $T$ is the length of the time horizon [hours];
  - $d$ is the safety distance between aircraft [Nautical Miles NM] [1];
  - $x^0_{ik}$ is the $k$-th component of the initial position of aircraft $i$;
  - $v_i$ is the initially planned speed of aircraft $i$ [NM/h];
  - $u_{ik}$ is the $k$-th component of the direction of aircraft $i$;
  - $q_i^{\mathsf{min}}$ and $q_i^{\mathsf{max}}$ define the feasible range of the speed modification ratios of aircraft $i$ s.t. $q_i^{\mathsf{min}} < 1 < q_i^{\mathsf{max}}$.
- **Variables:** $q_i$ is the ratio of the implemented speed w.r.t. the initially planned speed of aircraft $i$: $q_i = 1$ if the speed is equal to the initially planned one, $q_i > 1$ if it is increased, $q_i < 1$ if it is decreased.

[1] 1 NM = 1852 m

Note that, having in mind real applications, we consider the upper bound $T$ on the time in our formulations. In fact, human ATCs monitor a portion of the airspace during a fixed time window. Moreover, a finite time horizon is often considered in the literature.

### 3.1 Natural problem formulation

The following provides a "natural" way to formulate SRADP:

$$\min_{q} \quad \sum_{i\in A}(q_i-1)^2 \tag{1a}$$

$$\forall i \in A \quad q_i^{\mathsf{min}} \le q_i \le q_i^{\mathsf{max}} \tag{1b}$$

$$\forall i<j\in A,\ t\in[0,T] \quad \sum_{k\in K}\left[(x_{ik}^0-x_{jk}^0)+t(q_iv_iu_{ik}-q_jv_ju_{jk})\right]^2 \ge d^2. \tag{1c}$$

Formulation (1a)–(1c) is a semi-infinite program, where the last line (Eq. (1c)) contains uncountably many constraints, which ensure aircraft separation. Specifically, Eq. (1c) requires the squared Euclidean distance between each two aircraft $i$ and $j$ to be greater than or equal to $d^2$ at each instant $t$ in the time window $[0,T]$. In these constraints, the $k$-th component of the position vector of aircraft $i$ at time $t$ is defined as $x_{ik}(t)=x_{ik}^0+tq_iv_iu_{ik}$.

The (convex) objective function is the sum of squared aircraft speed changes. This is equivalent to finding the feasible solution with the minimum speed change, which must be in $[q_i^{\mathsf{min}},q_i^{\mathsf{max}}]$ for every aircraft $i$. As mentioned earlier, each aircraft $i$ will start flying with the implemented speed, which is equal to $v_iq_i$.

### 3.2 Bilevel formulation of the problem

Since it is difficult to deal with the uncountably many constraints (1c) of the natural formulation, we propose a bilevel reformulation of SRADP with multiple lower-level subproblems. More details on the connections of semi-infinite and bilevel programming can be found in [20]. For an introduction to bilevel programming, see, for instance, [21].

We first introduce a bilevel formulation with a lower-level subproblem for each pair of aircraft $i<j\in A$:

$$\min_{q,\tau} \quad \sum_{i\in A}(q_i-1)^2 \tag{2a}$$

$$\forall i \in A \quad q_i^{\mathsf{min}} \le q_i \le q_i^{\mathsf{max}} \tag{2b}$$

$$\forall i<j\in A \quad \sum_{k\in K}\left[(x_{ik}^0-x_{jk}^0)+\tau_{ij}(q_iv_iu_{ik}-q_jv_ju_{jk})\right]^2 \ge d^2 \tag{2c}$$

$$\forall i<j\in A \quad \tau_{ij}\in \arg\min_{t_{ij}\in[0,T]}\sum_{k\in K}\left[(x_{ik}^0-x_{jk}^0)+t_{ij}(q_iv_iu_{ik}-q_jv_ju_{jk})\right]^2. \tag{2d}$$

Note that each lower-level subproblem is an optimization problem in the lower-level variables $t_{ij}$, parametrized by the upper-level variables $q_i$ and $q_j$. An optimal solution of each lower-level subproblem, denoted by $\tau_{ij}$, corresponds to the time instant at which aircraft $i$ and $j$ are closest.

Formulation (2a)–(2c) is equivalent to (1a)–(1c) because, if aircraft pairwise separation constraints (constraints (2c)) hold at the time instant $\tau_{ij}$ which is a minimizer of aircraft relative distance (constraints (2d)), it will be true for each time instant in the time horizon $[0, T]$. Next, we reformulate Eq. (2a)–(2c) by means of the *optimal value function* of each lower-level subproblem:

$$\varphi_{ij}(q) := \min_{t_{ij}\in[0,T]} \sum_{k\in K} \left[(x^0_{ik} - x^0_{jk}) + t_{ij}(q_i v_i u_{ik} - q_j v_j u_{jk})\right]^2, \tag{3}$$

which yields the so-called (and equivalent) *optimal value formulation*:

$$\min_{q,\tau} \quad \sum_{i\in A}(q_i - 1)^2 \tag{4a}$$

$$\forall i \in A \quad q_i^{\mathsf{min}} \le q_i \le q_i^{\mathsf{max}} \tag{4b}$$

$$\forall i < j \in A \quad \sum_{k\in K} \left[(x^0_{ik} - x^0_{jk}) + \tau_{ij}(q_i v_i u_{ik} - q_j v_j u_{jk})\right]^2 \le \varphi_{ij}(q) \tag{4c}$$

$$\forall i < j \in A \quad \sum_{k\in K} \left[(x^0_{ik} - x^0_{jk}) + \tau_{ij}(q_i v_i u_{ik} - q_j v_j u_{jk})\right]^2 \ge d^2. \tag{4d}$$

By inspection, Eq. (4c)–(4d) can be replaced by

$$\forall i < j \in A \qquad \varphi_{ij}(q) \ge d^2.$$

Hence, using the definition (3) of $\varphi_{ij}(q)$, we obtain the following bilevel formulation

$$\min_{q,t} \quad \sum_{i\in A}(q_i - 1)^2 \tag{5a}$$

$$\forall i \in A \quad q_i^{\mathsf{min}} \le q_i \le q_i^{\mathsf{max}} \tag{5b}$$

$$\forall i < j \in A \quad \min_{t_{ij}\in[0,T]} \sum_{k\in K} \left[(x^0_{ik} - x^0_{jk}) + t_{ij}(q_i v_i u_{ik} - q_j v_j u_{jk})\right]^2 \ge d^2, \tag{5c}$$

which is an exact reformulation of the semi-infinite formulation in Eq. (1a)–(1c).

### 3.3 KKT reformulation

Certain bilevel programs, notably those having a convex lower-level subproblem, can be easily reformulated to single-level by replacing the lower-level subproblem by its KKT conditions — see [21, Sec. 3.5] and [22] for the specific case of linear bilevel programs. Assuming some regularity condition (e.g. Slater's condition)

holds, this yields a single-level mathematical program with complementarity constraints. Given the subproblem for each $(i, j)$

$$\begin{array}{ll} \min\limits_{t_{ij}} & \sum\limits_{k\in K}\Big[(x^0_{ik}-x^0_{jk})+t_{ij}(q_iv_iu_{ik}-q_jv_ju_{jk})\Big]^2 \\ s.t. & -t_{ij}\le 0 \\ & t_{ij}\le T, \end{array} \qquad (SRLL_{ij})$$

and the KKT multipliers $\mu_{ij}$ and $\lambda_{ij}$ defined for each pair of lower-level subproblem constraints $-t_{ij} \le 0$ and $t_{ij} \le T$ respectively, we have the following single-level reformulation of (5a)–(5c):

$$\min_{q,t,\mu,\lambda} \quad \sum_{i\in A}(q_i-1)^2 \tag{6a}$$

$$\forall i \in A \quad q_i^{\mathsf{min}} \le q_i \le q_i^{\mathsf{max}} \tag{6b}$$

$$\forall i<j\in A \quad \sum_{k\in K}\big[(x^0_{ik}-x^0_{jk})+t_{ij}\psi_{ijk}\big]^2 \ge d^2 \tag{6c}$$

$$\forall i<j\in A \quad \sum_{k\in K}\big[2t_{ij}\psi^2_{ijk}+2(x^0_{ik}-x^0_{jk})\psi_{ijk}\big]-\mu_{ij}+\lambda_{ij}=0 \tag{6d}$$

$$\forall i<j\in A \quad \mu_{ij},\ \lambda_{ij}\ge 0 \tag{6e}$$

$$\forall i<j\in A \quad \mu_{ij}\,t_{ij}=0 \tag{6f}$$

$$\forall i<j\in A \quad \lambda_{ij}\,t_{ij}-\lambda_{ij}\,T=0 \tag{6g}$$

$$\forall i<j\in A \quad 0\le t_{ij}\le T, \tag{6h}$$

where the symbol $\psi_{ijk}$ appearing in Eq. (6c) and (6d), and defined as:

$$\psi_{ijk} := q_iv_iu_{ik}-q_jv_ju_{jk}, \tag{7}$$

is used throughout the paper as short-hand for its definition in the right hand side.

Constraints (6d) (setting the gradient of the lower-level Lagrangian function equal to zero) correspond to the stationary condition of problem $(SRLL_{ij})$, Eq. (6e) and (6h) to dual and primal feasibility conditions respectively, and Eq. (6f)–(6g) to complementary slackness. Eq. (6c) enforce the safety distance for each KKT solution. No dual variable is introduced for Eq. (6c) since they are upper-level constraints.

We remark that the complementarity constraints Eq. (6f)–(6g) involve products of continuous decision variables, and, therefore, define nonconvex feasible sets in general. A possible reformulation based on MILP modeling may define mixed-integer linear feasible sets instead, but also requires the determination of some big-$M$ constant providing a valid bound to $\mu, \lambda$. Finding a correct big-$M$ cannot be done efficiently, i.e., in polynomial-time, unless $\mathsf{P} = \mathsf{NP}$ [23]. This particular reformulation, moreover, would not dispose of the nonconvexities in constraints Eq. (6c) and (6d). We therefore propose to solve the formulation above by means of global optimization techniques (see Section 6).

### 3.4 Dual reformulations

We propose another reformulation of the bilevel problem (5a)–(5c), which arises because the lower-level subproblems in (5c) are convex Quadratic Programs (QPs) occurring in constraints having general form:

$$\min_y\{\frac{1}{2}\, y^\top Q_x y + p_x^\top y + c_x \mid Ay \geq b\} \geq \mathsf{const} \tag{8}$$

with $Q_x$ positive semidefinite.

In general, given the dual variable $z$, by strong duality we have:

$$\begin{aligned} &\max_z\{\mathsf{LowerDualObj}(x,z) \mid \mathsf{LowerDualConstr}(x,z)\} \\ &= \min_y\{\frac{1}{2}\, y^\top Q_x y + p_x^\top y + c_x \mid Ay \geq b\}, \end{aligned} \tag{9}$$

where $\mathsf{LowerDualObj}(x,z)$ and $\mathsf{LowerDualConstr}(x,z)$ denote the objective function and the constraints of the dual problem of the left hand side of Eq. (8), respectively. If we impose the following inequality:

$$\max_z\{\mathsf{LowerDualObj}(x,z) \mid \mathsf{LowerDualConstr}(x,z)\} \geq \mathsf{const} \tag{10}$$

then, of course, Eq. (8) will also hold due to Eq. (9). The two constraints Eq. (8) and (10) are then equivalent. This was first proved in [7] limited to Dorn's dual problem. The following lemma generalizes the result.

**Lemma 1** *Assume Eq.* (8) *occurs in a bilevel formulation. Replacing Eq.* (8) *by the constraint set:*

$$\left.\begin{aligned} \mathsf{LowerDualObj}(x,z) &\geq \mathsf{const} \\ \mathsf{LowerDualConstr}(x,z) & \end{aligned}\right\} \tag{11}$$

*yields a single-level formulation with the same optima of the bilevel formulation.*

*Proof* We can replace Eq. (8) by the equivalent Eq. (10). If Eq. (10) is active (i.e. it is satisfied as an equality), then the maximum objective function value of the dual lower-level subproblem is const. Because of the max operator, its objective function cannot be greater than this value. This means that Eq. (11) can only be feasible when $\mathsf{LowerDualObj}(x,y)$ attains its maximum over its feasible region, defined by $\mathsf{LowerDualConstr}(x,z)$. If Eq. (10) is inactive, it has no effect on the optimum. Since Eq. (11) is a relaxation of Eq. (10), the same holds. □

Lemma 1 provides us with a scheme for reformulating (5a)–(5c), since, as already mentioned, Eq. (5c) is of the form (8). We observe that we can consider two different duals of the lower-level subproblems: Dorn's dual [24,25] (as done for $K = \{1,2\}$ in [7]), and Wolfe's dual [26]. We recall that strong duality is used to reformulate bilevel linear problems (see e.g. [27]), and bilevel linear integer programs as well (see [28]).

*3.4.1 Dorn's dual reformulation*

Given the dual variables $g_{ij}$ and $z_{ij}$ of each lower-level subproblem in the left hand side of Eq. (5c) (defined for constraints $-t_{ij} \leq 0$ and $t_{ij} \leq T$ respectively), using Dorn's dual [24,25] and Lemma 1, the following reformulation of (5a)–(5c) follows:

$$\min_{q,g,z} \quad \sum_{i \in A} (q_i - 1)^2 \tag{12a}$$

$$\forall i \in A \quad q_i^{\mathsf{min}} \leq q_i \leq q_i^{\mathsf{max}} \tag{12b}$$

$$\forall i < j \in A \quad -\sum_{k \in K} \psi_{ijk}^2 \, g_{ij}^2 - T z_{ij} \geq d^2 - \sum_{k \in K} (x_{ik}^0 - x_{jk}^0)^2 \tag{12c}$$

$$\forall i < j \in A \quad -\frac{z_{ij}}{2} - \sum_{k \in K} \psi_{ijk}^2 \, g_{ij} \leq \sum_{k \in K} (x_{ik}^0 - x_{jk}^0) \psi_{ijk} \tag{12d}$$

$$\forall i < j \in A \quad z_{ij} \geq 0, \tag{12e}$$

obtained by the application of Lemma 1 to replace the lower-level subproblems of Eq. (5a)–(5c) by their Dorn duals in the variables $g_{ij}, z_{ij}$ for each aircraft pair $i < j \in A$. This yields Eq. (12c)–(12d). Note that the primal lower-level variable $t_{ij}$ does not appear in (12a)–(12e). This is not an issue because we just want to know the new aircraft speeds such that each potential conflict is avoided.

**Proposition 1** *Eq.* (12a)–(12e) *is an exact reformulation of* (5a)–(5c).

*Proof* By Dorn's duality theory [24], $(D)$ is a dual problem of $(P)$:

$$\left.\begin{array}{rl} \min\limits_{y} & \frac{1}{2} y^\top Q y + p^\top y \\ & Ay \geq b \\ & y \geq 0 \end{array}\right\} \quad (P) \qquad \left.\begin{array}{rl} \max\limits_{g,z} & -\frac{1}{2} g^\top Q g + b^\top z \\ & A^\top z - Qg \leq p \\ & z \geq 0 \end{array}\right\} \quad (D)$$

In our case, we have:

- $y := t_{ij}$,
- $Q := 2 \sum\limits_{k \in K} \psi_{ijk}^2$,
- $p := 2 \sum\limits_{k \in K} (x_{ik}^0 - x_{jk}^0) \psi_{ijk}$,
- $A := -1$,
- $b := -T$.

Recall that $\psi_{ijk}$ is constant in the lower level since, by Eq. (7), it only depends on the upper-level variables $q_i$ and $q_j$. By easy replacements and Lemma 1, with $\mathsf{const} = d^2 - \sum\limits_{k \in K} (x_{ik}^0 - x_{jk}^0)^2$ in Eq. (11), the formulation Eq. (12a)–(12e) follows. □

*3.4.2 Wolfe's dual reformulation*

Another single-level reformulation can be obtained using Lemma 1 and Wolfe's dual [26] of the convex lower-level subproblems in Eq. (5c). The lower-level dual objective function is the Lagrangian of the lower-level primal problem in Eq. (5c)

$$\sum_{k\in K}\left[(x^0_{ik}-x^0_{jk})+t_{ij}\psi_{ijk}\right]^2+\alpha_{ij}(t_{ij}-T)-\beta_{ij}t_{ij},$$

where $\alpha_{ij}$ and $\beta_{ij}$ are the Lagrangian multipliers associated to the constraints $-t_{ij}\leq 0$ and $t_{ij}\leq T$, respectively. Therefore, by Lemma 1, we obtain the following reformulation of Eq. (5a)–(5c):

$$\min_{q,t,\alpha,\beta} \quad \sum_{i\in A}(q_i-1)^2 \tag{13a}$$

$$\forall i\in A \quad q_i^{\mathsf{min}}\leq q_i\leq q_i^{\mathsf{max}} \tag{13b}$$

$$\forall i<j\in A \quad \sum_{k\in K}\left[(x^0_{ik}-x^0_{jk})+t_{ij}\psi_{ijk}\right]^2+\alpha_{ij}(t_{ij}-T)-\beta_{ij}t_{ij}\geq d^2 \tag{13c}$$

$$\forall i<j\in A \quad \sum_{k\in K}\left[2t_{ij}\psi^2_{ijk}+2(x^0_{ik}-x^0_{jk})\psi_{ijk}+\alpha_{ij}-\beta_{ij}\right]=0 \tag{13d}$$

$$\forall i<j\in A \quad \alpha_{ij},\beta_{ij}\geq 0. \tag{13e}$$

We note that the single-level reformulation presented above involves some of the KKT conditions as constraints: the stationarity condition Eq. (13d) and the nonnegativity of the Lagrangian multipliers Eq. (13e). The (nonlinear) complementarity constraints, however, are not needed in Wolfe's duality [26]. The obtained reformulation Eq. (13a)–(13e) is exact.

**Proposition 2** *Eq.* (13a)–(13e) *is an exact reformulation of Eq.* (5a)–(5c).

*Proof* By Wolfe's duality theory [26], $(D)$ is a dual problem of $(P)$:

$$\left.\begin{array}{rl}\min\limits_{y} & \frac{1}{2}y^\top Qy+p^\top y+c\\ & Ay\geq b\\ & y\geq 0\end{array}\right\}\ (P) \qquad \left.\begin{array}{rl}\max\limits_{\alpha,\beta} & \mathcal{L}(y,\alpha,\beta)\\ & \frac{\partial\mathcal{L}}{\partial y}=0\\ & \alpha,\beta\geq 0\end{array}\right\}\ (D)$$

with:

$$\mathcal{L}(y,\alpha,\beta)=\frac{1}{2}y^\top Qy+p^\top y+c+\alpha(b-Ay)-\beta y,$$

and

$$\frac{\partial\mathcal{L}}{\partial y}=Qy+p+\alpha-\beta.$$

In our case, we have:

- $y:=t_{ij}$,

- $Q := 2 \sum_{k \in K} \psi_{ijk}^2$,
- $p := 2 \sum_{k \in K} (x_{ik}^0 - x_{jk}^0)\psi_{ijk}$,
- $c := \sum_{k \in K} (x_{ik}^0 - x_{jk}^0)^2$,
- $A := -1$,
- $b := -T$.

Again, we recall that $\psi_{ijk}$ is constant in the lower level because, by Eq. (7), it only depends on the upper level variables $q_i$ and $q_j$. By easy replacements and Lemma 1, with $\mathsf{const} = d^2$ in Eq. (11), the formulation Eq. (13a)–(13e) follows. □

## 4 Aircraft deconfliction via heading angle changes

In this section, we present several formulations to model the HACADP in two dimensions. The goal is again to satisfy the minimum safety distance for each pair of aircraft while minimizing the total deviations with respect to the original flight plan. The outcome of the HACADP will be the set of new heading angles of the aircraft (see Figure 3).

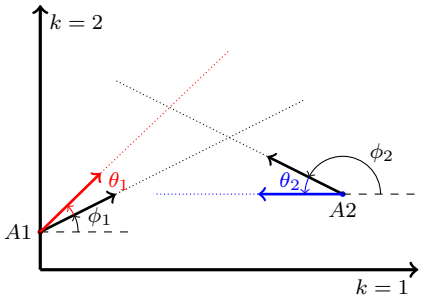


Fig. 3: Heading angles of two aircraft (in black before deconfliction)

The new parameters and variables used in the mathematical formulations introduced in this section are listed below.

- **Parameters:**
  - $\phi_i$ is the initial heading angle of aircraft $i$
  - $x_i^0$ is the first component of the initial position of aircraft $i$
  - $y_i^0$ is the second component of the initial position of aircraft $i$
  - $\theta_i^{\mathsf{min}}$ and $\theta_i^{\mathsf{max}}$ are the bounds on heading angle variation, with $\theta_i^{\mathsf{min}} < 0$ and $\theta_i^{\mathsf{max}} > 0$
- **Variables:** $\theta_i$ is the heading angle variation for each aircraft $i$

### 4.1 Bilevel formulation of the problem

We introduce a bilevel formulation where the upper-level decision variables are the heading angle variations $\theta_i$ (for all $i \in A$) and the lower-level decision variables are $t_{ij}$ (for all $i < j \in A$). Each lower-level subproblem is parametrized by the upper-level variables $\theta_i$, $\theta_j$.

$$\min_{\theta,t} \quad \sum_{i\in A} \theta_i^2 \tag{14a}$$

$$\forall i \in A \quad \theta_i^{\mathsf{min}} \leq \theta_i \leq \theta_i^{\mathsf{max}} \tag{14b}$$

$$\begin{aligned}\forall i < j \in A \quad \min_{t_{ij}\in[0,T]} &\left[(x_i^0 - x_j^0) + t_{ij}(\cos(\phi_i+\theta_i)v_i - \cos(\phi_j+\theta_j)v_j)\right]^2 \\ &+ \left[(y_i^0 - y_j^0) + t_{ij}(\sin(\phi_i+\theta_i)v_i - \sin(\phi_j+\theta_j)v_j)\right]^2 \geq d^2.\end{aligned} \tag{14c}$$

The convex objective function of the upper level is the sum of squared heading angle changes, which are bounded by $[\theta_i^{\mathsf{min}}, \theta_i^{\mathsf{max}}]$. The objective of each lower-level subproblem is to minimize the squared Euclidean distance between aircraft $i$ and $j$ over $t_{ij} \in [0, T]$. Its expression is obtained by defining the position $(x_i(t), y_i(t))$ of aircraft $i$ at time $t$ as

$$x_i(t) = x_i^0 + \cos(\phi_i + \theta_i)v_i t \quad \text{and} \quad y_i(t) = y_i^0 + \sin(\phi_i + \theta_i)v_i t.$$

Note that the lower-level objective function is also convex in $t_{ij}$. Similarly to Eq. (5c) of the previous section, Eq. (14c) guarantees that the minimum squared distance between each pair within the time horizon is at least $d^2$. In Appendix A, we detail the procedure to calculate the optimal instant of time in which each aircraft $i$ has to change again its heading angle to reach its original trajectory.

### 4.2 KKT reformulation

We derive the KKT reformulation of Eq. (14a)–(14c), based on KKT multipliers $\lambda_{ij}$ (resp. $\mu_{ij}$) associated to constraints $t_{ij} \leq T$ (resp. $-t_{ij} \leq 0$) of each lower level subproblem

$$\begin{aligned}\min_{t_{ij}} \quad &\left[(x_i^0 - x_j^0) + t_{ij}(\cos(\phi_i+\theta_i)v_i - \cos(\phi_j+\theta_j)v_j)\right]^2 \\ &+ \left[(y_i^0 - y_j^0) + t_{ij}(\sin(\phi_i+\theta_i)v_i - \sin(\phi_j+\theta_j)v_j)\right]^2 \qquad (HACLL_{ij}) \\ s.t.\ \ &-t_{ij} \leq 0 \\ &t_{ij} \leq T.\end{aligned}$$

Since Slater's condition holds, and the lower level is convex in the variable $t_{ij}$, the reformulation is exact [21, Sec. 3.5].

$$\min_{\theta,t,\lambda,\mu} \quad \sum_{i\in A} \theta_i^2 \tag{15a}$$
$$\forall i \in A \quad \theta_i^{\mathsf{min}} \leq \theta_i \leq \theta_i^{\mathsf{max}} \tag{15b}$$
$$\forall i < j \in A \quad \left[(x_i^0 - x_j^0) + t_{ij}c_{ij}\right]^2 + \left[(y_i^0 - y_j^0) + t_{ij}s_{ij}\right]^2 \geq d^2 \tag{15c}$$
$$\forall i < j \in A \quad 2t_{ij}(c_{ij}^2 + s_{ij}^2) + 2(x_i^0 - x_j^0)c_{ij} + 2(y_i^0 - y_j^0)s_{ij} + \lambda_{ij} - \mu_{ij} = 0 \tag{15d}$$
$$\forall i < j \in A \quad \lambda_{ij},\ \mu_{ij} \geq 0 \tag{15e}$$
$$\forall i < j \in A \quad \lambda_{ij}\, t_{ij} - \lambda_{ij}\, T = 0 \tag{15f}$$
$$\forall i < j \in A \quad \mu_{ij}\, t_{ij} = 0 \tag{15g}$$
$$\forall i < j \in A \quad 0 \leq t_{ij} \leq T. \tag{15h}$$

where the symbols $c_{ij}$ and $s_{ij}$ are shorthand for the following nonlinear expressions:

$$c_{ij} := \cos(\phi_i + \theta_i)v_i - \cos(\phi_j + \theta_j)v_j \tag{16}$$
$$s_{ij} := \sin(\phi_i + \theta_i)v_i - \sin(\phi_j + \theta_j)v_j. \tag{17}$$

The formulation in Eq. (15a)–(15c) is a single-level Nonlinear Programming (NLP) problem in the variables $\theta$, $t$, $\lambda$, and $\mu$. Constraints Eq. (15d) correspond to stationarity conditions of the lower-level subproblems, Eq. (15h) to primal feasibility, Eq. (15e) to dual feasibility, Eq. (15f) and Eq. (15g) to complementarity conditions. We, again, require that the safety distance is satisfied for each pair of aircraft and for each $t_{ij}$ satisfying the KKT conditions imposed in (15d)–(15h), with constraints (15c).

### 4.3 Dual reformulations

We follow the procedure discussed in Section 3.4 in order to obtain two *dual reformulations* of Eq. (14a)–(14c). The first involves Dorn's dual of the lower-level subproblems, while the second involves Wolfe's dual.

*4.3.1 Dorn's dual reformulation*

Given the dual variables $g_{ij}$ and $z_{ij}$ of the lower-level subproblems, using Dorn's dual and Lemma 1, the following reformulation of Eq. (14a)–(14c) is obtained:

$$\min_{\theta,g,z} \quad \sum_{i\in A} \theta_i^2 \tag{18a}$$

$$\forall i \in A \quad \theta_i^{\mathsf{min}} \leq \theta_i \leq \theta_i^{\mathsf{max}} \tag{18b}$$

$$\forall i < j \in A \quad -g_{ij}^2(c_{ij}^2 + s_{ij}^2) - Tz_{ij} \geq d^2 - (x_i^0 - x_j^0)^2 - (y_i^0 - y_j^0)^2 \tag{18c}$$

$$\forall i < j \in A \quad -\frac{z_{ij}}{2} - (c_{ij}^2 + s_{ij}^2)g_{ij} \leq (x_i^0 - x_j^0)c_{ij} + (y_i^0 - y_j^0)s_{ij} \tag{18d}$$

$$\forall i < j \in A \quad z_{ij} \geq 0. \tag{18e}$$

The formulation in Eq. (18a)–(18e) is a single-level problem in the variables $\theta, g$, and $z$, the exactness of which is proved below. Note that the primal lower-level variable $t_{ij}$ does not appear in (18a)–(18e). This is not an issue because we just want to know the new heading angles such that each potential conflict is avoided.

**Proposition 3** *Eq.* (18a)–(18e) *is an exact reformulation of Eq.* (14a)–(14c).

*Proof* By Dorn's duality theory [24], $(D)$ is a dual problem of $(P)$:

$$\left.\begin{array}{rl} \min\limits_{y} & \frac{1}{2}y^\top Qy + p^\top y \\ & Ay \geq b \\ & y \geq 0 \end{array}\right\} \quad (P) \qquad \left.\begin{array}{rl} \max\limits_{g,z} & -\frac{1}{2}g^\top Qg + b^\top z \\ & A^\top z - Qg \leq p \\ & z \geq 0 \end{array}\right\} \quad (D)$$

In our case, we have:

- $y := t_{ij}$,
- $Q := 2(c_{ij}^2 + s_{ij}^2)$,
- $p := 2(x_i^0 - x_j^0)c_{ij} + 2(y_i^0 - y_j^0)s_{ij}$,
- $A := -1$,
- $b := -T$.

We recall that $c_{ij}$ and $s_{ij}$ are constant in the lower level because, by Eq. (16)–(17), they only depend on the upper-level variables $\theta_i$ and $\theta_j$. By easy replacements and Lemma 1, with $\mathsf{const} = d^2 - (x_i^0 - x_j^0)^2 - (y_i^0 - y_j^0)^2$ in Eq. (11), Eq. (18a)–(18e) follow. □

*4.3.2 Wolfe's dual reformulation*

Using Lemma 1 and Wolfe's dual of each lower-level subproblem in the variables $\alpha_{ij}$ and $\beta_{ij}$, we obtain the following single-level reformulation of of Eq. (14a)–

(14c):

$$\min_{\theta,t,\alpha,\beta} \quad \sum_{i\in A} \theta_i^2 \tag{19a}$$

$$\forall i \in A \quad \theta_i^{\mathsf{min}} \leq \theta_i \leq \theta_i^{\mathsf{max}} \tag{19b}$$

$$\forall i < j \in A \quad \left[(x_i^0 - x_j^0) + t_{ij}c_{ij}\right]^2 + \left[(y_i^0 - y_j^0) + t_{ij}s_{ij}\right]^2 + \alpha_{ij}(t_{ij} - T) - \beta_{ij}t_{ij} \geq d^2 \tag{19c}$$

$$\forall i < j \in A \quad 2t_{ij}(c_{ij}^2 + s_{ij}^2) + 2(x_i^0 - x_j^0)c_{ij} + 2(y_i^0 - y_j^0)s_{ij} + \alpha_{ij} - \beta_{ij} = 0 \tag{19d}$$

$$\forall i < j \in A \quad \alpha_{ij}, \beta_{ij} \geq 0. \tag{19e}$$

With Eq. (19c), the Lagrangian of each lower-level subproblem is required to exceed the minimum required safety distance. The stationarity KKT condition (gradient of the Lagrangian equal to zero) corresponds to (19d). Constraints (19e) impose the nonnegativity of the dual variables $\alpha_{ij}$ and $\beta_{ij}$. The exactness of formulation (19a)–(19e) is proved below.

**Proposition 4** *Eq.* (19a)–(19e) *is an exact reformulation of Eq.* (14a)–(14c).

*Proof* By Wolfe's duality theory [26], $(D)$ is a dual problem of $(P)$:

$$\left.\begin{array}{rl} \min\limits_{y} & \frac{1}{2}y^\top Qy + p^\top y + c \\ & Ay \geq b \\ & y \geq 0 \end{array}\right\} \quad (P) \qquad \left.\begin{array}{rl} \max\limits_{\alpha,\beta} & \mathcal{L}(y,\alpha,\beta) \\ & \frac{\partial \mathcal{L}}{\partial y} = 0 \\ & \alpha, \beta \geq 0 \end{array}\right\} \quad (D)$$

with:

$$\mathcal{L}(y,\alpha,\beta) = \frac{1}{2}y^\top Qy + p^\top y + c + \alpha(b - Ay) - \beta y,$$

and

$$\frac{\partial \mathcal{L}}{\partial y} = Qy + p + \alpha - \beta.$$

In our case, we have:

- $y := t_{ij}$,
- $Q := 2(c_{ij}^2 + s_{ij}^2)$,
- $p := 2(x_i^0 - x_j^0)c_{ij} + 2(y_i^0 - y_j^0)s_{ij}$,
- $c := (x_i^0 - x_j^0)^2 + (y_i^0 - y_j^0)^2$,
- $A := -1$,
- $b := -T$.

Again we recall that $c_{ij}$ and $s_{ij}$ are constant in the lower level because, by Eq. (16)–(17), they only depend on the upper-level variables $\theta_i$ and $\theta_j$. By easy replacements and Lemma 1, with $\mathsf{const} = d^2$ in Eq. (11), Eq. (19a)–(19e) follow. □

We recall that in this section we used the symbols $c_{ij}$ and $s_{ij}$ defined in (16)–(17) to replace differences between trigonometric operators having the main control variable $\theta_i$ in the argument.

## 5 Cut generation algorithm

Cutting-plane approaches are one of the major techniques used for solving linear, quadratic [30], and convex semi-infinite programs. In [7], we proposed a tailored CG algorithm for the bilevel formulation Eq. (5a)–(5c) in two dimensions. We recall our CG algorithm in Section 5.1 for SRADP in $k$ dimensions. In Section 5.2, we tailor the CG algorithm for the bilevel formulation Eq. (14a)–(14c) of HACADP.

The problem solved at each iteration of the CG the algorithm is nonconvex. In our implementation, its solution is obtained either with global solvers or, in the interest of efficiency, by executing a local NLP solver several times within a multistart procedure that starts from randomly chosen points.

We assume that aircraft are separated at the beginning of the time horizon considered, otherwise the problem is infeasible.

In Appendix B we discuss dominance relationships among quadratic cuts, which we do not take into account in the rest of the paper. In fact, we prove that a cut added in a certain iteration for a pair of aircraft will not be dominated by any other cut added in future iterations for the same pair.

### 5.1 Cut generation algorithm for ADP via speed regulation

Algorithm 1 is a solution algorithm for the bilevel formulation (5a)–(5c), which iteratively defines the feasible set of the upper-level problem by means of quadratic cuts in the upper-level variables $q$. At each iteration $h$, the relaxation $R_h$ of the original bilevel problem, obtained by considering the upper-level problem together with the cuts added in previous iterations, is solved. At the outset, $R_1$ is:

$$\begin{aligned} \min_{q} \quad & \sum_{i \in A} (q_i - 1)^2 \\ \forall i \in A \quad & q_i^{\mathsf{min}} \leq q_i \leq q_i^{\mathsf{max}}. \end{aligned}$$

**Algorithm 1** CG algorithm for SRADP

1: Let $h = 1$. Initialize the relaxation $R_h$ of the bilevel program, obtained by considering the upper-level problem only.
2: **while** true **do**
3: Solve $R_h$ to obtain the optimal solution $q^*$.
4: **for** each aircraft pair $(i, j)$ **do**
5: **if** $\forall\, k \in K : (q_i^* v_i u_{ik} - q_j^* v_j u_{jk}) = 0$ **then**
6: Set $\tau_{ij}^h = \frac{T}{2}$.
7: **else**
8: Compute the instant $\tau_{ij}^h \in [0, T]$ as

$$\tau_{ij}^h = \min\left\{T, \max\left\{0, -\frac{\sum_{k\in K}(x_{ik}^0 - x_{jk}^0)(q_i^* v_i u_{ik} - q_j^* v_j u_{jk})}{\sum_{k\in K}(q_i^* v_i u_{ik} - q_j^* v_j u_{jk})^2}\right\}\right\}.$$

9: **end if**
10: **end for**
11: **if** $\sum_{k\in K}((x_{ik}^0 - x_{jk}^0) + \tau_{ij}^h(q_i^* v_i u_{ik} - q_j^* v_j u_{jk}))^2 \geq d^2 \quad \forall i < j \in A$ **then**
12: The algorithm terminates and $q^*$ is the optimal solution of the bilevel formulation.
13: **else**
14: For each pair $(i, j)$ violating the inequality, define $R_{h+1}$ as $R_h$ with the adjoined inequality:

$$\sum_{k\in K}((x_{ik}^0 - x_{jk}^0) + \tau_{ij}^h(q_i v_i u_{ik} - q_j v_j u_{jk}))^2 \geq d^2. \tag{20}$$

15: $h := h + 1$
16: **end if**
17: **end while**

The problem $R_h$, solved at each iteration of Algorithm 1, is nonconvex since constraints (20) are of the form $f(q_i, q_j) \geq d^2$ with $f(q_i, q_j)$ convex. Therefore, in order to find global optima of $R_h$, a global optimization algorithm should be employed. This, however, would make the CG algorithm excessively slow. In our implementation (see Section 6) we chose to heuristically solve $R_h$ using a multistart algorithm calling a local NLP solver, from randomly chosen starting points, when global optimization solvers are too slow.

$\tau_{ij}^h$ is the instant for which the distance between $i$ and $j$ is minimum. If this distance is greater than or equal to the safety value for each pair of aircraft, the algorithm terminates at Step 12, as $q^*$ must be an optimal solution of the bilevel formulation.

Note that, in Step 8, $\tau_{ij}^h$, easily computed in closed form, is set to 0 or $T$ if

$$-\frac{\sum_{k\in K}(x_{ik}^0 - x_{jk}^0)(q_i^* v_i u_{ik} - q_j^* v_j u_{jk})}{\sum_{k\in K}(q_i^* v_i u_{ik} - q_j^* v_j u_{jk})^2}$$

is negative or greater than $T$ respectively. In Step 6, it is set to $\frac{T}{2}$ if

$$(q_i^* v_i u_{ik} - q_j^* v_j u_{jk}) = 0,$$

for all $k \in K$ i.e., if aircraft $i$ and $j$ fly on parallel trajectories with the same speed. Having assumed that aircraft are separated at the beginning (namely $\sum_{k \in K} (x^0_{ik} - x^0_{jk})^2 \geq d^2$), no cut will be added in the next steps of the algorithm.

### 5.2 Cut generation algorithm for ADP via HAC

We propose a tailored version of the CG algorithm for the bilevel formulation Eq. (14a)–(14c), which models the HACADP. In this case, the nonconvex problem $R_1$ solved at the first iteration is

$$\begin{aligned} \min_{\theta} \quad & \sum_{i \in A} \theta_i^2 \\ \forall i \in A \quad & \theta_i^{\mathsf{min}} \leq \theta_i \leq \theta_i^{\mathsf{max}}. \end{aligned}$$

**Algorithm 2** CG algorithm for HACADP

Let $h = 1$. Initialize the relaxation $R_h$ of the bilevel program, obtained by considering the upper-level problem only.
**while** true **do**
  Solve $R_h$ to obtain the optimal solution $\theta^*$.
  **for** each aircraft pair $(i, j)$ **do**
    **if** $c^*_{ij} = 0$ and $s^*_{ij} = 0$ **then**
      Set $\tau^h_{ij} = \frac{T}{2}$.
    **else**
      Compute the instant $\tau^h_{ij} \in [0, T]$ as

$$\tau^h_{ij} = \min \left\{ T, \max \left\{ 0, -\frac{(x^0_i - x^0_j)c^*_{ij} + (y^0_i - y^0_j)s^*_{ij}}{(c^*_{ij})^2 + (s^*_{ij})^2} \right\} \right\},$$

      with $c^*_{ij} = \cos(\phi_i + \theta^*_i)v_i - \cos(\phi_j + \theta^*_j)v_j$ and $s^*_{ij} = \sin(\phi_i + \theta^*_i)v_i - \sin(\phi_j + \theta^*_j)v_j$.
    **end if**
  **end for**
  **if** $\left[(x^0_i - x^0_j) + \tau^h_{ij} c^*_{ij}\right]^2 + \left[(y^0_i - y^0_j) + \tau^h_{ij} s^*_{ij}\right]^2 \geq d^2 \quad \forall i < j \in A$ **then**
    The algorithm terminates and $\theta^*$ is the optimal solution of the bilevel formulation.
  **else**
    For each pair $(i, j)$ violating the inequality, define $R_{h+1}$ as $R_h$ with the adjoined inequality:

$$\begin{aligned} &\left[(x^0_i - x^0_j) + \tau^h_{ij}(\cos(\phi_i + \theta_i)v_i - \cos(\phi_j + \theta_j)v_j)\right]^2 \\ &+ \left[(y^0_i - y^0_j) + \tau^h_{ij}(\sin(\phi_i + \theta_i)v_i - \sin(\phi_j + \theta_j)v_j)\right]^2 \geq d^2. \end{aligned} \tag{21}$$

    $h := h + 1$
  **end if**
**end while**

Again, the problem $R_h$, solved at each iteration of the algorithm, is nonconvex since the constraints (21) are of the form $f(\theta_i, \theta_j) \geq d^2$ with $f(\theta_i, \theta_j)$

convex. We find $\theta^*$ in Step 3 using a global NLP solver or, when the time limit is exceeded, with a local NLP solver within a multistart procedure from randomly chosen starting points.

As in Algorithm 1, $\tau_{ij}^h \in [0, T]$ indicates when the distance between $i$ and $j$ is minimized and it is always computed in closed form in Step 8. If this distance satisfies the safety threshold for each pair of aircraft, the algorithm terminates at Step 12. Again, in Step 6 we discarded the case in which $\tau_{ij}^h$ is not well defined whenever aircraft $i$ and $j$ share the same direction and speed.

## 6 Computational experiments

For the SRADP in $k$ dimensions, we use a 3D generalization of the 2D instances tested in [7] (named *sphere instances* in Table 1), where $n$ aircraft are placed on a sphere of a given radius $r$ — see Figure 4. We consider also instances in which aircraft move along straight 3D trajectories (named *non-sphere instances* in Table 1), which intersect in at least $\frac{n}{2}$ conflict points. For reproducibility purposes these instances are available online at the public repository `https://github.com/MartinaCerulli/SRADP`.

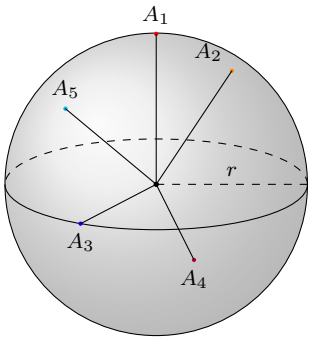


Fig. 4: $n$ conflicting aircraft flying towards the center of a sphere

A trajectory is defined by two angles: the so-called pitch angle $\gamma_i$ (angle that the vector of the direction $u_i$ forms with the axis $k_3$) and the heading angle $\phi_i$ (angle between the projection of $u_i$ onto the $k_1k_2$-plane and the axis $k_1$) — see Figure 5.

We test our approaches for the HACADP using the set of instances proposed in [17], where $n$ aircraft are randomly placed on a circle of a given radius $r$. All aircraft speeds are initially set to the same value, and their trajectories are such that the aircraft fly exactly or almost exactly towards the center of the circle — see Figure 6 from [7]. These problems, characterized by an unrealistic highly symmetric configuration, are known in literature as *circle problems*.

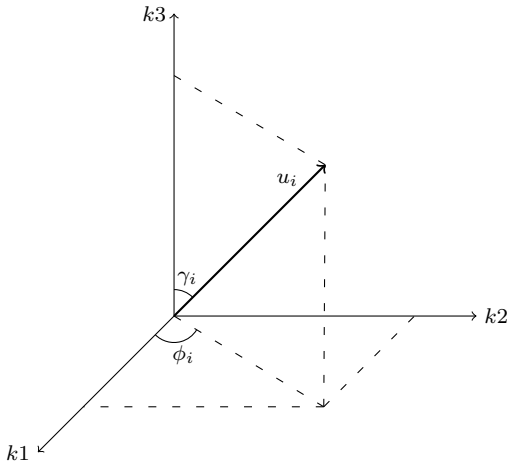


Fig. 5: The 3-dimensional airspace

We also consider instances, always from [17], in which aircraft are placed around a circle and have trajectories with a starting heading angle $\phi_i$ randomly chosen in $[-\frac{\pi}{6}, \frac{\pi}{6}]$ with respect to the diameter of the circle. The end point of each trajectory belongs to the circle as well. Note that these problems, named *random circle instances* in Table 2, are more realistic than *circle problems* without deviation.

Moreover, we test some *non-circle instances* from [11] in which aircraft move along straight trajectories intersecting in $n_c$ conflict points.

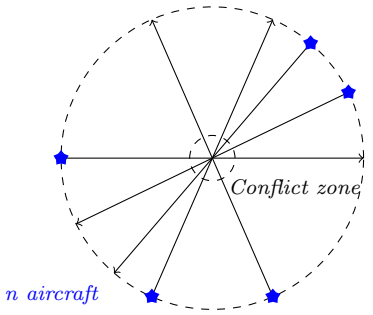


Fig. 6: $n$ conflicting aircraft flying towards the center of a circle - from [7]

In all the experiments, we consider standard safety distance $d = 5\,\text{NM}$ and a time horizon of $T = 2$ hours. It could be worth carrying out a sensitivity analysis to evaluate the impact of varying the value of $T$ in future works. The choice of solver was carried out through some preliminary computational experiments. All the solvers are run with their default settings. The tests are performed on a 2.53GHz Intel(R) Xeon(R) CPU with 49.4 GB RAM.

### 6.1 ADP in 3 dimensions via speed regulation

For the *sphere instances* the initial speed is $v_i = 400\,\text{NM/h}$ for each $i \in A$ and the angles $\gamma_i$ and $\phi_i$ are randomly generated within $[0, \frac{\pi}{2}]$ and parameters $x^0_{ik}$ and $u_{ik}$ are given by

$$u_{i1} = \cos(\phi_i)\sin(\gamma_i), \quad u_{i2} = \sin(\phi_i)\sin(\gamma_i), \quad u_{i3} = \cos(\gamma_i), \quad x^0_{ik} = -r\,u_{ik}$$

where the sphere radius $r$ is chosen in $\{100, 200, \ldots, 700\}$. The bounds $q_i^{\mathsf{min}}$ and $q_i^{\mathsf{max}}$ are set to 0.94 and 1.03 respectively, following the weaker bounds proposed by the ERASMUS project. We decided to stick to these well-known bounds on $q_i$, having in mind the ADP application. In the context of urban air mobility, vehicles might have less strict bounds on admissible deceleration or acceleration. However, urban air mobility concepts are still being developed and estimations on these control parameters are not yet available.

We implement the single-level formulations using the AMPL modeling language [31] and solve them with the global optimization solver Baron [32] (B in the Table 1). When Baron exceeds the time-limit (set to 3600 seconds), we use a Multistart algorithm (MS in Table 1), which performs 1000 calls to the SNOPT [33] local NLP solver from randomly sampled starting points.

We solve the bilevel formulation using the CG algorithm in Section 5 (CG in Table 1) with maximum iteration number set to 1000; at each iteration we solve the relaxed formulation $R_h$ using the SNOPT [33] local NLP solver called 50 times within a Multistart procedure from randomly chosen points.

All the results are reported in Table 1. The headings are the following: $n$ number of aircraft; $r$ radius of the sphere in NM; *obj* best objective value found by each model; *cpu* computing time in seconds; *slv* solver used (for the CG algorithm the solver used to solve the inner problem $R_h$); *UB* and *LB* respectively upper and lower bound on the optimal solution value, determined by Baron when time limit is reached (we write 0 whenever a bound is less than $10^{-6}$); *it* number of CG iterations, i.e., number of times $R_h$ is solved. Table 1 is divided into two blocks, one for spherical instances and another for non-spherical ones. Each block is followed by a row that shows the percentage of instances for which each approach outperforms (or is as good as) the rest. To build this last row, only those instances for which the same resolution method was applied (namely B or MS) were considered.

The value of the objective function is always very small, given the nature of the problem ($q$ must be in $[0.94, 1.03]$). The tight speed variation bounds imposed by ERASMUS project lead to an additional complication since instances are not guaranteed to be feasible. Best objective values and minimum required time are reported in bold for each instance. The best formulation in terms of solution quality is the one in Eq. (6a)–(6c) based on KKT conditions of the lower-level subproblems. In terms of computational efficiency, for most of the instances the CG Algorithm 1 is the best.

Table 1: Results obtained solving 4 different formulations of SRADP

| Instances | | KKT reformulation | | | | Dorn's dual reformulation | | | | Wolfe's dual reformulation | | | | CG | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $n$ | $r$ | *obj* | *cpu* | slv | UB | LB | *obj* | *cpu* | slv | UB | LB | *obj* | *cpu* | slv | UB | LB | *obj* | *cpu* | slv | it |
| Sphere | | | | | | | | | | | | | | | | | | | | |
| 2 | 100 | **0.002220** | 0.63 | B | - | - | 0.002223 | 0.86 | B | - | - | 0.002226 | **0.62** | B | - | - | 0.002226 | 0.68 | B | 5 |
| 3 | 200 | 0.001406 | 9.64 | B | - | - | **0.001404** | 5.86 | B | - | - | 0.001407 | 11.4 | B | - | - | 0.001408 | **0.66** | MS | 10 |
| 4 | 200 | 0.003713 | 312 | B | - | - | **0.003703** | 270 | B | - | - | 0.003709 | 3582 | B | - | - | 0.003714 | **0.86** | MS | 11 |
| 5 | 300 | **0.002959** | 5.90 | MS | 0.002959 | 0.000070 | **0.002959** | 4.53 | MS | 0.002959 | 0.000473 | **0.002959** | 11.7 | MS | 0.002959 | 0 | **0.002959** | **1.19** | MS | 13 |
| 6 | 300 | **0.005847** | 9.69 | MS | 0.005847 | 0.000001 | **0.005847** | 7.41 | MS | 0.005847 | 0.000188 | **0.005847** | 12.6 | MS | 0.005847 | 0 | 0.005916 | **3.05** | MS | 24 |
| 7 | 500 | **0.002855** | 15.3 | MS | 0.002903 | 0 | **0.002855** | 20.2 | MS | 0.002855 | 0 | 0.002856 | 26.4 | MS | 0.002906 | 0 | 0.002887 | **12.2** | MS | 41 |
| 8 | 500 | 0.004549 | **16.8** | MS | 0.004557 | 0 | 0.004605 | 35.6 | MS | 0.004641 | 0 | 0.004572 | 23.1 | MS | 0.004744 | 0 | **0.004521** | 64.09 | MS | 55 |
| 9 | 500 | **0.006987** | **20.4** | MS | 0.007491 | 0 | 0.007137 | 53.0 | MS | 0.007188 | 0 | 0.007109 | 28.1 | MS | 0.007380 | 0 | 0.007135 | 78.9 | MS | 64 |
| 10 | 600 | 0.006410 | **32.2** | MS | 0.006484 | 0 | 0.006436 | 70.7 | MS | 0.006420 | 0 | **0.006402** | 34.7 | MS | 0.006408 | 0 | 0.006497 | 204 | MS | 121 |
| 12 | 700 | 0.008511 | 79.4 | MS | 0.008713 | 0 | 0.008685 | **57.4** | MS | 0.008963 | 0 | **0.008404** | 81.4 | MS | 0.008800 | 0 | 0.008676 | 624 | MS | 172 |
| % best *cpu* | | | | 30 | | | | | 10 | | | | | 10 | | | | 50 | | |
| Non-sphere | | | | | | | | | | | | | | | | | | | | |
| 2 | - | **0.000305** | **0.14** | B | - | - | **0.000305** | 0.17 | B | - | - | **0.000305** | 0.26 | B | - | - | **0.000305** | 1.05 | B | 10 |
| 4 | - | **0.003278** | 54.9 | B | - | - | 0.003283 | 5.19 | MS | 0.003278 | 0.001648 | 0.003709 | 3584 | B | - | - | 0.003283 | **0.46** | MS | 6 |
| 6 | - | 0.006004 | 14.5 | MS | 0.006001 | 0.000288 | 0.006004 | 10.2 | MS | 0.006004 | 0.000003 | 0.006004 | 16.2 | MS | 0.006004 | 0 | **0.006003** | **0.45** | MS | 7 |
| 8 | - | 0.011705 | 19.0 | MS | 0.011705 | 0.000043 | 0.011705 | 20.9 | MS | 0.011705 | 0 | 0.011705 | 17.9 | MS | 0.011705 | 0 | **0.011704** | **0.41** | MS | 6 |
| 10 | - | **0.015025** | 34.5 | MS | 0.015025 | 0.000174 | **0.015025** | 67.3 | MS | 0.015025 | 0 | **0.015025** | 30.0 | MS | 0.015025 | 0 | **0.015025** | **0.63** | MS | 8 |
| % best *cpu* | | | | 25 | | | | | 0 | | | | | 0 | | | | 75 | | |

### 6.2 ADP in 2 dimensions via heading angles changes

As mentioned above, the HACADP instances are taken from [17,11]. The authors set $v_i = 400\,\text{NM/h}$ for each $i \in A$ for all the instances. For the *circle instances* the angles $\phi_i$ are randomly generated and parameters $x_i^0$ and $y_i^0$ are given by

$$x_i^0 = -r\cos(\phi_i), \quad y_i^0 = -r\sin(\phi_i).$$

For the *random circle instances* both the angles $\phi_i$ and the parameters $x_i^0$ and $y_i^0$ are randomly generated. The bounds $\theta_i^{\mathsf{min}}$ and $\theta_i^{\mathsf{max}}$ are set to $-\pi/6$ and $\pi/6$ respectively.

We again implement the formulations using the AMPL modeling language [31] and solve them with the global optimization solver Couenne [34] (C in the Table 2). We do not use Baron because it cannot handle the trigonometric functions sine and cosine. When Couenne exceeds the time-limit (set to 3600 seconds), we use a Multistart algorithm (MS in Table 2). It performs 1000 calls to SNOPT [33] for KKT reformulation Eq. (15a)–(15c) and Wolfe's dual reformulation Eq. (19a)–(19e), and 1000 calls to IPOPT [35] in the case of Dorn's dual reformulation Eq. (18a)–(18e). In all of the calls, starting points are randomly chosen.

We solve the bilevel formulation using the CG algorithm in Section 5 (CG in Table 2) with maximum iteration number set to 1000; at each iteration we solve the relaxed formulation $R_h$ using Couenne, or, when the CG exceed the time-limit of 3600 seconds, the SNOPT [33] local NLP solver called 50 times within a Multistart procedure from randomly chosen points.

Our results are reported in Tables 2 and 3. The headings are the following: name of the instance; $n$ number of aircraft; $n_c$ number of potential conflicts; *obj* best objective value found by each model; *cpu* computing time in seconds; *slv* solver used (in the last column, the solver used to solve the inner problem $R_h$ of the CG algorithm); *UB* and *LB* respectively upper and lower bound on the optimal solution value, determined by Couenne when time limit is reached (we write 0 whenever a bound is less than $10^{-6}$); *it* number of CG iterations, i.e., number of times $R_h$ is solved. Table 2 includes Circle instances, while Table 3 is divided into two blocks, which correspond to Random Circle and Non-Circle instances. Both Table 2 and each block of Table 3 are followed by a row that shows the percentage of instances for which each approach outperforms (or is as good as) the rest. As before, only instances solved with the same method (namely C or MS) were used for the average. In Tables 2 and 3 the results on *circle* and *random circle* instances are also compared with those that are obtained using HAC only (without pre-processing) in [17]. Best objective values and minimum required time are reported in bold for each instance.

Among the models proposed in this paper, for most of the instances the CG algorithm is the best in terms of objective function and computational time, even when the inner problem $R_h$ is solved using Couenne. Looking at the comparison of our results with the ones obtained in [17], it appears that they are comparable.

Table 2: Results obtained solving 4 different formulations of HACADP compared with those obtained in [17], *Circle instances*

| Instances | | | KKT reformulation | | | | Dorn's dual reformulation | | | | Wolfe's dual reformulation | | | | CG | | | | [17] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | $n$ | $n_c$ | *obj* | *cpu* | *slv* | UB | LB | *obj* | *cpu* | *slv* | UB | LB | *obj* | *cpu* | *slv* | UB | LB | *obj* | *cpu* | *slv* | it | *obj* |
| Circle | | | | | | | | | | | | | | | | | | | | | | |
| *pb_n2* | 2 | 1 | **0.001250** | 0.15 | C | - | - | **0.001250** | 0.14 | C | - | - | **0.001250** | 0.18 | C | - | - | **0.001250** | **0.11** | C | 2 | **0.001250** |
| *pb_n3_1* | 3 | 3 | **0.002501** | 0.83 | C | - | - | **0.002501** | 1.27 | C | - | - | **0.002501** | 0.83 | C | - | - | **0.002501** | **0.74** | C | 3 | **0.002501** |
| *pb_n3_2* | 3 | 3 | 0.006672 | 7.83 | C | - | - | 0.006672 | 5.82 | C | - | - | **0.006657** | 7.89 | C | - | - | 0.006671 | **2.00** | MS | 20 | 0.006665 |
| *pb_n3_3* | 3 | 3 | **0.000950** | **1.03** | C | - | - | **0.000950** | 1.16 | C | - | - | **0.000950** | 1.26 | C | - | - | **0.000950** | 1.67 | C | 6 | **0.000950** |
| *pb_n4_1* | 4 | 6 | 0.007240 | **416** | C | - | - | 0.007240 | 529.65 | C | - | - | **0.007228** | 1410 | C | - | - | 0.007240 | **1.64** | MS | 15 | 0.007240 |
| *pb_n4_2* | 4 | 6 | **0.017061** | 2497 | C | - | - | 0.017094 | 38.2 | MS | 0.017095 | 0.017074 | 0.017095 | 21.8 | MS | 0.017065 | 0.0169614 | 0.017098 | **3.25** | MS | 27 | 0.017065 |
| *pb_n4_3* | 4 | 6 | **0.001318** | **25.3** | C | - | - | **0.001318** | 85.2 | C | - | - | **0.001318** | 33.5 | C | - | - | **0.001318** | 57.1 | C | 10 | **0.001318** |
| *pb_n5_1* | 5 | 10 | 0.011629 | 25.3 | MS | 0.011629 | 0.004543 | 0.011629 | 38.4 | MS | 0.011629 | 0.011615 | 0.011629 | 27.4 | MS | 0.011629 | 0.004583 | **0.011627** | **2.16** | MS | 17 | 0.011629 |
| *pb_n5_2* | 5 | 10 | **0.018467** | 28.1 | MS | 0.018467 | 0.003903 | **0.018467** | 37.7 | MS | 0.018467 | 0.010120 | **0.018467** | 30.0 | MS | 0.018467 | 0.003660 | 0.018483 | **5.74** | MS | 40 | 0.018468 |
| *pb_n5_3* | 5 | 10 | 0.017122 | 23.6 | MS | 0.017122 | 0.006728 | 0.017121 | 40.8 | MS | 0.017122 | 0.015603 | 0.017122 | 27.7 | MS | 0.017122 | 0.004560 | 0.017123 | **4.68** | MS | 29 | **0.017100** |
| *pb_n5_4* | 5 | 10 | **0.014750** | 22.9 | MS | 0.014750 | 0.014624 | **0.014750** | 40.5 | MS | 0.014750 | 0.009179 | **0.014750** | 27.6 | MS | 0.014750 | 0.003219 | **0.014750** | **3.42** | MS | 24 | **0.014750** |
| *pb_n5_5* | 5 | 10 | 0.012163 | 33.0 | MS | 0.012163 | 0.012096 | 0.012163 | 53.4 | MS | 0.012163 | 0.012148 | 0.012163 | 35.0 | MS | 0.012163 | 0.005287 | 0.012162 | **6.09** | MS | 24 | **0.012149** |
| *pb_n5_6* | 5 | 10 | 0.011235 | 36.7 | MS | 0.011235 | 0.005837 | 0.011235 | 111 | MS | 0.011235 | 0.011217 | 0.011235 | 38.2 | MS | 0.011235 | 0.008566 | 0.011235 | **1.71** | MS | 15 | **0.011225** |
| *pb_n5_7* | 5 | 10 | 0.012273 | 31.6 | MS | 0.012273 | 0.003918 | 0.012273 | 42.3 | MS | 0.012273 | 0.012257 | 0.012273 | 33.3 | MS | 0.012273 | 0.009218 | 0.012273 | **2.46** | MS | 20 | **0.012262** |
| *pb_n5_8* | 5 | 10 | 0.017556 | 36.0 | MS | 0.017556 | 0.002707 | 0.017556 | 42.0 | MS | 0.017556 | 0.008683 | 0.017556 | 39.5 | MS | 0.017556 | 0.006314 | **0.017555** | **3.47** | MS | 26 | 0.017556 |
| *pb_n5_9* | 5 | 10 | 0.019140 | 29.1 | MS | 0.019140 | 0.007309 | 0.019140 | 46.2 | MS | 0.019140 | 0.015657 | 0.019140 | 32.9 | MS | 0.019140 | 0.007695 | 0.019138 | **5.59** | MS | 35 | **0.019119** |
| *pb_n5_10* | 5 | 10 | 0.026022 | 20.9 | MS | 0.026022 | 0.006197 | 0.026022 | 42.3 | MS | 0.026022 | 0.014691 | 0.026022 | 23.6 | MS | 0.026022 | 0.005229 | 0.026051 | **5.18** | MS | 32 | **0.025960** |
| *pb_n5_11* | 5 | 10 | 0.011202 | 42.4 | MS | 0.011202 | 0.011167 | 0.011199 | 38.2 | MS | 0.011201 | 0.011188 | 0.011202 | 37.4 | MS | 0.011201 | 0.011110 | 0.011199 | **2.30** | MS | 19 | **0.011190** |
| *pb_n5_12* | 5 | 10 | 0.012122 | 23.7 | MS | 0.012122 | 0.009387 | 0.012121 | 39.0 | MS | 0.012122 | 0.007619 | 0.012122 | 25.3 | MS | 0.012122 | 0.007107 | 0.012121 | **2.55** | MS | 19 | **0.012111** |
| *pb_n5_13* | 5 | 10 | 0.023265 | 32.1 | MS | 0.023265 | 0.005860 | 0.023265 | 43.9 | MS | 0.023265 | 0.009149 | 0.023265 | 33.4 | MS | 0.023265 | 0.004473 | **0.023264** | **5.82** | MS | 40 | 0.023265 |
| *pb_n5_14* | 5 | 10 | **0.013789** | 28.1 | MS | 0.013789 | 0.006102 | **0.013789** | 40.6 | MS | 0.013789 | 0.013385 | **0.013789** | 32.7 | MS | 0.013789 | 0.005103 | **0.013789** | **3.55** | MS | 25 | 0.013790 |
| *pb_n5_15* | 5 | 10 | 0.014578 | 32.9 | MS | 0.014578 | 0.006508 | 0.014578 | 51.4 | MS | 0.014578 | 0.007278 | 0.014578 | 35.3 | MS | 0.014578 | 0.005682 | 0.014577 | **3.28** | MS | 25 | **0.014551** |
| *pb_n5_16* | 5 | 10 | 0.010378 | 31.9 | MS | 0.010378 | 0.006824 | 0.010378 | 93.8 | MS | 0.010378 | 0.010367 | 0.010378 | 35.8 | MS | 0.010378 | 0.007561 | 0.010375 | **1.73** | MS | 15 | **0.010367** |
| *pb_n5_17* | 5 | 10 | 0.011956 | 34.9 | MS | 0.011956 | 0.008863 | 0.011956 | 38.8 | MS | 0.011956 | 0.008161 | 0.011956 | 39.2 | MS | 0.011956 | 0.005857 | 0.011956 | **2.05** | MS | 17 | **0.011940** |
| *pb_n5_18* | 5 | 10 | 0.011166 | 24.6 | MS | 0.011166 | 0.007224 | 0.011166 | 32.1 | MS | 0.011166 | 0.011154 | 0.011166 | 27.5 | MS | 0.011166 | 0.011106 | 0.011166 | **2.41** | MS | 18 | **0.011153** |
| *pb_n5_19* | 5 | 10 | 0.009920 | 2498 | C | - | - | 0.009920 | 1613 | C | - | - | 0.009920 | 36.9 | MS | 0.009920 | 0.008954 | **0.009919** | **0.75** | MS | 8 | 0.009920 |
| *pb_n5_20* | 5 | 10 | 0.019774 | 27.6 | MS | 0.019774 | 0.006149 | 0.019773 | 430 | MS | 0.019773 | 0.010206 | 0.019774 | 29.2 | MS | 0.019773 | 0.004218 | 0.019773 | **5.34** | MS | 34 | **0.019739** |
| *pb_n5_21* | 5 | 10 | **0.009050** | 1402 | C | - | - | **0.009050** | 1503 | C | - | - | 0.009051 | 61.8 | MS | 0.009050 | 0.009022 | 0.009051 | **0.30** | MS | 4 | 0.009051 |
| *pb_n5_22* | 5 | 10 | 0.030611 | 30.5 | MS | 0.030610 | 0.003752 | 0.030610 | 49.3 | MS | 0.030610 | 0.013442 | 0.030611 | 33.0 | MS | 0.030610 | 0.003154 | 0.030638 | **12.4** | MS | 53 | **0.030577** |
| *pb_n5_23* | 5 | 10 | 0.001543 | **27.4** | MS | 0.001543 | 0.001530 | 0.001543 | 38.9 | MS | 0.001550 | 0.001523 | 0.001543 | 40.1 | MS | 0.001543 | 0.001528 | **0.001542** | 2655 | C | 26 | 0.001543 |
| *pb_n6_1* | 6 | 15 | 0.001667 | 49.9 | MS | 0.001667 | 0.000080 | 0.001667 | 43.2 | MS | 0.001949 | 0.000273 | 0.001667 | 71.3 | MS | 0.001667 | 0.000095 | 0.001667 | **2.44** | MS | 17 | **0.001649** |
| *pb_n6_2* | 6 | 15 | 0.001667 | 51.5 | MS | 0.001667 | 0.000078 | 0.001667 | 44.0 | MS | 0.007084 | 0.000164 | 0.001667 | 75.8 | MS | 0.001667 | 0.000197 | 0.001667 | **2.46** | MS | 17 | **0.001661** |
| % best *cpu* | | | | 7.69 | | | | | 0 | | | | | 0 | | | | | 92.31 | | | |

Table 3: Results obtained solving 4 different formulations of HACADP compared with those obtained in [17], *Random Circle* and *Non-Circle instances*

| Instances | | | KKT reformulation | | | | Dom's dual reformulation | | | | Wolfe's dual reformulation | | | | CG | | | [17] | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | $n$ | $n_c$ | *obj* | *cpu* | *slv* | UB | LB | *obj* | *cpu* | *slv* | UB | LB | *obj* | *cpu* | *slv* | UB | LB | *obj* | *cpu* | *slv* | it | *obj* |
| Random Circle | | | | | | | | | | | | | | | | | | | | | | |
| *rpb2_1* | 2 | 1 | **0.000141** | 0.07 | C | - | - | **0.000141** | **0.06** | C | - | - | **0.000141** | **0.06** | C | - | - | **0.000141** | 0.07 | C | 2 | **0.000141** |
| *rpb2_2* | 2 | 1 | **0.000795** | 0.07 | C | - | - | **0.000795** | **0.05** | C | - | - | **0.000795** | **0.06** | C | - | - | **0.000795** | **0.06** | C | 2 | **0.000795** |
| *rpb3_1* | 3 | 2 | **0.000078** | 0.39 | C | - | - | **0.000078** | 2946 | C | - | - | **0.000078** | **0.29** | C | - | - | **0.000078** | 0.45 | C | 6 | **0.000078** |
| *rpb3_2* | 3 | 2 | 0.000515 | 0.63 | C | - | - | **0.000509** | 9.9 | C | - | - | 0.000515 | 0.85 | C | - | - | 0.000515 | **0.62** | C | 6 | 0.000513 |
| *rpb3_3* | 3 | 1 | 0.000114 | 0.11 | C | - | - | 0.000114 | 234 | C | - | - | 0.000114 | 0.11 | C | - | - | 0.000114 | **0.06** | C | 2 | **0.000113** |
| *rpb4_1* | 4 | 1 | **0.000156** | 4.36 | C | - | - | **0.000156** | 4.70 | C | - | - | **0.000156** | 3.65 | C | - | - | **0.000156** | **0.06** | C | 2 | **0.000156** |
| *rpb4_2* | 4 | 3 | 0.001177 | 13.3 | C | - | - | 0.001177 | 24.7 | MS | 0.001206 | 0.001172 | 0.001177 | 28.6 | C | - | - | 0.001177 | **1.28** | C | 7 | **0.001175** |
| *rpb4_3* | 4 | 2 | **0.000202** | 1.55 | C | - | - | **0.000202** | 3.39 | C | - | - | **0.000202** | 6.39 | C | - | - | **0.000202** | **0.17** | C | 2 | **0.000202** |
| *rpb5_1* | 5 | 6 | 0.000413 | 356 | C | - | - | 0.000413 | 39.3 | MS | 0.000413 | 0.000407 | 0.000413 | 319 | C | - | - | 0.000413 | **2.60** | C | 6 | **0.000408** |
| *rpb5_2* | 5 | 3 | 0.000454 | 1323 | C | - | - | 0.000454 | 40.7 | MS | 0.004249 | 0.000327 | 0.000454 | 36.4 | MS | 0.000454 | 0.000430 | 0.000454 | **14.6** | C | 11 | **0.000450** |
| *rpb5_3* | 5 | 8 | 0.000612 | 213 | C | - | - | 0.000612 | 39.2 | MS | 0.001263 | 0.000602 | 0.000612 | 171 | C | - | - | 0.000612 | **6.77** | C | 7 | 0.000613 |
| *rpb6_1* | 6 | 5 | 0.000965 | **44.2** | MS | 0.000965 | 0.000053 | 0.000965 | 48.1 | MS | 0.009621 | 0.000146 | 0.000965 | 58.3 | MS | 0.000965 | 0.000012 | 0.000964 | 794 | C | 16 | **0.000955** |
| *rpb6_2* | 6 | 9 | 0.000858 | 50.7 | MS | 0.001133 | 0.000103 | 0.000858 | 45.4 | MS | 0.018710 | 0.000094 | 0.000858 | 58.8 | MS | 0.000858 | 0.000071 | 0.000858 | **5.60** | C | 6 | **0.000855** |
| *rpb6_3* | 6 | 4 | 0.000694 | **44.0** | MS | 0.000694 | 0.000032 | 0.000694 | 49.5 | MS | 0.006783 | 0.000070 | 0.000694 | 57.9 | MS | 0.000694 | 0.000008 | 0.000694 | 61.5 | C | 11 | **0.000693** |
| *rpb7_1* | 7 | 2 | 0.002199 | 50.7 | MS | 0.000213 | 0 | 0.000213 | 60.6 | MS | 0.000313 | 0.000106 | 0.000213 | 96.0 | MS | 0.000213 | 0.000053 | 0.000213 | **0.49** | C | 6 | **0.000210** |
| *rpb7_2* | 7 | 7 | **0.001161** | 64.5 | MS | 0.001161 | 0 | **0.001161** | 64.8 | MS | 0.003693 | 0.000046 | **0.001161** | 89.0 | MS | 0.001161 | 0 | **0.001161** | **2.46** | MS | 17 | 0.001162 |
| *rpb7_3* | 7 | 9 | 0.002794 | 66.4 | MS | 0.002418 | 0 | **0.001914** | 58.1 | MS | 0.003851 | 0.000040 | **0.001914** | 99.8 | MS | 0.001914 | 0 | **0.001914** | **2.52** | MS | 13 | 0.002637 |
| *rpb8_1* | 8 | 12 | 0.001174 | 99.1 | MS | 0.001237 | 0 | 0.001174 | 82.9 | MS | 0.006447 | 0.000007 | 0.001174 | 93.8 | MS | 0.001174 | 0 | **0.001173** | **3.07** | MS | 15 | 0.001189 |
| *rpb8_2* | 8 | 5 | **0.000370** | 102 | MS | 0.000370 | 0 | **0.000370** | **79.6** | MS | 0.015749 | 0 | **0.000370** | 97.2 | MS | 0.000370 | 0 | **0.000370** | 150 | C | 14 | 0.000373 |
| *rpb8_3* | 8 | 10 | **0.001006** | 97.8 | MS | 0.001006 | 0.000002 | 0.001007 | 84.3 | MS | 0.006782 | 0 | **0.001006** | 98.7 | MS | 0.001006 | 0 | **0.001006** | **7.50** | MS | 28 | 0.001019 |
| % best *cpu* | | | | 0 | | | | | 18.18 | | | | | 18.18 | | | | | 72.73 | | | |
| Non-Circle | | | | | | | | | | | | | | | | | | | | | | |
| *pb6_5* | 6 | 5 | 0.128250 | 59.3 | MS | 0.128250 | 0 | 0.128249 | 40.4 | MS | 0.133770 | 0 | 0.128250 | 146 | MS | 0.128249 | 0 | **0.128233** | **1.50** | MS | 8 | - |
| *pb7_4* | 7 | 4 | 0.001628 | 219 | MS | 0.001672 | 0 | **0.001602** | 71.7 | MS | 0.001602 | 0 | **0.001602** | 327 | MS | 0.001602 | 0 | **0.001602** | **2.89** | MS | 12 | - |
| *pb7_6* | 7 | 6 | 0.001567 | 222 | MS | 0.001672 | 0 | 0.001567 | 75.6 | MS | 0.009484 | 0 | 0.001567 | 324 | MS | 0.001567 | 0 | **0.001566** | **1.85** | MS | 8 | - |
| *pb8_4* | 8 | 4 | 0.002401 | 108 | MS | 0.002548 | 0 | **0.002384** | 81.6 | MS | 0.003082 | 0 | **0.002384** | 261 | MS | 0.002384 | 0 | **0.002384** | **1.73** | MS | 8 | - |
| *pb10_10* | 10 | 10 | 0.127695 | 166 | MS | 0.669673 | 0 | 0.127693 | 90.9 | MS | 0.127693 | 0 | 0.127695 | 270 | MS | 0.127693 | 0 | **0.127677** | **0.53** | MS | 8 | - |
| % best *cpu* | | | | 0 | | | | | 0 | | | | | 0 | | | | | 100 | | | |

## 7 Conclusions

We propose bilevel programming as a suitable approach to model the well-known aircraft deconfliction problem or ADP. In particular, we present two bilevel formulations of the ADP: one based on speed regulation in $k$ dimensions and another where potential conflicts are avoided via heading angle changes in two dimensions. In both cases, the convexity of the lower-level subproblems allows us to derive three different single-level problems respectively, using KKT conditions, Dorn's duality, and Wolfe's duality.

The single level reformulations of both problems are solved by using state-of-the-art solvers, which provide good solutions in reasonable computing time. Alternatively, we propose a cut generation algorithm to solve the bilevel problems. This algorithm, compared with state-of-the-art solvers, obtains the best results for most of the tested instances in few seconds. Numerical results, when compared with other approaches in the literature, are encouraging and stress the potential of the proposed approach.

## Appendix

## A Returning aircraft to original trajectories

The optimal heading angle change $\theta_i^\star$ for each aircraft $i$ is obtained by solving Eq. (14a)–(14c). The trajectory deviation is followed until necessary to guarantee the safety distance, then the aircraft must return to their initial trajectories. Following what is done in [29,17], for each pair of aircraft the convex unconstrained QP Eq. (22) is solved as a post-processing step to return each aircraft to its original flight plan as soon as possible after conflict resolution:

$$\min_{t_{ij}} \left\| \begin{pmatrix} (x_i^0 - x_j^0) + t_{ij}(\cos(\phi_i + \theta_i^\star)v_i - \cos(\phi_j + \theta_j^\star)v_j) \\ (y_i^0 - y_j^0) + t_{ij}(\sin(\phi_i + \theta_i^\star)v_i - \sin(\phi_j + \theta_j^\star)v_j) \end{pmatrix} \right\|^2 . \tag{22}$$

The objective function of the problem Eq. (22) is the relative squared Euclidean distance between aircraft, which is computed using the optimal heading angles of the proposed bilevel problem Eq. (14a)–(14c).

Once the optimal solution $\tau_{ij}^\star$ for problem Eq. (22) is found, we compute

$$T_i^\star := \max_{j:i\neq j} \tau_{ij}^\star \tag{23}$$

as the optimal time for which aircraft $i$ can return to its initial trajectory after the deconfliction (there will be, for each $i$, a different $\tau_{ij}$ for every pair of aircraft $(i,j)$).

Knowing $(x_i(T_i^\star), y_i(T_i^\star))$ and the exit point from the air sector, it is easy to determine the new trajectory each aircraft has to follow in order to go back to its initial trajectory, as shown in Figure 7.

As clarified in [29], when the aircraft are returning to the initial trajectories, new conflicts may occur. In order to ensure a conflict-free situation, the HACADP must be solved again. Sometimes, the maneuver to return to the initial trajectory must start when the aircraft is already close to the boundary of the air sector. This could lead to an angle variation exceeding the bounds. In this case, [29] proposes turning at the maximum bound and sending a warning message to the air traffic controllers of the following sector notifying that the aircraft is arriving in that sector at a different entry point w.r.t. the scheduled one.

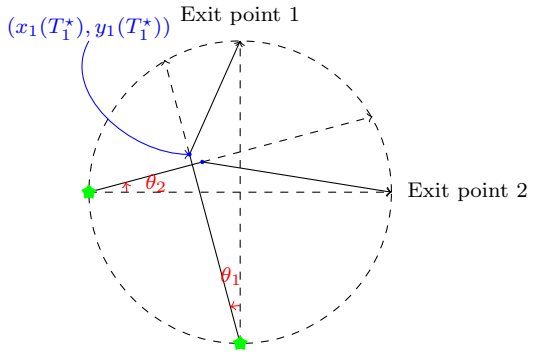


Fig. 7: New trajectories of two aircraft that, after conflict resolution, return to their initial trajectories (dashed lines)

## B Cut dominance

The time per iteration taken by Algorithm 1 and Algorithm 2 increases with the number of cuts added to the formulation. In fact, while solutions of the lower-level subproblems are easily computed in closed form in Step 4 of both algorithms respectively, increasing the number of quadratic constraints (20) and (21) yields a time increase when solving $R_h$. It would therefore be desirable to remove as many unnecessary cuts as possible. Consequently, we study the existence of dominance relationships between cuts in the proposed cut generation algorithms.

**Proposition 5** *Let $(i, j)$ be a pair of aircraft. There is no dominance relationship between any pair of the cuts added for $(i, j)$ by either Algorithm 1 or Algorithm 2.*

*Proof* Let $h, h'$ be two different iterations of the cut generation algorithm in which cuts were added for the pair $(i, j)$. We consider the time instants at which $i$ and $j$ are closest in each iteration, $\tau_{ij}^h$ and $\tau_{ij}^{h'}$, with $\tau_{ij}^h \neq \tau_{ij}^{h'}$. Taking the case of Algorithm 1 (the proof for Algorithm 2 is analogous), we proceed by contradiction and suppose that the cut added in iteration $h$ dominates that of iteration $h'$. That is, for all feasible $q_i, q_j$:

$$\sum_{k\in K}((x_{ik}^0 - x_{jk}^0) + \tau_{ij}^h(q_i v_i u_{ik} - q_j v_j u_{jk}))^2 \geq d^2$$

implies

$$\sum_{k\in K}((x_{ik}^0 - x_{jk}^0) + \tau_{ij}^{h'}(q_i v_i u_{ik} - q_j v_j u_{jk}))^2 \geq d^2,$$

or, equivalently,

$$\sum_{k\in K}((x_{ik}^0 - x_{jk}^0) + \tau_{ij}^h(q_i v_i u_{ik} - q_j v_j u_{jk}))^2 \leq \sum_{k\in K}((x_{ik}^0 - x_{jk}^0) + \tau_{ij}^{h'}(q_i v_i u_{ik} - q_j v_j u_{jk}))^2.$$

In particular, for $q_i, q_j$ equal to the solution obtained at iteration $h'$, $q_i^{h'}, q_j^{h'}$, the inequality

$$\sum_{k\in K}((x_{ik}^0 - x_{jk}^0) + \tau_{ij}^h(q_i^{h'} v_i u_{ik} - q_j^{h'} v_j u_{jk}))^2 \leq \sum_{k\in K}((x_{ik}^0 - x_{jk}^0) + \tau_{ij}^{h'}(q_i^{h'} v_i u_{ik} - q_j^{h'} v_j u_{jk}))^2$$

must hold. Being $\tau_{ij}^{h'}$ a minimizer of the function

$$f(t) := \sum_{k \in K} ((x_{ik}^0 - x_{jk}^0) + t(q_i^{h'} v_i u_{ik} - q_j^{h'} v_j u_{jk}))^2,$$

it has to be:

$$\sum_{k \in K} ((x_{ik}^0 - x_{jk}^0) + \tau_{ij}^h (q_i^{h'} v_i u_{ik} - q_j^{h'} v_j u_{jk}))^2 = \sum_{k \in K} ((x_{ik}^0 - x_{jk}^0) + \tau_{ij}^{h'} (q_i^{h'} v_i u_{ik} - q_j^{h'} v_j u_{jk}))^2.$$

Since $\tau_{ij}^h \neq \tau_{ij}^{h'}$, it yields that $f(t)$ attains its minimum at two different points. This is only possible if $f(t)$ is constant for all $t$, i.e., if $q_i^{h'} v_i u_{ik} = q_j^{h'} v_j u_{jk}$. But, if a cut is added for $(i, j)$ at iteration $h'$ is because the separation distance was violated also at $t = 0$ (since $f$ is constant), something that we discarded by assumption. □

Proposition 5 ensures that a cut added in a certain iteration for a pair of aircraft will not be dominated by any cut added in future iterations for the same pair. However, whether several cuts together dominate a single cut is a question that remains open. Similarly, the study of potential dominance between cuts involving conflicting triplets of aircraft in special cases would be worth-studying for future work.

## References


1. J. Villiers: Automatisation du controle de la circulation aèrienne: "ERASMUS", une voie conviviale pour franchir le mur de la capacitè, *Institut du Transport Aèrien* **58** (2004)
2. F. Drogoul, P. Averty, R Weber: ERASMUS Strategic Deconfliction to Benefit SESAR, *Proc. 8th USA/Europe Air Traffic Management Research and Development Seminar, Napa, USA*, 1–10 (2009)
3. https://ec.europa.eu/transport/modes/air/sesar_en
4. M. Lao, J. Tang: Cooperative Multi-UAV Collision Avoidance Based on Distributed Dynamic Optimization and Causal Analysis, *Applied Sciences* **7(1)**, 83 (2017)
5. D. Claes, K. Tuyls: Multi robot collision avoidance in a shared workspace, *Autonomous Robots* **42(8)**, 1749–1770 (2018)
6. C. G. Co, J.M.A. Tanchoco, A review of research on AGVS vehicle management, *Engineering Costs and Production Economics* **21(1)**, 35–42 (1991)
7. M. Cerulli, C. D'Ambrosio, L. Liberti: Flying safely by bilevel programming, *Advances in Optimization and Decision Science for Society, Services and Enterprises: ODS, Genoa, Italy, September 4-7, 2019* – AIRO Springer Serie **3**, 197–206 (2019)
8. D. Rey, C. Rapine, S. Constans, R. Fondacci: A Mixed Integer Linear Model for Potential Conflict Minimization by Speed Modulations, *ICRAT 2010. Fourth International Conference on Research in Air Transportation* (2010)
9. D. Rey, C. Rapine, V.V. Dixit, S. T. Waller: Equity-Oriented Aircraft Collision Avoidance Model, *IEEE Transactions on Intelligent Transportation Systems* **16(1)**, 172–183 (2015)
10. S. Cafieri, N. Durand: Aircraft Deconfliction with Speed Regulation: New Models from Mixed-integer Optimization, *Journal of Global Optimization* **58(4)**, 613–629 (2014)
11. S. Cafieri, C. D'Ambrosio: Feasibility pump for aircraft deconfliction with speed regulation, *Journal of Global Optimization* **71(3)**, 501–515 (2018)
12. A. Vela, S. Solak, W. Singhose and J. Clarke: A mixed integer program for flight-level assignment and speed control for conflict resolution, *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference, Shanghai*, 5219–5226 (2009)
13. A. Alonso-Ayuso, L. F. Escudero, F. J. Martín-Campo: Collision avoidance in air traffic management: A mixed-integer linear optimization approach, *IEEE Transactions on Intelligent Transportation Systems* **12(1)**, 47–57 (2010)

14. F.H. Dias, H. Hijazi, D. Rey: Disjunctive linear separation conditions and mixed-integer formulations for aircraft conflict resolution by speed and altitude control, *arXiv preprint arXiv:1911.12997* (2019)
15. K. D. Bilimoria: A Geometric Optimization Approach to Aircraft Conflict Resolution, *18th Applied Aerodynamics Conference*, (2000)
16. L. Pallottino, E. Feron, A. Bicchi: Conflict resolution problems for air traffic management systems solved with mixed integer programming, *IEEE transactions on intelligent transportation systems* **3(1)**, 3–11 (2002)
17. S. Cafieri, R. Omheni: Mixed-integer nonlinear programming for aircraft conflict avoidance by sequentially applying velocity and heading angle changes, *European Journal of Operational Research* **260(1)**, 283–290 (2017)
18. E. Frazzoli, Z.-H. Mao, J.-H. Oh, E. Feron: Resolution of Conflicts Involving Many Aircraft via Semidefinite Programming, *Journal of Guidance, Control and Dynamic* **24(1)**, 79–86 (2001)
19. D. Rey, H. Hijazi: Complex number formulation and convex relaxations for aircraft conflict resolution, *IEEE 56th Annual Conference on Decision and Control*, 88–93 (2017)
20. O. Stein: How to solve a semi-infinite optimization problem, *European Journal of Operational Research* **223**, 312—320 (2012)
21. S. Dempe, V. Kalashnikov, G.A. Prez-Valds, N. Kalashnykova: Bilevel Programming Problems: Theory, Algorithms and Applications to Energy Networks – Springer, Berlin (2015)
22. C. Audet, P. Hansen, B. Jaumard, G. Savard: Links Between Linear Bilevel and Mixed 0–1 Programming Problems, *Journal of Optimization Theory and Applications* **93(2)**, 273—300 (1997)
23. T. Kleinert, M. Labbé, F. Plein, M. Schmidt: There's No Free Lunch: On the Hardness of Choosing a Correct Big-M in Bilevel Optimization, *Optimization online* **7172**, (2019)
24. W. S. Dorn: Self-Dual Quadratic Programs, *Journal of the Society for Industrial and Applied Mathematics* **9(1)**, 51–54 (1961)
25. W. S. Dorn: Duality in quadratic programming, *Quarterly of Applied Mathematics* **18(2)**, 155–162 (1960)
26. P. Wolfe: A duality theorem for non-linear programming, *Quarterly of Applied Mathematics* **19**, 239–244 (1961)
27. T. Kleinert, M. Labbé, F. Plein, M. Schmidt: Closing the Gap in Linear Bilevel Optimization: A New Valid Primal-Dual Inequality, *preprint*, (2020)
28. M.H. Zare, J.S. Borrero, B. Zeng: A note on linearized reformulations for a class of bilevel linear integer problems, *Annals of Operations Research* **272**, 99-–117 (2019)
29. A. Alonso-Ayuso, L. F. Escudero, F. J. Martín-Campo: Exact and Approximate Solving of the Aircraft Collision Resolution Problem via Turn Changes, *Transportation Science* **50**, 263–274 (2016)
30. S. Fang, C. Lin, S. Wu: Solving quadratic semi-infinite programming problems by using relaxed cutting-plane scheme, *Journal of Computational and Applied Mathematics* **129**, 89—104 (2001)
31. R. Fourer, D. M. Gay, B. W. Kernighan: AMPL: A Modeling Language for Mathematical Programming, *Cengage Learning* (2002)
32. N.V. Sahinidis and M. Tawarmalani: *BARON 7.2.5: Global Optimization of Mixed-Integer Nonlinear Programs*, User's Manual (2005)
33. P.E. Gill: *User's guide for SNOPT version 7.2* (2006)
34. P. Belotti, J. Lee, L. Liberti and A. Wächter: Branching and bounds tightening techniques for non-convex MINLP, *Optimization Methods and Software* **24(4-5)**, 597–634, (2009)
35. A. Wächter and L. Biegler: On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming, *Mathematical Programming* (2006)