# La programmation DC et la méthode Cross-Entropy pour certaines classes de problèmes en finance, affectation et recherche d'informations : codes et simulations numériques

Duc Manh Nguyen

THÈSE

pour l'obtention du titre de

**DOCTEUR DE L'INSTITUT NATIONAL DES SCIENCES**

**APPLIQUÉES DE ROUEN**

(arrêté ministériel du 7 août 2006)

**Spécialité : Mathématiques Appliquées**

Présentée et soutenue par

**NGUYEN DUC Manh**

— Titre de la thèse —

# La Programmation DC et la Méthode Cross-Entropy pour certaines classes de problèmes en Finance, Affectation et Recherche d'Informations

CODES ET SIMULATIONS NUMÉRIQUES

Date de soutenance: le 24 février 2012

Membres de Jury:

| | | |
|---|---|---|
| Président | Adnan Yassine | *Professeur, Université du Havre* |
| Rapporteurs | Nguyen Van Thoai | *Professeur, Université de Trier, Allemagne* |
| | Ibrahima Sakho | *Professeur, Université de Lorraine* |
| Examinateurs | Ali Khenchaf | *Professeur, ENSTA Bretagne* |
| | Frédéric Dambreville | *Docteur HDR, Direction Générale de L'Armement* |
| | Thierry Lecroq | *Professeur, Université de Rouen* |
| Directeurs de thèse | Pham Dinh Tao | *Professeur, INSA de Rouen* |
| | Le Thi Hoai An | *Professeur, Université de Lorraine* |

## REMERCIEMENTS

La présente thèse a été réalisée au sein de du Laboratoire de Mathématiques (LMI) de l'Institut National des Sciences Appliquées (INSA), France, sous la co-direction des Professeur PHAM DINH Tao, LMI INSA de Rouen et Professeur LE THI Hoai An, Directrice de l'Équipe Algorithmique et Optimisation, LITA, de l'Université de Lorraine,

Je remercie sincèrement en premier lieu Professeur PHAM DINH Tao et Professeur LE THI Hoai An, mes directeurs de thèse pour leurs précieux conseils, leur aide constant ainsi que pour la confiance qu'ils m'ont accordée durant la préparation de la thèse.

J'adresse mes vifs remerciements à Monsieur Nguyen Van Thoai, Professeur à l'Université de Trier, Allemagne et Monsieur Ibrahima Sakho, Professeur à l'Université de Lorraine de m'avoir fait l'honneur d'accepter la charge du rapporteur de ma thèse.

Je remercie Monsieur Ali Khenchaf, Professeur à l'ENSTA, Brestagne, Monsieur Adnan Yassine, Professeur à l'Université du Havre, Monsieur Frédéric Dambreville, Docteur HDR, Direction Générale de L'Armement (DGA), et Monsieur Thierry Lecroq, Professeur à l'Université de Rouen pour avoir participé à juger mon travail.

Je me permets également de remercier le Ministère de l'Éducation Vietnamienne pour l'aide financière qu'elle m'a attribuée.

Mes remerciements s'adressent également au Département de Mathematiques l'École Normale Supérieure de Hanoï, particulièrement Monsieur NGUYEN Hac Hai pour leur soutien et leur encouragement.

A vous mes parents, je dis un immense merci. Je vous suis infiniment reconnaissent pour votre soutien et vos encouragements.

Je remercie plus particulièrement Thoa mon épouse pour sa patience, sa compréhension, son amour et l'équilibre qu'elle a apporté dans ma vie en acceptant de faire de moi son époux. Ces éléments ont été des alliés de poids pendant ces années.

Je remercie tous mes collègues et mes amis rencontrés à Rouen et à Metz pour les moments agréables lors de mon séjour en France.

Enfin, je remercie tous ceux qui m'ont aidé de près ou de loin et tous ceux qui m'ont motivé.

# Contents

# General introduction

It is widely recognized that the optimization approach plays a key role in modeling and solving real-life problems in different fields of applied sciences.

A general optimization problem is of the form

$$(P) \quad \min\{f(x) : x \in S \subset \mathbb{R}^n\}.$$

Convex programming corresponds to both convexity of the objective function $f$ and the constraint set $S$. Convex programs are featured by the fact that local and global optimal solutions are identical. As a crucial result, optimality conditions (the famous KKT conditions) are available and they serve to devise (iterative) convex optimization algorithms for solving these problems. The golden age of modern convex analysis and convex programming is situated in the period 1960-1985, where a powerful arsenal of theoretical and algorithmic tools was developed. This period is followed by extensive works on interior point methods and their applications in semidefinite programming (SDP) [94, 142, 201], during the past fifteen years. Under usual precautions, especially for the large-scale setting, one can say, at the present time, that convex programs can be solved by well suitable algorithms.

The passage to nonconvex programming generates enormous difficulty, due to the following fact:

- Distinction between local and global optimal solutions: there is no verifiable global optimality conditions, and consequently there is no iterative algorithms converging to global solutions.
- Computing a global solution of a high-dimensional nonconvex program then is a grail request for mathematicians, especially optimizers.
- The last twenty-five years have seen the development of extensive research in nonconvex programming and global optimization. The reason is quite simple: most real-world optimization problems are of nonconvex nature. Moreover, industrialists have begun to replace convex models by nonconvex ones, which are more complex but more reliable and especially more economic.

However, several approaches for solving nonconvex optimization problems have been proposed in the literature. We can classify them in two main categories:

1. The heuristic methods: they are developed to find acceptably good solutions but ignore whether the solutions can be proven to be optimal. For instance: the method of simulated annealing (S. Kirkpatrick et al. in 1983 [93], and by Vlado Cerný in 1985 [36]), genetic algorithm (J.H. Holland in 1975 [81]), tabu search (F. Glover [67]), the method of variable neighborhood search (P. Hansen and N. Mladenovic in 1997 [130]), the cross-entropy algorithm (R.Y. Rubinstein [176–178]), etc.

2. The deterministic methods: there are two types of different but complementary approaches for nonconvex programming.

   (a) **The global approaches**: the approaches are developed according to the spirit of the combinatorial optimization, but with the difference that one works in the continuous framework. In these approaches, optimal solutions are located by using the approximation methods, for instance, cutting techniques, decomposition methods, branch and bound, etc. It was Hoang Tuy who has incidentally put forward by his pioneering paper in 1964 [75]: the new global optimization concerning convex maximization over a polyhedral convex set. Among the most important contributions to these approaches, it is worth citing the ones by Hoang Tuy, R. Horst, H. Konno, P. Pardalos, Le Dung Muu, Le Thi Hoai An, Nguyen Van Thoai, Phan Thien Thach and Pham Dinh Tao [76–78, 84, 85, 95, 115, 116, 119, 133, 134, 154, 160–162]. Although these approaches are known to be heavy and costly to implement, especially in very large-scale settings, they have successfully established some standard methodologies for global optimization, which are the foundation for many other researches.

   (b) **The convex approaches**: they are based on convex analysis tools, the DC duality and the local optimal conditions in DC programming. Here the DC programming (Difference of two convex functions) and DCA (DC Algorithms) play the central role because most of nonconvex optimization problems are formulated/reformulated as DC. They are introduced by Pham Dinh Tao in 1985 in their preliminary form and extensively developed by Le Thi Hoai An and Pham Dinh Tao since 1994 to become now classic, and widely used by researchers and practitioners in various fields of applied sciences (see [2, 33, 90, 110–114, 124, 152–157, 163, 185, 194, 209, 211]).

   DC programming and DCA aim to solve a general DC program that takes the form

$$\alpha = \inf\{f(x) := g(x) - h(x) : x \in \mathbb{R}^p\} \quad (P_{dc}) \tag{1}$$

   where $g, h$ are lower semicontinuous proper convex functions on $\mathbb{R}^p$. Such a function $f$ is called DC function, and $g - h$, DC decomposition of $f$ while $g$ and $h$ are DC components of $f$. The construction of DCA involves DC components $g$ and $h$ but not the function $f$ itself: each iteration $k$ of DCA consists of computing

$$y^k \in \partial h(x^k), x^{k+1} \in \arg\min\{g(x) - h(x^k) - \langle x - x^k, y^k\rangle : x \in \mathbb{R}^p\} \quad (P_k).$$

   Hence, for a DC program, each DC decomposition corresponds to a different version of DCA. Since a DC function $f$ has an infinite number of DC decompositions which have crucial impacts on the qualities (speed of convergence, robustness, efficiency, globality of computed solutions,...) of DCA, the search for a "good" DC decomposition is important from algorithmic point of views. Moreover, despite its local character, DCA with a good initial point can converge to global solutions.

Finding a "good" initial point is then also an important stage of DCA. How to develop an efficient algorithm based on the generic DCA scheme for a practical problem is thus a judicious question to be studied, and the answer depends on the specific structure of the problem being considered.

In this thesis, we consider some optimization problems in Search Theory, Assignment and Portfolio Management: the problem of planning a multisensor multizone search for a (moving) target, the Nonlinear UAV (Unmanned Aerial Vehicles) Task Assignment Problem, the Value-at-Risk constrained Optimization problem and the Multidimensional Assignment Problem (MAP). They are large-scale nonconvex optimization problems. We focus on developing deterministic and heuristic approaches for their solution:

- The deterministic approaches are based on **DC programming and DCA**. Our motivation is due to their robustness and efficiency compared to existing methods, their adaptation to the structures of treated problems and their ability to solve large-scale real world nonconvex programs, such as Transport-Logistics, Telecommunication, Finance, Data Mining and Machine Learning, Cryptology, Mechanics, Image Processing, Robotic & Computer Vision, Petrochemistry, Optimal Control, etc.

- The heuristic approaches are based on **the Cross-Entropy (CE) method**. The CE method was originally developed in [176] for an adaptive networks, where an adaptive variance minimization algorithm for estimating probabilities of rare events for stochastic networks was presented. It was modified in [177, 178] to solve optimization problems. It has been demonstrated that this method is particularly relevant for solving both continuous multi-extremal and combinatorial optimization problems. Several recent publications demonstrate the power of the CE method as simple and efficient approach for many applications such as Telecommunication Systems, Buffer Allocation, Vehicle Routing, DNA Sequence Alignment, Machine Learning, etc. In fact, when deterministic methods failed to find the optimal solution within a reasonable time, in most cases the CE method finds a fairly good solution more quickly. This motivates us to investigate the CE method for our considered problems.

  However, to apply the Cross-Entropy method, the most important point is how to construct a family of distributions on the feasible set of the optimization problem such that updating the parameters could be done as easily as possible. Depending on the structure of feasible sets of our considered problems, we design appropriate families of distributions such that the Cross-Entropy method could be applied efficiently.

The main contributions of the thesis:

- We develop a *deterministic continuous optimization approach* based on DC programming and DCA for solving the problem of **planning a multisensor multizone search for a target**.

- When the **target is moving (Markovian)**, the above problem becomes very complicated because of the huge number of possible target trajectories. We use the forward-backward split technique (FAB) introduced by Brown [29] to split the problem into the sub-problems with the "static" target, then treat the sub-problems via the DC programming and DCA as above. Finally, we obtain a combination of the FAB and the DCA to the case of moving target.

- We propose two approaches for solving **the Nonlinear UAV (Unmanned Aerial Vehicles) Task Assignment Problem**. The first approach is the Cross-Entropy (CE) method, in which we present a family of discrete distributions characterized by probability matrices on the feasible sets of this problem. The second one is the Branch and Bound algorithm, where the DCA is used to compute the lower bounds.

- We develop both deterministic and heuristic approaches for solving **the Value-at-Risk constrained Optimization problem**. In the first approach, we reformulate the problem as a polyhedral DC program by using an exact penalty technique and propose DCA for its solution. In the second one, we introduce an appropriate family of continuous distributions based on the family of exponential distributions on the feasible sets of this problem to apply the Cross-Entropy method. The ability of these methods to solve large scale problems is showed by firstly testing on the 1304 daily returns for 43 assets of the Eurostoxx50 index (from January 1, 2003 to December 31, 2007). Then, they are tested on the empirical distribution of 11 years (from January 1, 2000 to December 31, 2010) of daily data, i.e., 2759 scenarios of the 87 assets comprising the NYSE US 100 index.

- We propose a Cross-Entropy (CE) algorithm for solving **the Multidimensional Assignment Problem (MAP)**, where a family of discrete distributions on the feasible set of MAP was designed. The efficiency of this method has been shown by testing for the large-scale problems, for instance, the MAP with 5 dimensions and there are 20 elements in each dimension, which is equivalent to a 0-1 linear program with 3.2 millions binary variables and 100 constraints.

- After developing mathematical approaches, we implement the proposed algorithms in C/C++, Matlab, etc., to give numerical simulations:

  - The DCA for the problem of planning a multisensor in multizone search for a target.
  - The combination of the forward-backward split technique and DCA for the above problem but in the case of moving target.
  - A Branch and Bound algorithm for the nonlinear UAV Task Assignment Problem.
  - The cross-entropy algorithm for the Nonlinear UAV Task Assignment Problem.
  - The cross-entropy algorithm for the Value-at-Risk constrained Optimization problem.

- The DCA for the Value-at-Risk constrained Optimization problem.
- The cross-entropy algorithm for the Multidimensional Assignment Problem.

The thesis is divided in two parts:

1. The first one consists of three Chapters 1 & 2 & 3. In the first chapter, we introduce some optimization problems in Search Theory, Assignment and Portfolio Management, which we study in this thesis. The background of DC programming and DCA is briefly presented in the second one, while the methodology of the Cross-Entropy method is given in the third one.

2. Part 2 is devoted to the resolutions of our considered problems in Search Theory, Assignment and Portfolio Management.

**PUBLICATIONS**

[1] Le Thi Hoai An, Nguyen Duc Manh and Pham Dinh Tao (2012), *Globally solving a Nonlinear UAV Task Assignment Problem by stochastic and deterministic optimization approaches,* Optimization Letters 6(2): 315-329.

[2] Nguyen Duc Manh, Le Thi Hoai An and Pham Dinh Tao, *A Cross-Entropy Method for Value-at-Risk Constrained Optimization*, Intelligent Information and Database Systems. N. Nguyen, C.-G. Kim and A. Janiak (Eds.), Springer Berlin / Heidelberg. 6592: 442-451 (2011).

[3] Nguyen Duc Manh, Le Thi Hoai An and Pham Dinh Tao, *A Cross-Entropy method for Nonlinear UAV Task Assignment Problem*, 2010 IEEE-RIVF International Conference on Computing and Communication Technologies Research, Innovation and Vision for the Future, Hanoi, Vietnam, November 1-4, 2010, pp. 267-271.

[4] Le Thi Hoai An, Nguyen Duc Manh and Pham Dinh Tao, *A deterministic optimization approach for planning a multisensor multizone search for a target.*
Submitted to Computers & Operations Research (August 2011)

[5] Le Thi Hoai An, Nguyen Duc Manh and Pham Dinh Tao, *A fast and scalable Cross-Entropy Method for Value-at-Risk constrained Optimization.*
Submitted to Journal of Global Optimization (August 2011)

[6] Nguyen Duc Manh, Le Thi Hoai An and Pham Dinh Tao, *Solving the Multidimensional Assignment Problem via the Cross-Entropy method.*
Submitted to Journal of Combinatorial Optimization (January 2012)

**ARTICLES IN PREPARATION**

[1] Le Thi Hoai An, Nguyen Duc Manh and Pham Dinh Tao, *A deterministic optimization approach for planning a multisensor multizone search for a moving target.*

[2] Le Thi Hoai An, Nguyen Duc Manh and Pham Dinh Tao, *A new method for Value-at-Risk constrained Optimization using the DC programming and DCA.*

**PRESENTATIONS**

1. *A deterministic optimization approach for planning a multisensor multizone search for a target*, 23rd European Conference on Operational Research, Bonn, July 2009.

2. *A deterministic optimization approach for planning a multisensor multizone search for a moving target*, The 3rd Asian Conference on Intelligent Information and Database Systems, Hue, Vietnam, March 2010.

3. *A deterministic optimization approach for planning a multisensor multizone search for a moving target*, Workshop: Optimization and Learning: Theory, Algorithms and

Applications, Metz, France, 2010.

4. *A new approach for solving Value-at-Risk constrained Optimization using the DC programming and DCA*, 24rd European Conference on Operational Research, Lisbon, July 11 - 14, 2010.

5. *A Cross-Entropy method for Nonlinear UAV Task Assignment Problem*, IEEE-RIVF International Conference on Computing and Communication Technologies Research, Innovation and Vision for the Future, Hanoi, Vietnam, November 2010.

6. *A Cross-Entropy method for Value-at-Risk constrained Optimization*, 12e congrès annuel de la Société française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF), Saint Etienne, France, March 2011.

7. *A Cross-Entropy method for Nonlinear UAV Task Assignment Problem*, Journée de Doctorat SPMII 2011, Université de Rouen, France, April 2011.

8. *A Cross-Entropy method for Value-at-Risk constrained Optimization*, The 3rd Asian Conference on Intelligent Information and Database Systems, Daegu, Korea, April 2011.

9. *Solving the Multidimensional Assignment Problem via the Cross-Entropy method*, Workshop: Optimization and Learning: Theory, Algorithms and Applications, Metz, France, May 2011.

# Part I

# Fundamentals of DC programming&DCA and the Cross Entropy method

# Chapter 1

# Introduction to some classes of nonconvex programs in Search Theory, Assignment and Portfolio Management

The purpose of this chapter is to introduce some classes of nonconvex programs in Search Theory, Assignment and Portfolio Management, which will be considered in this thesis. The importance, the difficulties encountered when solving them and the related researches in the literature will also be presented.

## 1.1 Search Theory

Search theory is defined by Cadre and Soiris [32] as a discipline that treats the problem of how a missing object can be searched optimally, when the amount of searching time is limited and only probabilities of the possible position of the missing object are given. The theory of how to search for missing objects has been a subject of serious scientific research for more than 50 years. It is a branch of the broader applied science known as operations research [59].

In fact, Search theory was first established during World War II by the work of B. O. Koopman and his colleagues [101] in the Antisubmarine Warfare Operations Research Group (ASWORG). The applications of search theory were firstly made on military operations [191]. Koopman [100] stated that the principles of search theory could be applied effectively to any situation where the objective is to find a person or object contained in some restricted geographic area. After military applications, it was also applied to different problems such as surveillance, explorations, medicine, industry and search and rescue operations [72]. The aim of searching in the context of Aeronautical Search and Rescue (ASAR), for instance, is to find the missing aircraft effectively and as quickly as possible with the available resources [190].

In this thesis, we consider two problems in Search Theory. The first one is a bi-level problem:

planning a multi-sensor multi-zone search for a target. It can be described as follows: suppose that a space of search is partitioned into zones of reasonable size. A unique sensor must be able to explore efficiently a whole zone. Each zone is itself partitioned into cells. A cell is an area in which every points have the same properties, according to the difficulty of detection (altitude, vegetation, etc.). Each sensor has its own coefficient of visibility over a cell. The visibility coefficients depend also on the kind of the target that is searched. Here, there is a unique target to detect. The objective is alloting sensors to search zones and finding the search resources sharing of multisensor in multizone so as to maximize the probability of detection of a target.

The optimization model of this problem is hierarchical:

- At upper level: finding the best allotment of sensors to search zones (a sensor is allotted to a unique zone);

- At lower level: determining the best resource sharing for every sensor, in order to have an optimal surveillance over the allotted zone.

At the upper level, the objective function can be nonconvex or implicitly defined via an algorithm applied to the lower level. This makes the problem very hard. In [186], Simonin et al. have proposed a hierarchical approach for solving this problem where a cross-entropy (CE) algorithm [28, 41, 180] has been developed for the upper level while an optimization method based on the algorithm of de Guenin [70] for detecting a stationary target has been used in the lower level. Besides this paper, we do not find in the literature the works considering this problem.

The second one is an extension of the first one, where the target is moving (Markovian). We consider a multi period search of a moving target. This means that information about the sensors and the target will be now indexed by time (the period index). The target prior is now trajectorial and we shall consider here a Markovian (target) prior. Furthermore, assuming that sensors act independently at the cell level. The target is said undetected for this multiperiod search if it has not been detected at any period of the search. The objective is alloting sensors to search zones and finding the search resources sharing of multisensor in multizone at each time period so as to mimimize the probability of non-detection of a target.

This problem, in general, is very complicated because of the huge number of possible target trajectories. For a unique sensor, the problem has been theoretically solved in [189, 193]; while extensions to double layered constraints have been considered in [80]. In practice, all feasible algorithms are based on a forward-backward split introduced by Brown [29]. Similar procedures are also much employed in order to estimate Hidden Markov Models parameters (see e.g. [57]).

## 1.2 Assignment Problem

### 1.2.1 Nonlinear Unmanned Aerial Vehicles (UAVs) Task Assignment Problem

A growing number of applications require the coordination of multiple autonomous agents to accomplish a team goal. Many of these efforts utilize unmanned aerial vehicles (UAVs) due to the unique capabilities they provide. In a growing number of these applications, agents must make both tactical and practical decisions autonomously. This is particularly true of systems involving teams of agents which are too complicated to be controlled or efficiently monitored by a human operator.

In fact, the prospect of building unmanned aerial vehicles is not new by any standard [66]. For most of the twentieth century, one has investigated the feasibility of building unmanned aerial vehicles and their potential value in military operations. A principal reason for the interest in UAVs was the desire to reduce the risk to humans in combat, but it also was to perform military missions in a more efficient and less costly fashion than has historically been the case with manned vehicles. Recently, unmanned aerial vehicles (UAVs) can be used for various civilian and military tasks. There has been considerable interest in making these unmanned vehicles completely autonomous, giving rise to the research area of *unmanned autonomous vehicles*. These are usually seen as rather simple vehicles, acting cooperatively in teams to accomplish difficult missions in dynamic, poorly known or hazardous environments [4, 14–16, 18, 25, 43–46, 48, 87, 89, 123, 127, 129, 131, 146, 164, 165, 183, 184, 210].

The mission to be accomplished by a group of UAVs usually involves completing a set of tasks spread over an extended region. The UAVs must reach each task location -possibly under temporal order constraints- and accomplish it while avoiding a spatially distributed set of threats or obstacles. For instance, in a military search-and-destroy mission, the tasks may correspond to targets to be attacked, and the threats may be enemy radar or anti-aircraft batteries. One of the primary challenges, then, is to assign the UAVs to the known tasks/targets and to plan paths for all UAVs such that the overall mission completion time is minimized and the UAVs are exposed to as little threat as possible. This is, in fact, a very complex optimization problem known to be NP-complete. It is very similar to the well-known vehicle routing problem (VRP), for which several heuristic methods have been presented [182, 195].

Typically, a UAV coordination problem includes three sub-problems [18], which are all computationally intensive optimization problems:

1. Determining the team composition: teams are formed and group goals are assigned to each team.

2. Performing the task assignment: tasks that achieve the group goals are assigned to each team member.

3. UAV trajectory optimization: a path is designed for each team member that achieves their tasks while adhering to spatial constraints, timing constraints, and the dynamic capabilities of the aircraft.

The coordination plan is designed to minimize some cost, such as the completion time or the probability of mission failure. The overall control system then monitors the execution of the coordinated plan, and reacts to changes in the operation, the environment, or the goals.

Several previous studies have investigated methods of trajectory planning for coordination and control. Trajectory generation methods include the use of Voronoi diagrams [128], adaptive random search algorithms [106], model predictive control [187], mixed-integer linear programming [17, 171], and the approximate method that yields a fast estimate of the finishing times for the UAV trajectories [18].

The task assignment problem, which is a decision-making process, is normally solved by using integer (or mixed-integer linear) programming techniques [47, 141, 171]. In this approach, the problem is solved as a deterministic optimization problem with known parameters. Since the MILP is NP-hard, it suffers from poor scalability although the solutions preserve global optimality. Recently, uncertainty is taken into account in terms of optimization parameters. The uncertainty in this data can come from many sources, and it is well known that it can have a significant impact on the performance of an optimization-based planner. Mitigating the effect of the uncertainty in this type of optimization problem has recently been addressed by numerous researchers [19, 22–24, 104, 104, 136, 139]. For example, Ben-Tal and Nemirovski [19] discuss the issue of robust feasibility of linear programs, and Bertsimas et al. [23] consider the problem of finding robust solutions to linear programs under more general norms, which extends the research of [139]. In [136], Mulvey et al. introduce general robust optimization techniques that embed higher order statistics from the uncertainty model. In [22, 24], the authors develop techniques that hedge against worst-case performance loss and maintain robust feasibility in the presence of uncertainty in integer optimizations. Solutions are presented with both ellipsoidal and polyhedral uncertainty sets, and the authors show that the robust equivalent of some uncertain integer programs can be found by solving a finite number of new deterministic integer programs. In [104, 105], the authors introduce Conditional Value at Risk (CVaR) as a scenario-based resource allocation problems solved by generating a finite (but possibly large) number of realizations, and finding the assignment that minimizes the conditional expectation of performance loss.

In this thesis, we consider a task allocation model where we seek to assign a set of $m$ UAVs to a set of $n$ tasks in an optimal way. The optimality is quantified by target scores. The mission is to maximize the target score while satisfying capacity constraints of both the UAVs and the tasks. The scoring scheme defining effectiveness in our work is a nonlinear function. More precisely, this problem is an integer nonlinear programming problem for which the classical solution method can not be used.

### 1.2.2 The Multidimensional Assignment Problem (MAP)

The multidimensional assignment problem (MAP) is a higher dimensional version of the linear assignment problem, where we find tuples of elements from given sets, such that the total cost of the tuples is minimal. Denoting $n_k$ the number of elements in dimension $k$ of $d$ dimensions, $n_1 \leq n_k, k = 2, ..., d$, the MAP can be expressed as an integer program as follows

$$\begin{cases} \min \sum\limits_{i_1=1}^{n_1} ... \sum\limits_{i_d=1}^{n_d} c_{i_1...i_d} x_{i_1...i_d} \\ \text{s.t. } \sum\limits_{i_2=1}^{n_2} ... \sum\limits_{i_d=1}^{n_d} x_{i_1...i_d} = 1, i_1 = 1, 2, ..., n_1, \\ \quad \sum\limits_{i_1=1}^{n_1} ... \sum\limits_{i_{k-1}=1}^{n_{k-1}} \sum\limits_{i_{k+1}=1}^{n_{k+1}} ... \sum\limits_{i_d=1}^{n_d} x_{i_1...i_d} \leq 1, i_k = 1, 2, ..., n_k, k = 2, ..., d-1, \\ \quad \sum\limits_{i_1=1}^{n_1} ... \sum\limits_{i_{d-1}=1}^{n_{d-1}} x_{i_1...i_d} \leq 1, i_d = 1, 2, ..., n_d, \\ \quad x_{i_1...i_d} \in \{0, 1\}. \end{cases} \tag{1.1}$$

An equivalent formulation of this problem, which offers us several approaches to find an optimal assignment is injection formulation

$$\begin{cases} \min \sum\limits_{i=1}^{n_1} c_{i,\pi_1(i),...,\pi_{d-1}(i)} \\ \text{s.t. } \pi_k : \{1, 2, ..., n_1\} \to \{1, 2, ..., n_{k+1}\} \text{ is injective}, \forall k = 1, 2, .., d-1. \end{cases} \tag{1.2}$$

The MAP was first introduced by Pierskalla (1968) [147], and since then has found numerous applications in the areas of data association [7], image recognition [202], multisensor multitarget tracking [132, 167], tracking of elementary particles [169], etc. For a survey of the MAP and its applications, see [30, 31, 35, 61, 143, 188].

It is worth to note that while the linear assignment problem is solvable in polynomial time, the MAP is known to be NP-hard. The computational time to find an optimal solution of an MAP grows exponentially with the number of dimensions and factorially with the dimension size [145]. Indeed, the total number cost coefficients, and the number of feasible solutions for a fully dense problem are given by the following expressions [39]:

$$\text{Total number cost coefficients} = \prod_{k=1}^{d} n_k.$$

$$\text{Number of feasible solutions } = \prod_{k=2}^{d} \frac{n_k!}{(n_k - n_1)!}.$$

In the literature, several exact and heuristic algorithms have been proposed for solving this problem [1, 12, 13, 39, 71, 107, 138, 144, 145, 147, 166, 168, 192]. Most of them are heuristic approaches, such as the GRASP (greedy randomized adaptive search procedure) with Path

Relinking for Three-Index Assignment [1], the Simulated Annealing [39], the Local Search Heuristics [71], Local Search Neighborhoods [145], etc. The exact solution methods are the Branch and Bound procedures [69, 144, 147], the Lagrangian Relaxation Based Algorithms [166, 168].

## 1.3 Value-at-Risk constrained optimization problem

The optimal stock selection is a classic financial problem since the seminal work of Markowitz [125]. It attempts to maximize portfolio expected return for a given amount of portfolio risk, or equivalently minimize risk for a given level of expected return. In the Markowitz approach, asset returns are random variables that can be controlled by two parameters: the portfolio efficiency is measured by the expectation, while risk is calculated by the standard deviation. There are several assumptions and consequences behind the Markovitz mean/variance model, such as returns are normally distributed, so that mean and variance are sufficient to fully describe the portfolio return distribution function. But in fact, it is frequently observed that returns in equity and other markets are not normally distributed. Another assumption that can assure the use the mean-variance approach is the quadratic shape of the decision maker utility function. But in this case one must accept that the utility function is decreasing with respect to wealth when a threshold is overcome. These observations lead to new research directions on portfolio models.

In the last years, some scholars developed new models for the optimal portfolio problem, taking into account the return non-normality. The simplest models are direct extensions of the Markowitz model. It is commonly accepted that efficiency is measured by the portfolio return expectation, but variance is now replaced by other risk measures.

Value-at-Risk, denoted $VaR$, is a new risk measure that most prominently imposed itself within the financial community in the last 10 years, especially due to regulatory reasons in the context of Basel-II for the banking sector, as well as Solvency-II for the insurance sector. In financial mathematics and financial risk management, Value-at-Risk is an estimate of the maximum potential loss with a certain confidence level, which a dealer or an end-user of financial instruments would experience during a standardized period (e.g. day, week, or year). In other words, with a certain probability, losses will not exceed $VaR$ [109].

Mathematically, let $X$ be the anticipated random returns, then the Value-at-Risk of $X$ is defined as

$$VaR_\alpha(X) = \inf\{u : F_X(u) \geq \alpha\} = F_X^{-1}(\alpha), \quad 0 < \alpha < 1,$$

where $F_X$ is the distribution function of $X$. $VaR_\alpha$ is said to be an acceptability functional [150].

If $X$ follows a discrete distribution taking the values $x_1, ..., x_S$ with equal probability, then

$$VaR_\alpha(X) = x_{\sigma(k)},$$

where $x_{\sigma(1)} \leq x_{\sigma(2)} \leq ... \leq x_{\sigma(s)}$ is the set of ordered values of $X$ ($\sigma$ is a permutation of the set {1, 2, ..., S}) and $k = \lfloor \alpha S \rfloor$. See [58] for the properties of this estimator. In the financial jargon, when no assumption is explicitly made about $F_X(.)$, then we say that $VaR_\alpha(X)$ has been estimated through "historical simulation" [40].

Let $\Theta$ be a linear space of measurable functions, defined on an appropriate probability space.

**Definition 1.1** *A functional $\rho : \Theta \to \mathbb{R} \cup \{+\infty\}$ is said to be coherent risk measure if it satisfies the following properties:*

1. *Normalized*
   $\rho(0) = 0$.

2. *Monotonicity*
   *If $X, Y \in \Theta$ and $X \leq Y$ then $\rho(X) \leq \rho(Y)$.*

3. *Sub-additivity*
   *If $X, Y \in \Theta$ then $\rho(X + Y) \geq \rho(X) + \rho(Y)$.*

4. *Positive homogeneity*
   *If $\lambda \geq 0$ and $X \in \Theta$, then $\rho(\lambda X) = \lambda \rho(X)$.*

5. *Translation invariance*
   *If $a \in \mathbb{R}$ and $X \in \Theta$, then $\rho(X + a) = \rho(X) + a$.*

**Definition 1.2** *A functional $\rho : \Theta \to \mathbb{R} \cup \{+\infty\}$ is said to be concave risk measure if $\lambda \in [0, 1], X, Y \in \Theta$ then*

$$\rho(\lambda X + (1 - \lambda)Y) \geq \lambda \rho(X) + (1 - \lambda)\rho(Y).$$

Although popular, $VaR$ -being the quantile of the return distribution- has some serious limitations, such as a lack of subadditivity and concavity [8, 9]. $VaR$ is coherent only when it is based on the standard deviation of normal distribution (for a normal distribution $VaR$ is proportional to the standard distribution) [9]. In the case of a finite number of scenarios, $VaR$ is a non-smooth, non-concave, and multi-extreme function with respect to positions [126].

Now, we introduce the two optimization models concerning Value-at-Risk, which are frequently considered in portfolio management.

Consider a finite set of arbitrary financial assets $i = 1, 2, ..., n$. Within a given observation period these assets generate returns

$$\xi = (\xi_1, \xi_2, ..., \xi_n)$$

measured as the relative increase (or decrease) of the asset prices during the period under consideration. These returns are unknown at the time of portfolio allocation and are treated as random variables. The investor has a budget of 1 unit (without loss of generality). He/she may decide on the positions

$$x = (x_1, x_2, ..., x_n)$$

in these assets, such that $x_i \geq 0$ (no short sales permitted) and $\sum_{i=1}^{m} x_i = 1$ (budget constraint). The return of the portfolio at the end of the observation period is

$$X = x^T \xi = \sum_{i=1}^{m} x_i \xi_i.$$

It is a random variable with distribution function $F$, i.e., $F(u) = P[X \leq u] = P[x^T \xi \leq u]$. Of course, $F$ depends on $x$. The expected return of portfolio $x$ is

$$\mathbb{E}(X) = \mathbb{E}(x^T \xi) = x^T \mathbb{E}(\xi).$$

The first model (Value-at-Risk constrained optimization) is

$$\begin{cases} \max \ \mathbb{E}(x^T \xi) \\ \text{s.t.} \ \sum_{i=1}^{n} x_i = 1, \\ \quad x_i \in [a_i, b_i], 1 \leq i \leq n, \\ \quad VaR_\alpha(x^T \xi) \geq a, \end{cases} \tag{1.3}$$

where $a$ is a lower bound of Value-at-Risk. This is model which we will study in this thesis.

For a given lower bound expectation $C$, the second one is

$$\begin{cases} \max \ VaR_\alpha(x^T \xi) \\ \text{s.t.} \ \sum_{i=1}^{n} x_i = 1, \\ \quad x_i \in [a_i, b_i], 1 \leq i \leq n, \\ \quad \mathbb{E}(x^T \xi) \geq C. \end{cases} \tag{1.4}$$

Because of the non-concavity of Value-at-Risk, both Problem (1.3) and Problem (1.4) are nonconvex programs and are known to be NP-hard [20]. In practice, in order to reduce the difficulty of these problems, $VaR$ is often replaced by the Average Value-at-Risk ($AVaR$, also called Conditional Value-at-Risk, $CVaR$), which is defined as

$$AVaR_\alpha(X) = \frac{1}{\alpha} \int_0^\alpha F_X^{-1}(t) dt = \frac{1}{\alpha} \int_0^\alpha VaR(t) dt, \quad 0 < \alpha \leq 1.$$

An important reason for this replacement is that $AVaR$ is a coherent risk measure [149]. It is thus relatively easy to incorporate into optimization problems above, especially in the

case of discrete random variables, linear programming formulations exist [6, 174, 199]. Some other reasons justifying the use of $AVaR$ instead of $VaR$ can be found in [208].

However, due to regulatory frameworks such as Basel II and Solvency II, Value-at-Risk remains to be an industry standard and is widely used in portfolio planning. Therefore, numerous approaches to solve either Problem (1.3) or Problem (1.4) have been proposed in the literature [20,34,37,62,63,79,109,140,208]. For instance, in [20], Benati and Rizzi reformulate them as mixed-integer programs which allow us to use solvers, for instance CPLEX, to obtain their solution. Some heuristic solution schemes have been designed, such as random search with threshold acceptance [62, 63], or evolutionary computation techniques [79]. In [34] complete enumeration on the risk-return grid is used to find near optimal portfolios for the $VaR$ portfolio optimization problem. Pang and Leyffer [140] formulate the problem (1.4) as a linear program with equilibrium constraints (LPEC) to derive lower and upper bounds for a Branch-and-Bound solution. Cheon et al. [37] propose a solution technique for the more general class of probabilistically constrained linear programs which is based on a Branch-Reduced-Cut algorithm. Larsen et al. [109] have developed two algorithms performing well even on large datasets for the optimization of $VaR$. Recently, Wozabal et al. [208] gave a representation of the $VaR$ as a DC function (Difference of Convex functions) in the case finite scenarios, and proposed a conical Branch-and-Bound algorithm to find global optima of (1.3). Later, Wozabal [209] introduced a DC Algorithm (DCA) [120, 156, 157] to the DC formulation of the problem (1.3).

## 1.4 Conclusion

In this chapter we have introduced some optimization problems in Search Theory, Assignment and Portfolio Management, which we study in this thesis. They are generally nonconvex, non-differentiable problems. Therefore, among the methods for solving optimization problems, we choose two powerful tools to develop solution methods. They are the DC programming&DCA and the Cross-Entropy (CE) method. The methodologies of these methods will be presented in the two next chapters.

# Chapter 2

# DC programming and DC Algorithm (DCA)

Let $X$ be an Euclidean space $\mathbb{R}^n$, equipped with an inner product $\langle x, y\rangle = \sum_{i=1}^n x_i y_i = x^T y$, and the corresponding Euclidean norm $\|x\|_2 = \sqrt{\langle x, x\rangle}$. The dual space of $X$, denoted $Y$, is thus identified with $X$ itself. One notes $\bar{\mathbb{R}} = \mathbb{R} \cup \{\pm\infty\}$, and uses the convention $(+\infty) - (+\infty) = +\infty$.

## 2.1 Convex analysis

In this section, we review some definitions and theorems in convex analysis, which are the basis of DC programming. For more details, we could refer some books of Rockafellar [173] and Hiriart-Urruty [74].

**Definition 2.1** *(Convex Set) A set $C \subset X$ is convex if the line segment joining any two points $x$ and $y$ in $C$ is contained in $C$, i.e.,*

$$\forall \lambda \in [0, 1], \lambda x + (1 - \lambda) y \in C.$$

**Definition 2.2** *(Convex Hull) Given a set $S \subset X$, the convex hull of $S$, denoted $co(S)$, is the smallest convex set containing $S$.*

It is not difficult to show that $co(S)$ is the set of all convex combination of elements in $S$, i.e.,

$$co(S) = \left\{ \sum_{i=1}^m \lambda_i x^i | x^i \in S, \lambda_i \geq 0, i = 1, ..., m, \sum_{i=1}^m \lambda_i = 1 \right\}.$$

**Definition 2.3** *(Affine hull) Given a convex set $C$, the affine hull of $C$, denoted $aff(C)$, is defined by:*

$$aff(C) = \left\{ \sum_{i=1}^m \lambda_i x^i | x^i \in C, \lambda_i \in \mathbb{R}, i = 1, ..., m, \sum_{i=1}^m \lambda_i = 1 \right\}.$$

The affine hull is the smallest affine set that contains $C$ in the following sense: if $S$ is any affine set such that $C \subset S$, then aff$(C) \subset S$.

**Definition 2.4** *(The relative interior) The relative interior of a convex set $C$ in $X$, denoted $ri(C)$, is its interior within its affine hull, i.e.,*

$$ri(C) = \{x \in C : \exists r > 0, B(x, r) \cap aff(C) \subset C\},$$

*where $B(x, r)$ is the ball with center $x$ and radius $r$.*

**Remark 2.5** *Any nonempty convex set in $\mathbb{R}^n$ has nonempty relative interior.*

Given a function $f : C \to (-\infty, +\infty]$ on a convex set $C$. The *domain* of the function $f$ is the set:

$$dom(f) = \{x \in C, f(x) < +\infty\}.$$

We say $f$ is proper if its domain is nonempty.

**Definition 2.6** *(Convex function) $f$ is convex if for all $x$, $y$ in $C$ and for all $\lambda \in [0, 1]$, we have :*

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y). \tag{2.1}$$

*$f$ is strictly convex if strict inequality holds in (2.1) whenever $x \neq y$, and $0 < \lambda < 1$.*

**Definition 2.7** *(Epigraph) The epigraph of the function $f$, denoted $epi(f)$, is defined by*

$$epi(f) = \{(x, \alpha) \in C \times \mathbb{R} : f(x) \leq \alpha\}.$$

The link between convex sets and convex functions is via epigraph: a function is convex if and only if its epigraph is a convex set in $X \times \mathbb{R}$.

**Definition 2.8** *(Strongly convex function) The function $f$ is called strongly convex with modulus $\rho$ on the convex set $C$ (also called $\rho$-convex) if there exists $\rho > 0$ such that for all $x$, $y$ in $C$, and for all $\lambda \in [0, 1]$, we have :*

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y) - \lambda(1 - \lambda)\frac{\rho}{2}\|x - y\|^2.$$

It means that the function $f - \frac{\rho}{2}\|.\|$ is convex on $C$.
The modulus of strongly convex of $f$ on $C$, denoted $\rho(f, C)$, is defined by

$$\rho(f, C) = \sup\{\rho > 0 : f - \frac{\rho}{2}\|.\| \text{ is convex on } C\}.$$

Clearly, $f$ is convex on $C$ if and only if $\rho(f, C) \geq 0$. One says that $f$ is strongly convex on $C$ if $\rho(f, C) > 0$.

**Remark 2.9** *Every strongly convex function is strictly convex and every strictly convex function is convex.*

**Definition 2.10** *(Subgradient) Let $f$ be a proper convex function on $X$, a vector $y$ in $Y$ is called subgradient of $f$ at $x^0 \in dom(f)$ if for all $x \in X$, we have:*

$$\langle y, x - x^0 \rangle \leq f(x) - f(x^0).$$

*We denote the set of subgradients (called the subdifferential) by $\partial f(x^0)$, defining $\partial f(x) = \emptyset$ for $x$ not in $dom(f)$. The domain of subdifferential of $f$ is defined as*

$$dom(\partial f) = \{x : \partial f(x) \neq \emptyset\}.$$

**Proposition 2.11** *Given $x \in dom(\partial f)$, then $\partial f(x)$ is a nonempty closed convex set in $Y$.*

**Definition 2.12** *($\epsilon$-subgradient) Let $\epsilon$ be a positive, a vector $y \in Y$ is called $\epsilon$-subgradient of $f$ at $x^0$ if*

$$\langle y, x - x^0 \rangle \leq f(x) - f(x^0) + \epsilon, \forall x \in X.$$

*We denote $\partial_\epsilon f(x^0)$ the set of all $\epsilon$-subgradients of $f$ at $x^0$.*

**Definition 2.13** *We say that $f$ is lower semi-continuous (l.s.c.) at $x^0 \in C$ if*

$$\lim_{y \to x^0} \inf f(y) = f(x^0).$$

The set of all lower semi-continuous proper convex functions on $X$ is denoted by $\Gamma_0(X)$.

**Definition 2.14** *(Conjugate function) The Fenchel conjugate of a function $f : X \to [-\infty, +\infty]$ is the function $f^* : Y \to [-\infty, +\infty]$, defined by*

$$f^*(y) = \sup\{\langle x, y \rangle - f(x) : x \in X\}.$$

**Proposition 2.15** *Given $f : X \to \bar{\mathbb{R}}$, thus:*

- *$f^*$ is convex.*
- *$f \in \Gamma_0(X) \Leftrightarrow f^* \in \Gamma_0(Y)$ and $(f^*)^* = f$.*
- *$f \in \Gamma_0(X)$, thus $y \in \partial f(x) \Leftrightarrow x \in \partial f^*(y)$.*
- *$y \in \partial f(x) \Leftrightarrow f(x) + f^*(y) = \langle x, y \rangle$.*
- *$x^0 \in argmin\{f(x) : x \in X\} \Leftrightarrow 0 \in \partial f(x^0)$.*

**Definition 2.16** *(Polyhedral convex set) A convex set $C$ is called polyhedral convex if it is a finite intersection of closed half-spaces in $\mathbb{R}^n$,i.e.,*

$$C = \bigcap_{i=1}^{m} \{x \in \mathbb{R}^n : \langle a_i, x \rangle - b_i \leq 0, a_i \in \mathbb{R}^n, b_i \in \mathbb{R}\}.$$

**Definition 2.17** *(Polyhedral function) A function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is called polyhedral if its epigraph is polyhedral in $\mathbb{R}^{n+1}$.*

Note that a polyhedral function is convex, proper and l.s.c.

**Proposition 2.18** *Let $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ be a convex function. Thus, $f$ is polyhedral if and only if $dom(f)$ is polyhedral convex set and*

$$f(x) = \max\{\langle a_i, x\rangle - b_i : i = 1, ..., l\} + \chi_{dom(f)}(x).$$

**Proposition 2.19** *The polyhedral convex functions have the following properties:*

- *If $f_1$ and $f_2$ are polyhedral convex, then $f_1 + f_2$ , $\max\{f_1, f_2\}$ are polyhedral convex.*
- *If $f$ is polyhedral convex then $f^*$ is also polyhedral convex and $dom(\partial f) = dom(f)$. Moreover, if $f$ is finite everywhere then*

$$dom(f^*) = co\{a_i : i = 1, ..., l\},$$

$$f^*(y) = \min\{\sum_{i=1}^{l} \lambda_i b_i : y = \sum_{i=1}^{l} \lambda_i a_i, \sum_{i=1}^{l} \lambda_i = 1, \lambda_i \geq 0, i = 1, ..., l\}.$$

- *If $f$ is polyhedral convex then $\partial f(x)$ is non-empty polyhedral convex set for all $x \in dom(f)$.*
- *Let $f_1, ..., f_m$ be the polyhedral convex functions on $X$ such that the convex sets $dom(f_i), i = 1, ..., m$ have a common point. Then,*

$$\partial(f_1 + ... + f_m)(x) = \partial f_1(x) + ... + \partial f_m(x), \forall x \in X$$

## 2.2 The DC function class

In this section, we introduce the definition of DC functions and their properties. For more details, see [3, 73, 85, 110, 158].

**Definition 2.20** *(DC function) Let $C$ be a convex set in $X$. A function $f : C \to \mathbb{R} \cup \{+\infty\}$ is called DC on $C$ if it can be expressed as the difference of two convex functions on $C$, i.e.,*

$$f(x) = g(x) - h(x), \forall x \in C,$$

*where $g$ and $h$ are convex functions on $C$. One says that $g - h$ is a DC decomposition of $f$.*

**Remark 2.21** *If $f = g - h$ is a DC function on $C$ then for all finite convex function $p$ on $C$, $f = (g + p) - (h + p)$ is also a DC decomposition of $f$. Thus, a DC function has infinitely many DC decompositions.*

Let $\mathrm{Conv}(C)$ be the set of convex functions on $C$. The set of DC functions on $C$, denoted $\mathrm{DC}(C)$, is the vector space spanned by $\mathrm{Conv}(C)$ defined by

$$\mathrm{DC}(C) = \mathrm{Conv}(C) - \mathrm{Conv}(C).$$

The DC functions were considered in years 50 by Aleksandrov [3], and Hartman [73], etc., who proved several important properties of these functions. It is worth noting the richness of DC(X): it contains almost realistic objective functions and is closed under all the operations usually considered in optimization.

**Proposition 2.22** *If $f \in DC(C)$ and $f = g - h$ then*

- $|f| \in DC(C)$, *because*
$$|f| = 2\max(g, h) - (g + h).$$
- $f^+, f^- \in DC(C)$, *where* $f^+(x) = \max(0, f(x)), f^-(x) = \min(0, f(x))$ *because*
$$f^+ = \max(g, h) - h, f^- = g - \max(g, h).$$

**Proposition 2.23** *If $f_i = g_i - h_i, i = 1, ..., m$ are functions in $DC(C)$ then*

- *A linear combination of DC functions is a DC function, i.e.,*
$$\sum_{i=1}^{m} \lambda_i f_i \in DC(C), \forall \lambda_i \in \mathbb{R}.$$
- $\max\limits_{1 \le i \le m} f_i \in DC(C)$ *and* $\min\limits_{1 \le i \le m} f_i \in DC(C)$ *because*
$$\max_{1 \le i \le m} f_i \in DC(C) = \max_{1 \le i \le m} (g_i + \sum_{j \ne i} h_j) - \max_{1 \le i \le m} h_i \in DC(C).$$
- $\prod_{i=1}^{m} f_i \in DC(C)$.

**Definition 2.24** *(Locally DC function) A function $f : C \to \overline{\mathbb{R}}$ defined on an open convex set $C \subset X$ is said to be locally DC if for every $x^0 \in C$, there exist a convex neighborhood of $x^0$ where it is DC.*

Let $C$ be an open convex set in $X$, we denote by $C^{1,1}(C) \subset C^1(C)$ the set of functions, whose gradients are locally Lipschitzian on $C$. $C^{1,1}(C)$ is contained in the convex cone $LC^2(C)$ (also called functions of sub-$C^2$) - the set of functions being locally lower hull of a family of functions in class $C^2(C)$. The following theorem shows the richness of the class of DC functions:

**Theorem 2.25**
- *A convex function of Conv(C) is a function of sub-$C^2$ on $C$.*

- *A function sub$-C^2$ on $C$ is a locally DC function on $C$.*
- *A locally DC function on $C$ is a DC function on $C$. Finally, we have:*

$$C^2(C) \cup C^{1,1}(C) \cup C^1(C) \cup Conv(C) \cup LC^2(C) \subset DC(C).$$

*Moreover, if $C$ is compact, then $DC(C)$ is a dense vector subspace in the set of continuous functions on $C$ equipped with the norm of uniform convergence on $C$.*

## 2.3 DC Programming

In recent years, there has been very active researches in the following classes of nonconvex and non-differentiable optimization problems:

(1) $\sup\{f(x) : x \in C\}$, $f$ and $C$ are convex.
(2) $\inf\{g(x) - h(x) : x \in \mathbb{R}^n\}$, $g, h$ are convex.
(3) $\inf\{g(x) - h(x) : x \in C, f_1(x) - f_2(x) \leq 0\}$, $g, h, f_1, f_2$ and $C$ are convex.

Problem (1) is a special case of Problem (2) with $g = \chi_C$, the indicator function of $C$ and $h = f$, while the latter (resp. Problem (3)) can be equivalently transformed into the form of (1) (resp. (2)) by introducing an additional scalar variable (resp. via exact penalty relative to the d.c. constraint $f_1(x) - f_2(x) \leq 0$, see [115, 117]). Clearly the complexity increases from (1) to (3), the solution of one of them implies the solution of the two others. Problem (2) is called a DC program whose particular structure has been permitting a good deal of development both in qualitative and quantitative studies.

## 2.4 DC duality

The concept of duality is a fundamental concept in mathematics. In convex analysis, duality theory has been given for several decades [173]. More recently, the duality has been proposed and developed in nonconvex analysis. Firstly, in Quasi-convex programming and Anti-convex programming [11, 148]. Then the DC duality introduced by Toland [198] is a generalization of the work of Pham Dinh Tao on convex maximization [151]. For more details on DC duality, we refer to the work of Le Thi Hoai An [110]. In this section we will present the main results about DC duality.

Consider the following DC program:

$$(P_{dc}) \qquad \alpha = \inf\{f(x) = g(x) - h(x) : x \in X\},$$

where $g, h \in \Gamma_0(X)$. We use a convention $(+\infty) - (+\infty) = (+\infty)$ as in convex optimization. Since $h \in \Gamma_0(X)$, we have $h^{**} = h$ and

$$h(x) = (h^*)^*(x) = \sup\{\langle x, y\rangle - h^*(y) : y \in Y\}.$$

With this relationship, we find that

$$\begin{aligned}\alpha &= \inf\{g(x) - \sup\{\langle x, y\rangle - h^*(y) : y \in Y\} : x \in X\} \\ &= \inf\{\inf\{g(x) - (\langle x, y\rangle - h^*(y)) : x \in X\} : y \in Y\} \\ &= \inf\{\beta(y) : y \in Y\},\end{aligned}$$

where

$$\begin{aligned}\beta(y) &= \inf\{g(x) - [\langle x, y\rangle - h^*(y)] : x \in X\} \\ &= \begin{cases} h^*(y) - g^*(y) & \text{if } y \in dom(h^*), \\ +\infty & \text{otherwise.}\end{cases}\end{aligned}$$

With the convention $(+\infty) - (+\infty) = (+\infty)$, we obtain finally the dual problem of $(P_{dc})$, denoted $(D_{dc})$ :

$$(D_{dc}) \qquad \alpha = \inf\{h^*(y) - g^*(y) : y \in Y\}.$$

The $(D_{dc})$ is also a DC program because $h^*$ et $g^*$ are two convex functions in $\Gamma_0(Y)$.

Note that there are cases, where the dual problem $(D_{dc})$ is a convex problem, although the primal problem $(P_{dc})$ is not convex (see [137]). In the following theorem, we observe the perfect symmetry between the primal and dual problems.

**Theorem 2.26** *Given $g, h \in \Gamma_0(X)$, then*

1. *$\inf\{g(x) - h(x) : x \in X\} = \inf\{h^*(y) - g^*(y) : y \in Y\}$.*
2. *If $y^0$ is a minimum of $h^* - g^*$ on $Y$ then each $x^0 \in \partial g^*(y^0)$ is a minimum of $g - h$ on $X$.*
3. *If $x^0$ is a minimum of $g - h$ on $X$ then each $y^0 \in \partial h(x^0)$ is a minimum of $h^* - g^*$ on $Y$.*

## 2.5 Optimality conditions in DC optimization

Let $\mathcal{P}$ and $\mathcal{D}$ be the solution sets of problems $(P_{dc})$ and $(D_{dc})$, a point $x^*$ is called critical point of problem $(D_{dc})$ if $\partial g(x^*) \cap \partial h(x^*) \neq \emptyset$ (A general KKT point).

In convex optimization, $x^*$ is a minimum of convex function $f$ if and only if $0 \in \partial f(x^*)$. In DC optimization, the optimal condition is formulated by using $\epsilon$-sub-differential of $g$ and $h$.

**Theorem 2.27** *(Global optimality condition) If $g, h \in \Gamma_0(X)$ and $f = g - h$, then $x^*$ is a global minimum of $g - h$ on $X$ if and only if*

$$\partial_\epsilon h(x^*) \subset \partial_\epsilon g(x^*), \forall \epsilon > 0.$$

**Remark 2.28**

1. *If $f \in \Gamma_0(X)$, we can write $g = f$ et $h = 0$. In this case the global optimality condition in DC programming is identical to that in convex programming $0 \in \partial f(x^*)$, since $\partial_\epsilon h(x^*) = \partial h(x^*) = 0, \forall \epsilon > 0$.*

2. *More generally, we consider the DC decompositions of $f \in \Gamma_0(X)$ in the form $f = g - h$ with $g = f + h$ and $h \in \Gamma_0(X)$ everywhere on $X$. The corresponding DC problem is a "false" DC problem because it is a convex optimization problem. In this case, the optimality condition $0 \in \partial f(x^*)$ is equivalent to $\partial h(x^*) \subset \partial g(x^*)$.*

**Definition 2.29** *Let $g$ and $h$ be two functions in $\Gamma_0(X)$. A point $x \in dom(g) \cap dom(h)$ is a local minimum of $g - h$ on $X$ if and only if*

$$g(x^*) - h(x^*) \leq g(x) - h(x), \forall x \in V_{x^*},$$

*where $V_{x^*}$ denotes a neighborhood of $x^*$.*

**Theorem 2.30** *(Necessary condition for local optimality) If $x^*$ is a local minimum of $g - h$ then $\partial h(x^*) \subset \partial g(x^*)$.*

**Remark 2.31** *If $h$ is polyhedral convex, the necessary condition for local optimality is also sufficient.*

**Corollary 2.32** *If $h \in \Gamma_0(X)$ is polyhedral convex then a necessary and sufficient such that $x^* \in X$ is a local minimum of $g - h$ is*

$$\partial h(x^*) \subset int(\partial g(x^*)).$$

**Theorem 2.33** *(Sufficient condition for local optimality) If $x^*$ admits a neighborhood $V$ such that*

$$\partial h(x) \cap \partial g(x^*) \neq \emptyset, \forall x \in V \cap dom(g),$$

*then $x^*$ is a local minimum of $g - h$.*

**Corollary 2.34** *If $x^* \in int(dom(h))$ satisfies*

$$\partial h(x^*) \subset int(\partial g(x^*)),$$

*then $x^*$ is a local minimum of $g - h$.*

**Theorem 2.35**

1. *$\partial h(x) \subset \partial g(x), \forall x \in \mathcal{P}$, and $\partial g^*(y) \subset \partial h^*(y), \forall y \in \mathcal{D}$,*

2. *Transport of global minima :*

$$\bigcup_{x\in\mathcal{P}} \partial h(x) \subseteq \mathcal{D} \subset dom(h^*).$$

*The first inclusion becomes equality if $g^*$ is sub-differentiable in $\mathcal{D}$ (especially if $D \subset ri(dom(g^*))$ or if $g^*$ is sub-differentiable in $dom(h^*)$). In this case $\mathcal{D} \subset (dom(\partial g^*) \cap dom(\partial h^*))$.*

$$\bigcup_{y\in\mathcal{D}} \partial g^*(y) \subseteq \mathcal{P} \subset dom(g).$$

*The first inclusion becomes equality if $h$ is sub-differentiable in $\mathcal{P}$ (especially if $P \subset ri(dom(h))$ or if $h$ is sub-differentiable in $dom(g)$). In this case $\mathcal{P} \subset (dom(\partial g) \cap dom(\partial h))$.*

3. *Transport of local minima : Let $x^* \in dom(\partial h)$ be a local minimum of $g-h$, $y^* \in \partial h(x^*)$, and $V_{x^*}$ is a neighborhood of $x^*$ such that $g(x)-h(x) \geq g(x^*)-h(x^*), \forall x \in V_{x^*} \cap dom(g)$. If*

$$x^* \in int(dom(g^*)), \partial g^*(y^*) \subset V_{x^*},$$

*then $y^*$ is a local minimum of $h^* - g^*$.*

## 2.6 DCA (DC Algorithm)

The DCA was introduced by Pham Dinh Tao in 1985 as an extension of his subgradient algorithms (for convex maximization programming) to DC programming, and the extensively developed by Le Thi Hoai An and Pham Dinh Tao since 1994 to become now classic, and widely used by researchers and practitioners in various fields of applied sciences (see [110–113, 152–157]).

The idea of the DCA (also called the simplified DCA) is to construct two sequences $\{x^k\}$ and $\{y^k\}$ (candidates for being optimal solutions of primal and dual programs, respectively), which are easy to calculate and satisfy the following conditions:

1. The sequences $\{g(x^k) - h(x^k)\}$ and $\{h^*(y^k) - g^*(y^k)\}$ are decreasing.
2. Every limit point $x^*$ of the sequence $\{x^k\}$ (resp. $y^*$ of the sequence $\{y^k\}$) is a critical point of $g - h$ (resp. $h^* - g^*$).

Starting $x^0 \in dom(g)$, these two sequences $\{x^k\}$ and $\{y^k\}$ are determined in the following way

$$x^k \to y^k \in \partial h(x^k) = \text{argmin}\{h^*(y) - g^*(y^{k-1}) - \langle y - y^{k-1}, x^k\rangle : y \in Y\} \ (D_k)$$

$$y^k \to x^{k+1} \in \partial g^*(y^k) = \text{argmin}\{g(x) - h(x^k) - \langle x - x^k, y^k\rangle : x \in X\} \ (P_k)$$

Problem $(P_k)$ is a convex program obtained from $(P_{dc})$ by replacing $h$ with its affine minorization defined by $y^k \in \partial h(x^k)$. Similarly, the convex problem $(D_k)$ is obtained from $(D_{dc})$ by using the affine minorization of $g^*$ defined by $x^k \in \partial g^*(y^{k-1})$. Here we can see the complete symmetry between problems $(P_k)$ and $(D_k)$, and between the sequences $\{x^k\}$ and $\{y^k\}$ relatively to the duality of DC optimization.

Implement the algorithm that consists of two steps:

1. Choose $x^0 \in X$.
2. Repeat
   - Compute $y^k \in \partial h(x^k)$.
   - Set $x^{k+1} \in \partial g^*(y^k)$ that leads to solving the convex program

$$\inf\{g(x) - h(x^k) + \langle x - x^k, y^k \rangle : x \in X\}.$$

   Until a stopping criterion is met.

**Lemma 2.36** *(Existence of the sequences) The following statements are equivalent*

1. *The sequences $\{x^k\}$ and $\{y^k\}$ are well defined*
2. *$dom(\partial g) \subset dom(\partial h)$ and $dom(\partial h^*) \subset dom(\partial g^*)$*

The following proposition establishes the conditions of boundedness for the sequences generated by DCA.

**Lemma 2.37** *(Boundedness of the sequences) If $g - h$ is coercive, then*

1. *The sequence $\{x^k\}$ is bounded.*
2. *If $\{x^k\} \in int(dom(h))$ then the sequence $\{y^k\}$ is also bounded.*

*If $h^* - g^*$ is coercive, then*

1. *The sequence $\{y^k\}$ is bounded.*
2. *If $\{y^k\} \in int(dom(g^*))$ then the sequence $\{x^k\}$ is also bounded.*

The convergence of the DCA is given by the following theorem:

**Theorem 2.38** *(Convergence of DCA) Assume that the sequences $\{x^k\}$ et $\{y^k\}$ are well defined*

1. *The sequences $\{g(x^k) - h(x^k)\}$ and $\{h^*(y^k) - g^*(y^k)\}$ are decreasing and*

- *$g(x^{k+1}) - h(x^{k+1}) = g(x^k) - h(x^k)$ if and only if $y^k \in \partial g(x^k) \cap \partial h(x^k), y^k \in \partial g(x^{k+1}) \cap \partial h(x^{k+1})$ and $[\rho(g) + \rho(h)]\|x^{k+1} - x^k\| = 0$. Moreover, if $g$ and $h$ are strictly convex on $X$, then $x^k = x^{k+1}$. In this case, DCA will terminate in a finite iterations. $x^k$ and $x^{k+1}$ are critical points of the function $g - h$.*
  *($\rho(g) = \rho(g, X)$ is the modulus of strong convexity of $g$ on $X$.)*
- *$h^*(y^{k+1}) - g^*(y^{k+1}) = h^*(y^k) - g^*(y^k)$ if and only if $x^{k+1} \in \partial g^*(y^k) \cap \partial h^*(y^k), x^k \in \partial g^*(x^{k+1}) \cap \partial h^*(x^{k+1})$ and $[\rho(g^*, D) + \rho(h^*, D)]\|y^{k+1} - y^k\| = 0$. Moreover, if $g^*$ and $h^*$ are strictly convex on $Y$, then $y^k = y^{k+1}$. In this case, DCA will terminate in a finite iterations. $y^k$ et $y^{k+1}$ are critical points of the function $h^* - g^*$.*

2. *If $\rho(g, C) + \rho(h, C) > 0$ (resp. $\rho(g^*, D) + \rho(h^*, D) > 0$), then the sequence $\{\|x^{k+1} - x^k\|\}$ (resp. $\{\|y^{k+1} - y^k\|\}$) converges.*

3. *If the optimal value of the problem $(P_{dc})$ is finite and if the sequences $\{x^k\}$ and $\{y^k\}$ are bounded, then every limit point $x^*$ (resp. $y^*$) of $\{x^k\}$ (resp. $\{y^k\}$) is a critical point of $g - h$ (resp. $h^* - g^*$).*

4. *If DCA converges to a point $x^*$ that admits a convex neighbourhood in which the objective function $f$ is finite and convex (i.e., the function $f$ is locally convex at $x^*$) and if the second DC component $h$ is differentiable at $x^*$, then $x^*$ is a local minimizer for $(P_{dc})$.*

In general, to apply DCA, we have to answer the following two questions:

1. How to find a "good" DC decomposition?

2. How to find a "good" starting point?

From the theoretical point of view, the question of optimal DC decompositions is still open. Of course, this depends strongly on the very specific structure of the problem being considered. In order to tackle the large scale setting, one tries in practice to choose $g$ and $h$ such that sequences $\{x^k\}$ and $\{y^k\}$ can be easily calculated, i.e., either they are in explicit form or their computations are inexpensive. Normally, we try to ensure that:

- The function $h$ allows us to compute easily $\partial h$.

- The convex program $\inf\{g(x) - \langle x, y^k\rangle : x \in X\}$ is easy to solve, this is particularly important for a large problem.

- The initial point $x^0$ is as close as possible to a global solution.

## 2.7 Polyhedral DC optimization

**Definition 2.39** *(Polyhedral DC optimization) A DC program is called polyhedral if one convex component (g and h) of the function $f = g - h$ is a polyhedral convex function.*

It can be shown that for solving a polyhedral DC program, DCA has a finite convergence. Denoting by $h_k$ the affine minorization of the convex function $h$ at $x^k$, we will construct $h^k$, polyhedral convex function generated by $\{h_k\}$.

$$h_k = h(x^k) + \langle x - x^k, y^k \rangle = \langle x, y^k \rangle - h^*(y^k), x \in X$$

$$h^k(x) = \sup\{h_i(x) : i = 0, ..., k\} = \sup\{\langle x, y^k \rangle - h^*(y^k) : i = 0, ..., k\}, x \in X,$$

where the sequences $\{x^k\}$ et $\{y^k\}$ are obtained through the DCA procedure.
We define similarly the dual function $g_k^*$ (resp. $(g^*)^k$)

$$g_k^* = g^*(x^{k-1}) + \langle y - y^{k-1}, x^k \rangle = \langle y, x^k \rangle - g(x^k), y \in Y$$

$$(g^*)^k(x) = \sup\{g_i^*(x) : i = 0, ..., k\} = \sup\{\langle y, x^k \rangle - g(x^k) : i = 0, ..., k\}, x \in X$$

Note that a proper, l.s.c., convex function is characterized as the supremum of its affine minorizations. Therefore, it is judicious to use $h^k$ (resp. $(g^*)^k$) as a polyhedral convex minorization of the convex function $h$ (resp. $g^*$) on $X$ (resp. $Y$), instead of the affine minorization $h_k$ (resp. $(g^*)_k$). We then obtain the following program:

$$(P^k) \quad \inf\{g(x) - h^k(x) : x \in X\}$$

$$(D^k) \quad \inf\{h^*(x) - (g^*)^k(x) : x \in X\}$$

The problems $(P^k)$ and $(D^k)$ are nonconvex DC problems. But the sub-problems $(P_k)$ and $(D_k)$ are convex. We have the following theorem:

**Theorem 2.40**

1. $g(x^{k+1}) - h(x^{k+1}) = h^*(y^k) - g^*(y^k)$ *if and only if* $h^k(x^{k+1}) = h(x^{k+1})$.
2. $h^*(y^k) - g^*(y^k) = g(x^k) - h(x^k)$ *if and only if* $g^*(y^k) = (g^*)^k(y^k)$.
3. $g(x^{k+1}) - h(x^{k+1}) = g(x^k) - h(x^k)$ *if and only if* $h^k(x^{k+1}) = h(x^{k+1})$ *and* $g^*(y^k) = (g^*)^k(y^k)$.

*Therefore, if* $g(x^{k+1}) - h(x^{k+1}) = g(x^k) - h(x^k) = h^*(y^k) - g^*(y^k)$ *then the following propositions are true:*

- $x^{k+1}$ *(resp.* $y^k$*) is an optimal solution of the problem* $(P^k)$ *(resp.* $(D^k)$*).*
- *If* $h$ *and* $h^k$ *coincide with a solution of* $(P_{dc})$ *or/and* $g^*$ *and* $(g^*)^k$ *coincide at some solution to* $(D_{dc})$*, then* $x^{k+1}$ *(resp.* $y^k$*) is also a solution to* $(P_{dc})$ *(resp.* $(D_{dc})$*), and* $h^k(x^{k+1}) = h(x^{k+1})$.

Suppose that the optimal value of problem $(P_{dc})$ is finite and the sequence $\{x^k\}$ generated by DCA is bounded, then for each limit point $x^\infty$ we have

$$g(x^\infty) - h_\infty(x^\infty) = \inf\{g(x^{i+1}) - h_i(x^{i+1}) i = 0, 1, ..., \infty\},$$

where $h_\infty$ is the affine minorization of $h$ at $x^\infty$ defined by

$$h_\infty = h(x^\infty) + \langle x - x^\infty, y^\infty \rangle = \langle x, y^\infty \rangle - h^*(y^\infty), x \in X,$$

with $y^\infty \in \partial h(x^\infty)$ a limit point of $\{y^k\}$. The point $x^\infty$ is therefore a solution of the DC program

$$(P^\infty) \quad \inf\{g(x) - h^\infty(x) : x \in X\}.$$

Similarly, the point $y^\infty$ is a solution of the DC program

$$(D^\infty) \quad \inf\{h^*(x) - (g^*)^\infty(x) : y \in Y\}.$$

We have the following theorem:

**Theorem 2.41** *If the optimal value of the problem $(P_{dc})$ (resp. $(D_{dc})$) is finite and the sequence $\{x^k\}$(resp. $\{y^k\}$) is bounded, then for every limit point $x^\infty$ (resp. $y^\infty$) of $\{x^k\}$(resp. $\{y^k\}$) is a solution to the problem $(P^\infty)$ (resp. $(D^\infty)$). Moreover, the optimal values are equal, i.e.,*

$$g(x^\infty) - h^\infty(x^\infty) = h^*(y^\infty) - (g^*)^\infty(y^\infty).$$

*If one of the following conditions is true:*

- *The functions $h$ and $h^\infty$ coincide at some optimal solution to $(P_{dc})$*
- *The functions $g^*$ and $(g^*)^\infty$ coincide at some optimal solution to $(D_{dc})$ then $x^\infty$ and $y^\infty$ are also optimal solutions to $(P_{dc})$ and $(D_{dc})$, respectively.*

## 2.8 Exact penalty

The exact penalty techniques are often very useful to transform a DC problem with anti-convex constraints into a DC problem with convex constraints. The latter problem can be solved by DCA. For more details on the exact penalty techniques for DC programming we can refer to [115, 122].

Consider an optimization problem

$$(P) \quad \alpha = \inf\{f(x) : x \in C, p(x) \leq 0\},$$

where $C$ is a non-empty, bounded and convex polyhedron in $\mathbb{R}^n$, $f, g$ are finite concave functions on $C$, and $p$ is a function of positive values or zero on $C$. The problem $(P)$ is a nonconvex program. The nonconvexity of the problem is due to the fact that the objective

function $f$ is concave and the constraint $p(x) \leq 0$ is non convex. A constraint $p(x) \leq 0$ with $p(x)$ concave is called anti-convex. The objective of the problem $(P)$ is to minimize a concave function $f$ with polyhedral convex constraint and anti-convex constraint. Suppose that the problem $(P)$, whose solution set is denoted by $\mathcal{P}$ is feasible. Given a parameter $t > 0$, we can define the penalty problem by:

$$(P_t) \quad \alpha(t) = \inf\{f(x) + tp(x) : x \in C\}.$$

The solution set is denoted by $\mathcal{P}_t$. We obtain the following theorem:

**Theorem 2.42** *(Exact penalty theorem) Let $C$ be a non-empty, bounded and convex polyhedron; $f$ and $p$ are two finite concave functions on $C$, and $p$ is a function of positive values or zero on $C$. Then, there exists a finite number $t_0 \geq 0$ such that for all $t \geq t_0$, the problem $(P)$ and $(P_t)$ are equivalent in the sense that $\mathcal{P}_t = \mathcal{P}$ and $\alpha(t) = \alpha$. The parameter $t_0$ can be determined as follows:*

- *If the set of vertices $V(C)$ of $C$ is contained in the set $\{x \in C : p(x) \leq 0\}$, then $t_0 = 0$ and $\alpha(0) = \alpha$.*
- *If $\alpha(0) < \alpha$ then $t_0 = \max\{\dfrac{\alpha - f(x)}{p(x)} : x \in V(C), p(x) > 0\}$*

*We also have the following properties:*

- *$\alpha(t) = \alpha$ if and only if $t \geq t_0$*
- *$\mathcal{P}_t \cap \{x \in C : p(x) \leq 0\} \neq \emptyset \Leftrightarrow \mathcal{P}_t \subset \mathcal{P} \Leftrightarrow t \geq t_0$*
- *$\mathcal{P}_t = \mathcal{P} \Leftrightarrow t > t_0$*

The class of Problem $(P)$ satisfying the assumption of Theorem 2.42 contains many important real-life ones (see [115]): Convex maximization over the Perato set, bilevel linear programs, linear programs with mixed linear complementarity constraints, mixed zero-one concave minimization programming, etc.

Recently, Le Thi et al. (see [122]) have proved the following result:
We consider the two nonconvex optimization problems

$$\alpha = \inf\{f(x) : x \in C, g(x) = 0\} \qquad (Q)$$

$$\alpha(\tau) = \inf\{f(x) + \tau g(x) : x \in C, g(x) \geq 0\} \qquad (Q_\tau)$$

**Theorem 2.43** ***(Le Thi et al. [122])***
*Let $C$ be a non-empty bounded polyhedral convex set in $\mathbb{R}^n$ and let $f, g$ be finite concave function on $C$. Suppose that the feasible set of $(Q)$ is not empty. Then there exists $\tau_0 \geq 0$ such that for all $\tau > \tau_0$, the problems $(Q)$ and $(Q_\tau)$ are identical. Furthermore, we can take $\tau_0 = \frac{f(x_0) - \alpha(0)}{m}$, with $m = inf\{g(x) : \ x \in V(C), g(x) > 0\}$ and any $x_0 \in C, g(x_0) = 0$. Here, the convention $\inf_{\emptyset} g(x) = +\infty$ is used.*

It is easy to see that Theorem 2.42 is a special case of Theorem 2.43 when $g$ is a function of positive values or zero on $C$. Indeed, if $g$ is a function of positive values or zero on $C$, we have

$$\alpha = \inf\{f(x) : x \in C, g(x) \leq 0\} = \inf\{f(x) : x \in C, g(x) = 0\},$$

and

$$\alpha(\tau) = \inf\{f(x) + \tau g(x) : x \in C, g(x) \geq 0\} = \inf\{f(x) + \tau g(x) : x \in C\}.$$

# Chapter 3

# Cross-Entropy Method

## 3.1 Introduction

### Importance sampling

Importance sampling (see e.g. [181], Section 5.6) involves choosing a sampling distribution that favors important samples. Let

$$\ell = \mathbb{E}_f(H(X)) = \int H(x)f(x)dx, \tag{3.1}$$

be some expected performance measure of a computer simulation model, where $X$ is the input random vector with a probability density function $f$ and $H$ is the sample performance measure.

Let $g$ be another probability density such that $H.f$ is dominated by $g$. That is, if $g(x) = 0$ then $H(x)f(x) = 0$. Using the density $g$, we can represent $\ell$ as

$$\ell = \int H(x)\frac{f(x)}{g(x)}g(x)dx = \mathbb{E}_g\left(H(X)\frac{f(x)}{g(x)}\right), \tag{3.2}$$

where the subscript $g$ means that the expectation is taken with respect to $g$. Such a density is called the *importance sampling* density, *proposal* density, or *instrumental* density (as we use $g$ as an instrument to obtain information about $\ell$). Consequently, if $X^1, X^2, ..., X^N$ is *random sample* from $g$, that is, $X^1, X^2, ..., X^N$ are independent and identically distributed (iid) random vectors with density $g$, then

$$\widehat{\ell} = \frac{1}{N}\sum_{i=1}^{N} H(X^i)\frac{f(X^i)}{g(X^i)} \tag{3.3}$$

is an unbiased estimator of $\ell$. This estimator is called the *importance sampling estimator*. The ratio of densities

$$W(x) = \frac{f(x)}{g(x)} \tag{3.4}$$

is called the *likelihood ratio*. For this reason the importance sampling estimator is also called the *likelihood ratio estimator*. In particular case where there is no change of measure, that is, $g = f$, we have $W = 1$, and the likelihood ratio estimator in (3.3) reduces to the usual crude Monte Carlo (CMC) estimator.

## The Cross-Entropy method

The Cross-Entropy (CE) method is a Monte Carlo simulation method. It takes its name from the Kullback-Leiber distance, also called cross-entropy, an information measure widely used in various engineering fields including artificial intelligence, for instance, neural networks, etc.

The CE method is introduced by D. Lieber, R. Rubinstein, and D. Elmakis in order to evaluate the probability of rare events. In fact, the evaluation of such probabilities is a crucial, difficult problem for many stochastic systems. In addition, there are few problems for which it is possible to evaluate directly. In most cases, using a simulation method is needed. However, if the law to evaluate is used directly for the rare event, a simple Monte Carlo becomes quickly unusable because of the large number of samples. A better way to estimate this probability is to use *importance sampling* (IS), which is well-known variance reduction technique in which the system is simulated under a different set of parameters -or, more generally, a different probability distribution- so as to make the occurrence of the rare event more likely. A major drawback of the IS technique is that the optimal reference (also called tilting) parameters to be used in IS are usually very difficult to obtain. The advantage of the CE method is that it provides a simple adaptive procedure for estimating the optimal reference parameters. Recently the CE method has been successfully applied to the estimation of rare event in dynamic models, in particular queueing models involving both *light* and *heavy* tail input distribution [26, 27, 102].

Also, it has been demonstrated that the CE is particularly suitable for solving "hard" optimization problems, especially combinatorial problems. Indeed, these problems are often solved by Branch and Bound. But when the problem is too complex, exhaustive methods are not able to be applied. On the other hand, obtaining near-optimal solution, in practice, is often satisfactory. Thus, when deterministic methods failed to find the optimal solution within a reasonable amount of computation, the CE can find optimal or near optimal solution in reasonable time.

Unlike most of the stochastic algorithms for optimization which are based on local search, the CE method is a global random search procedure. The CE method was successfully applied to various problems such as the traveling salesman problem [177], the bipartition problem [177], the maximal cut problem [179], the image matching [55], the image segmentation [56], etc. To use the CE method for solving a deterministic optimization problem, this problem must be first translated into a stochastic one. The set of feasible solutions is then regarded as a set of events subjected to an importance density. Thus, the rare event simulation technique is applied.

In the two next sections, we present the methodology of the CE method for the rare event simulation, and for optimization. For a comprehensive overview and history of the CE method, the reader is referred to [28, 180, 181].

## 3.2 The CE for the rare event simulation

In this section we review the main ideas behind the CE algorithm for rare event simulation (for more details, see [28, 180, 181]).

Let $X = (X_1, ..., X_n)$ be the random vector taking values in the space $\mathcal{X}$. Let $\{f(\cdot; v)\}$ be the family of probability density functions (pdfs) on $\mathcal{X}$, with respect to some base measure $\nu$. Here $v$ is a real-valued parameter (vector). Thus,

$$\mathbb{E}H(X) = \int_{\mathcal{X}} H(x) f(x; v) \nu(dx),$$

for any mesurable function $H$. In most (or all) application $\nu$ is either a counting measure or Lebesgue measure. In the former case $f$ is often called a probability mass function, but in this thesis we use the generic terms density or pdf. For the rest of this section, we take for simplicity $\nu(dx) = dx$.

Let $S$ be some real-valued function on $\mathcal{X}$. Suppose that we are interested in the probability that $S(x)$ is greater than or equal to some real number $\gamma$, under $f(\cdot; u)$. This probability can be expressed as

$$\ell = \mathbb{P}_u(S(X) \geq \gamma) = \mathbb{E}_u I_{\{S(X) \geq \gamma\}}.$$

If this probability is small, say smaller than $10^{-5}$, we call $\{S(X) \geq \gamma\}$ a rare event.

A direct way to estimate $\ell$ is to use crude Monte-Carlo simulation: Draw a $N$ samples $X^1, X^2, ..., X^N$ from $f(\cdot; u)$, then

$$\frac{1}{N} \sum_{i=1}^{N} I_{\{S(X^i) \geq \gamma\}}$$

is a unbiased estimator of $\ell$. However, this poses serious problem when $\{S(X^i) \geq \gamma\}$ is a rare event. In that case, a large simulation effort is required in order to estimate $\ell$ accurately, i.e., with small relative error or a narrow confidence interval.

An alternative is based on importance sampling: take a random sample $X^1, X^2, ..., X^N$ from an *importance sampling* (different) density $g$ on $\mathcal{X}$, and evaluate $\ell$ using likelihood ratio estimator:

$$\widehat{\ell} = \frac{1}{N} \sum_{i=1}^{N} I_{\{S(X^i) \geq \gamma\}} \frac{f(X^i; u)}{g(X^i)}. \tag{3.5}$$

It is well known that the best way to estimate $\ell$ is to use the change of measure with density

$$g^*(x) = \frac{I_{\{S(X)\geq\gamma\}} f(x;u)}{\ell}.$$

Namely, by using this change of measure we have in (3.5)

$$I_{\{S(X)\geq\gamma\}} \frac{f(X^i;u)}{g(X^i)} = \ell, \tag{3.6}$$

for all $i$. In other words, the estimator (3.5) has zero variance, and we need to produce only $N = 1$ sample.

The obvious difficulty is of course that this $g^*$ depends on the unknown parameter $\ell$. Moreover, it is often convenient to choose a $g$ in the family of densities $\{f(\cdot;v)\}$. The idea now is to choose the parameter vector, called the *reference parameter* (sometimes called tilting parameter) $v$ such that the distance between the density $g^*$ and $f(\cdot;v)$ is minimal. A particular convenient measure of distance between two densities $g$ and $h$ is Kullback-Leibler distance, which is also termed the cross-entropy between $g$ and $h$. The Kullback-Leibler distance is defined as:

$$\mathcal{D}(g,h) = \mathbb{E}_g \ln \frac{g(X)}{h(X)} = \int g(x) \ln g(x) dx - \int g(x) \ln h(x) dx.$$

We note that $\mathcal{D}$ is not "distance" in the formal sense, for instance, it is not symmetric.

Minimizing the Kullback-Leibler distance between $g^*$ in (3.6) and $f(\cdot;v)$ is equivalent to choosing $v$ such that $-\int g^*(x) \ln f(x;v) dx$ is minimized, which is equivalent to solving the maximization problem

$$\max_v \int g^*(x) \ln f(x;v) dx. \tag{3.7}$$

Substituting $g^*$ from (3.6) into (3.7) we obtain the maximization program

$$\max_v \int \frac{I_{\{S(x)\geq\gamma\}} f(x;u)}{\ell} \ln f(x;v) dx, \tag{3.8}$$

which is equivalent to the program

$$\max_v D(v) := \mathbb{E}_u I_{\{S(X)\geq\gamma\}} \ln f(X;v). \tag{3.9}$$

Using again importance sampling, with a change of measure $f(\cdot;w)$, we can rewrite (3.9) as

$$\max_v D(v) = \max_v \mathbb{E}_w I_{\{S(X)\geq\gamma\}} W(X;u,w) \ln f(X;v), \tag{3.10}$$

for any reference parameter $w$, where

$$W(x;u,w) = \frac{f(x;u)}{f(x;w)}$$

is the likelihood ratio, at $x$, between $f(\cdot;u)$ and $f(\cdot;w)$. The optimal solution of (3.10) can be written as

$$v^* \in \arg\max_v \mathbb{E}_w I_{\{S(X)\geq\gamma\}} W(X;u,w)\ln f(X;v), \tag{3.11}$$

We may estimate $v^*$ by solving the following stochastic program (also called stochastic counterpart of (3.10))

$$\max_v \widehat{D}(v) = \max_v \frac{1}{N}\sum_{i=1}^{N} I_{\{S(X^i)\geq\gamma\}} W(X^i;u,w)\ln f(X^i;v), \tag{3.12}$$

where $X^1, X^2, ..., X^N$ is a random sample from $f(\cdot;w)$. In typical applications, the function $\widehat{D}$ in (3.12) is concave and differentiable with respect to $v$ (see [175]), and thus, the solution of (3.12) may be readily obtained by solving (with respect to $v$) the following system of equation

$$\frac{1}{N}\sum_{i=1}^{N} I_{\{S(X^i)\geq\gamma\}} W(X^i;u,w)\nabla\ln f(X^i;v) = 0, \tag{3.13}$$

where the gradient is with respect to $v$.

The advantage of this approach is that the solution of (3.13) can be calculated analytically. In particular, this happens if the distributions of the random variables belong to a natural exponential family (NEF). For further details, see [180].

It is important to note that the stochastic program (3.12) is useful only in the case where the probability of the "target event" $\{S(X)\geq\gamma\}$ is not too small, say $\ell \geq 10^{-5}$. For rare event probabilities, however (when, say, $\ell < 10^{-5}$), the program (3.12) is difficult to carry out. Namely, due to the rareness of the events $\{S(X)\geq\gamma\}$, most of the indicator random variables $I_{\{S(X^i)\geq\gamma\}}, i = 1, ..., N$ will be zero, for moderate $N$. The same holds for the derivatives of $\widehat{D}(v)$ as given in the left-hand side of (3.13).

A multi-level algorithm can be used to overcome this difficulty. The idea is to construct a sequence of reference parameters $\{v_t, t\geq 0\}$ and a sequence of levels $\gamma_t, t\geq 1$, and iterate in both $\gamma_t$ and $v_t$ (see Algorithm 1 below).

We initialize by choosing a not very small $\rho$, say $\rho = 10^{-2}$ and by defining $v_0 = u$. Next, we let $\gamma_1(\gamma_1 < \gamma)$ be such that, under the original density $f(x;u)$, the probability $\ell_1 = \mathbb{E}_u\{S(X)\geq\gamma_1\}$ is at least $\rho$. We then let $v_1$ be the optimal CE reference parameter for estimating the pair $\{l, v^*\}$. In other words, each iteration of the algorithm consist of two main phases. In the first phase, $\gamma_t$ is updated, in the second $v_t$ is updated. Specially, starting with $v_0 = u$, we obtain the subsequent $\gamma_t$ and $v_t$ as follows:

1. **Adaptive updating of $\gamma_t$.** For a fixed $v_{t-1}$, let $\gamma_t$ be the $(1-\rho)$-quantile of $S(X)$ under $v_{t-1}$. That is, $\gamma_t$ satisfies

$$\mathbf{P}_{v_{t-1}}(S(X)\geq\gamma_t)\geq\rho, \tag{3.14}$$

$$\mathbf{P}_{v_{t-1}}(S(X) \leq \gamma_t) \geq 1 - \rho, \tag{3.15}$$

where $X \sim f(\cdot; v_{t-1})$.
A simple estimator of $\gamma_t$, denote $\widehat{\gamma}_t$ can be obtained by drawing a random sample $X^1, X^2, ..., X^N$ from $f(\cdot; v_{t-1})$. Suppose that $S(X^{\sigma(1)}) \leq S(X^{\sigma(2)}) \leq ... \leq S(X^{\sigma(N)})$, where $\sigma$ is a permutation of the set $\{1, ..., N\}$. Evaluating the $(1-\rho)$-quantile of $S(X)$ as

$$\widehat{\gamma}_t = S_{\lfloor (1-\rho)N \rfloor}. \tag{3.16}$$

2. **Adaptive updating of** $v_t$. For a fixed $\gamma_t$ and $v_{t-1}$, derive $v_t$ by minimizing the Kullback-Leibler distance, or equivalent to solving the following program

$$\max_v \mathbf{E}_{v_{t-1}} I_{\{S(X) \geq \gamma_t\}} W(X; u, v_{t-1}) \ln f(X; v), \tag{3.17}$$

where

$$W(x; u, v_{t-1}) = \frac{f(x; u)}{f(x; v_{t-1})}.$$

The stochastic counterpart of (3.28) is as follows: for fixed $\widehat{\gamma}_t$ and $\widehat{v}_{t-1}$ (the estimate of $v_{t-1}$), derive $\widehat{v}_t$ from the following program

$$\max_v D(v) := \frac{1}{N} \sum_{i=1}^{N} I_{\{S(X^i) \geq \gamma_t\}} W(X^i; u, v_{t-1}) \ln f(X^i; v). \tag{3.18}$$

Thus, at the first iteration, starting with $\widehat{v}_0 = u$, to get a good estimate for $\widehat{v}_1$, the target event is artificially made less rare by (temporarily) using a level $\widehat{\gamma}_1$ which is chosen smaller than $\gamma$. The value for $\widehat{v}_1$ obtained in this way will (hopefully) make the event $\{S(X) \geq \gamma\}$ less rare in the next iteration, so in the next iteration a value $\widehat{\gamma}_2$ can be used which is closer to $\gamma$ itself. The algorithm terminates when at some iteration $t$ a level is reached which is at least $\gamma$ and thus the original value of $\gamma$ can be used without getting too few samples.
As mentioned before, the optimal solutions of (3.17) and (3.18) can often be obtained analytically, in particular when $f(x; v)$ belongs to a NEF.

The above rationale results in the following algorithm.
**Algorithm 1: Main CE Algorithm for rare event simulation**

1. Defin $\widehat{v}_0 = u$, and $0 < \rho < 1$. Set $t = 1$.

2. Generate $N$ samples $X^1, X^2, ..., X^N$ according to $f(\cdot; \widehat{v}_{t-1})$, and compute $(1-\theta)$-quantile $\widehat{\gamma}_t$ of $S$ according to (3.16).

3. Using the same samples $X^1, X^2, ..., X^N$ to solve the stochastic programming (3.17). Denote the solution by $\widetilde{v}_t$.

4. If $\widetilde{v}_t < \gamma$, set $t = t + 1$, and iterate from step 2. Else proceed with step 5.

5. Estimate the rare-event probability $\ell$ using likelihood ratio estimate

$$\widehat{\ell} = \frac{1}{N_1} \sum_{i=1}^{N_1} I_{\{S(X^i) \geq \gamma_t\}} W(X^i; u, \widehat{v}_T), \tag{3.19}$$

where $T$ denotes the final number of iterations.

We have intentionally used the notation $\widetilde{\gamma}_t$ and $\widetilde{v}_t$ in Algorithm 1 above, rather than the more obvious $\gamma_t$ and $v_t$, in order to distinguish it from its deterministic counterpart, which is obtained by replacing sample means and sample quantiles by expectations and quantiles.

**Remark 3.1** *(Static Simulation) The above method has been formulated for finite-dimensional random vector only. This is sometimes referred to as static simulation. For infinite-dimensional random vectors or stochastic processes we need a more subtle treatment.*

*We will not go into detail here, but the main point is that Algorithm 1 holds true without much alteration and can be readily applied to estimation problems involving both light and heavy tail distribution [10, 27, 180]*

**Remark 3.2** *(Variance minimization) An alternative way to obtain a good reference parameter is to choose $v$ such that the variance, or equivalently, the second moment, of the importance sampling estimator is minimal. In other words we wish to find*

$$_*v = arg\min_v \mathbb{E}_v [I_{\{S(X) \geq \gamma\}} W(X; u, v)]^2. \tag{3.20}$$

*More generally, using again the principe of importance sampling, this is equivalent to finding*

$$_*v = arg\min_v \mathbb{E}_w I_{\{S(X) \geq \gamma\}} W(X; u, v) W(X; u, w) \tag{3.21}$$

*for any reference parameter $v_t$. As in (3.11), we can estimate $_*v$ as the solution to the stochastic program*

$$\min_v \widehat{V}(v) = \min_v \frac{1}{N} \sum_{i=1}^{N} I_{\{S(X^i) \geq \gamma\}} W(X^i; u, v) W(X^i; u, w), \tag{3.22}$$

*where $X^1, X^2, ..., X^N$ is a random sample from $f(\cdot; w)$. However, the evaluation of (3.22) in general involves complicated numerical optimization, and it is much more convenient to use the closed-form updating formulas that follow from CE minimization.*

## 3.3 The Cross-Entropy method for optimization

The generic CE scheme for optimization problems can be described as follows (for more detail, see [28, 180, 181]).

Suppose that we wish to maximize a real-valued performance function $S$ over a set $\mathcal{X}$. Denote the maximum by $\gamma^*$, thus

$$\gamma^* = \max_{x \in \mathcal{X}} S(x). \tag{3.23}$$

The starting point in the methodology of the CE method is to associate an estimation problem with the optimization problem (3.23). To this end one defines a collection of indicator functions $I_{\{S(x) \geq \gamma\}}$ on $\mathcal{X}$ for various thresholds or levels $\gamma \in \mathbb{R}$. Next, let $\{f(\cdot; v), v \in V\}$ be a family of (discrete) probability density functions (pdfs) on $\mathcal{X}$, parameterized by a real-valued (vector) $v$.

For some $u \in V$, we consider the *associated stochastic problem* (ASP):

$$\begin{aligned} \ell(\gamma) &= \mathbf{P}_u(S(x) \geq \gamma) = \sum_{x \in \mathcal{X}} I_{\{S(x) \geq \gamma\}} f(x; u) \\ &= \mathbf{E}_u I_{\{S(x) \geq \gamma\}}, \end{aligned} \tag{3.24}$$

where $\mathbf{P}_u$ is the probability measure under which the random state $\mathcal{X}$ has the pdf $f(\cdot; u)$, and $\mathbf{E}_u$ denotes the corresponding expectation operator. The idea of CE method is to construct simultaneously a sequence of levels $\widehat{\gamma}_1, \widehat{\gamma}_2, ..., \widehat{\gamma}_T$ and parameters (vectors) $\widehat{v}_1, \widehat{v}_2, ..., \widehat{v}_T$ such that $\widehat{\gamma}_T$ is close to the optimal $\gamma^*$ and $\widehat{v}_T$ is such that the corresponding density assigns high probability mass to the collection of states that give a high value. More specifically, we initialize by setting $v_0 = u$, choosing a not very small quantity $\theta$, and then we proceed as follows:

1. **Adaptive updating of $\gamma_t$.** For a fixed $v_{t-1}$, let $\gamma_t$ be the $(1-\theta)$-quantile of $S(X)$ under $v_{t-1}$. That is, $\gamma_t$ satisfies

$$\mathbf{P}_{v_{t-1}}(S(X) \geq \gamma_t) \geq \theta, \tag{3.25}$$

$$\mathbf{P}_{v_{t-1}}(S(X) \leq \gamma_t) \geq 1 - \theta, \tag{3.26}$$

   where $X \sim f(\cdot; v_{t-1})$.
   A simple estimator of $\gamma_t$, denote $\widehat{\gamma}_t$ can be obtained by drawing a random sample $X^1, X^2, ..., X^N$ from $f(\cdot; v_{t-1})$. Suppose that $S(X^{\sigma(1)}) \leq S(X^{\sigma(2)}) \leq ... \leq S(X^{\sigma(N)})$, where $\sigma$ is a permutation of the set $\{1, ..., N\}$. Evaluating the $(1-\theta)$-quantile of $S(X)$ as

$$\widehat{\gamma}_t = S_{\lfloor (1-\theta)N \rfloor}. \tag{3.27}$$

2. **Adaptive updating of $v_t$.** For a fixed $\gamma_t$ and $v_{t-1}$, derive $v_t$ by minimizing the Kullback-Leibler distance, or equivalent to solving the next program

$$\max_v \mathbf{E}_{v_{t-1}} I_{\{S(X) \geq \gamma_t\}} W(X^i; u, v_{t-1}) \ln f(X; v), \tag{3.28}$$

   where

$$W(x; u, v_{t-1}) = \frac{f(x; u)}{f(x; v_{t-1})}.$$

The stochastic counterpart of (3.28) is as follows: for fixed $\widehat{\gamma}_t$ and $\widehat{v}_{t-1}$ (the estimate of $v_{t-1}$), derive $\widehat{v}_t$ from following program

$$\max_v D(v) := \frac{1}{N}\sum_{i=1}^{N} I_{\{S(X^i)\geq\gamma_t\}} W(X^i;u,v_{t-1})\ln f(X^i;v). \tag{3.29}$$

In typical applications, the function $D$ is concave and differentiable with respect to $v$, and thus the updating equation (3.29) is equivalent to solving the following system of equations:

$$\frac{1}{N}\sum_{i=1}^{N} I_{\{S(X^i)\geq\gamma_t\}} W(X^i;u,v_{t-1})\nabla\ln f(X^i;v) = 0, \tag{3.30}$$

where the gradient is with respect to $v$.

**Remark** (**smoothed updating**): Instead of updating the parameter $v$ directly via the solution of (3.29) we use the following smoothed version

$$\widehat{v}_t = \alpha\widetilde{v}_t + (1-\alpha)\widehat{v}_{t-1}, t = 1,2,..., \tag{3.31}$$

where $\widetilde{v}_t$ is the parameter vector from the solution of (3.29), and $\alpha$ is called the smoothing parameter, with $0.7 \leq \alpha \leq 1$. The reason for using this smoothed is: (a) to smooth out the value of $\widehat{v}_t$, (b) to reduce the probability that some component $\widehat{v}_{t,i}$ of $\widehat{v}_t$ will be zero or one at the first few iterations. This is particularly important when $\widehat{v}_t$ is a vector or matrix of probabilities.

**CE Algorithm for Optimization**

1. Choose $\widehat{v}_0$, and $0 < \theta < 1$. Set $t = 1$.
2. Generate $N$ samples $X^1, X^2, ..., X^N$ according to $f(\cdot;\widehat{v}_{t-1})$, and compute $(1-\theta)$-quantile $\widehat{\gamma}_t$ of $S$ according to (3.27).
3. Using the same samples $X^1, X^2, ..., X^N$ to solve the stochastic programming (3.29). Denote the solution by $\widetilde{v}_t$.
4. Applying (3.31) to smooth out the vector $\widetilde{v}_t$.
5. Repeat step 2-4 until a pre-precified stopping criterion is met.

To apply the Cross-Entropy method for an optimization problem, the most important point is how to find an appropriate family of pdfs, say $\{f(\cdot;v), v \in V\}$, on the feasible set of this optimization problem such that updating the parameters could be done as easily as possible. In fact, it seems that there is no common way to do that, since it depends strongly on the structure of the feasible set. In this thesis, when using the Cross-Entropy method, we will propose such families of pdfs for the considered problems.

# Part II

# Modeling and Approaches based on DC programming&DCA and the Cross-Entropy method for solving some classes of nonconvex programs in Search Theory, Assignment and Portfolio Management

# A deterministic optimization approach for planning a multisensor multizone search for a (moving) target

# A deterministic optimization approach for planning a multisensor multizone search for a target

Le Thi Hoai An[a,*], Nguyen Duc Manh[b], Pham Dinh Tao[b]

[a]*Laboratory of Theoretical and Applied Computer Science, UFR MIM, Paul Verlaine University - Metz, Ile du Saulcy, 57045 Metz, France*
[b]*Laboratory of Modelling, Optimization and Operations Research, National Institute for Applied Sciences - Rouen, 76801 Saint-Etienne-du-Rouvray, France*

**Abstract**

In this paper, we consider a well-known problem in the general area of search theory: planning a multisensor in multizone search so as to maximize the probability of detection of a target under a given resource effort to be shared. We propose a new optimization model that is a nonlinear mixed 0-1 programming problem. This problem is then reformulated as a DC (Difference of Convex functions) program via a penalty technique. DC programming and DCA (DC algorithm) have been investigated for solving the resulting DC program. Numerical experiments demonstrate the efficiency and the superiority of the proposed algorithm in comparison with the existing method.

*Keywords:* Search theory, Hierarchical optimization, Combinatorial optimization, DC programming and DCA, nonlinear mixed 0-1 programming, exact penalty

## 1. Introduction

Search theory is defined by Cadre and Soiris [3] as a discipline that treats the problem of how a missing object can be searched optimally, when the amount of searching time is limited and only probabilities of the possible position of the missing object are given. The theory of how to search for missing objects has been a subject of serious scientific research for more than

*Corresponding author
*Email address:* lethi@univ-metz.fr ( Le Thi Hoai An )
[1]Phone: +33 387315441 Fax: +33 387315309

50 years. It is a branch of the broader applied science known as operations research [6].

In fact, Search theory was first established during World War II by the work of B. O. Koopman and his colleagues [12] in the Antisubmarine Warfare Operations Research Group (ASWORG). The applications of search theory were first made on military operations [19]. Koopman [11] stated that the principles of search theory could be applied effectively to any situation where the objective is to find a person or object contained in some restricted geographic area. After military applications, it was also applied to different problems such as surveillance, explorations, medicine, industry and search and rescue operations [8]. The aim of searching in the context of Aeronautical Search and Rescue (ASAR), for instance, is to find the missing aircraft effectively and as quickly as possible with the available resources [18].

A search theory problem, in general, is characterized by three pieces of data ([3]):

- the probabilities of the searched object (the "target") being in various possible locations;
- the local detection probability that a particular amount of local search effort could detect the target;
- the total amount of searching effort available.

The problem is to find the optimal distribution of this total effort that maximizes the probability of detection.

Stone [18] summarized major steps in the development of search theory, for instance, stationary target problems, moving target problems, optimal searcher path algorithms, and dynamic search games. The last item (search games) is the primary focus of recent research, including numerous sub-domains such as mobile evaders, avoiding target, ambush games, inspection games, and tactical games. For moving target problems, decisive progress has been made in developing search strategies that maximize the probability of detecting the (moving) target within a fixed amount of time. In particular, Brown [2] and Washburn [21] have proposed an iterative algorithm in which the motion space and the time frame have been discretized, and the search effort available in each period is infinitely divisible between the grid cells of the target motion space. In this approach, the search effort available in each period is bounded above by a constant that does not depend on the allocations made during any other periods.

In the "classical" search theory, the target is said to be detected if a detection occurs any time during the measurement epoch. Here, the target track will be said to be detected if (multiple) elementary detections occur at various times, and this is the fundamental difference. Thus, there is a test for acceptance (or detection) of a target track [5, 20]. Track detection is also associated with a spatio-temporal modeling of the target track. A quite common hypothesis in classical search theory is that the objective function, which is generally the non-detection probability, is a convex functional relatively to the search efforts. This is quite a reasonable assumption in this context, which ensures convergence of the iterative algorithms.

In this paper we consider an instance of search theory problems that can be described as follows: suppose that a space of search is partitioned into zones of reasonable size. A unique sensor must be able to explore efficiently a whole zone. Each zone is itself partitioned into cells. A cell is an area in which all points have the same properties, according to the difficulty of detection (altitude, vegetation, etc.). Each sensor has its own coefficient of visibility over a cell. The visibility coefficients depend also on the kind of the target that is searched. Here, there is a unique target to detect. The objective is to allot sensors to search zones and to find the best resource sharing for every sensor so as to maximize the probability of detection of a target.

The optimization model of this problem is hierarchical:

- At upper level: finding the best allotment of sensors to search zones (a sensor is allotted to a unique zone);
- At lower level: determining the best resource sharing for every sensor, in order to have an optimal surveillance over the allotted zone.

At the upper level, the objective function can be non-convex or implicitly defined via an algorithm applied to the lower level. This makes the problem very hard. In [17], Simonin et al. have proposed a hierarchical approach for solving this problem where a cross-entropy (CE) algorithm [1, 4, 16] has been developed for the upper level while an optimization method based on the algorithm of de Guenin [7] for detecting a stationary target has been used in the lower level. Besides this paper, we do not find in the literature any other work considering this problem. We can say that, up to now, there are no deterministic models and methods for solving this problem.

In this work we develop a *deterministic continuous optimization approach* based on DC (Difference of Convex functions) programming and DCA (DC optimization Algorithms). The contribution of the paper is 3-fold:

Firstly, we propose in this paper an original deterministic mathematical model for the problem. By introducing the binary variables (the assignment variables) we formulate this hierarchical optimization problem as a mixed $0-1$ nonlinear program. Due to the very large dimension of this problem in practice and the non-linearity of the objective function, the standard methods in combinatorial optimization such as branch and bound, branch and cut, cutting plan cannot be applied. Attempting to develop robust numerical solution approaches, we try to reformulate the problem as a DC program.

Secondly, we investigate an exact penalty technique to reformulate the mixed $0-1$ nonlinear program into a continuous optimization problem that is in fact a DC program. We prove also that a penalty parameter can be computed explicitly.

Thirdly, we investigate DC programming and DCA for solving the related DC program.

DC programming and DCA ([13–15] and references therein) aim to solve a general DC program that takes the form

$$\alpha = \inf\{f(x) := g(x) - h(x) : x \in \mathbb{R}^p\} \quad (P_{dc}) \tag{1}$$

where $g, h$ are lower semicontinuous proper convex functions on $\mathbb{R}^p$. Such a function $f$ is called a DC function, and $g-h$, a DC decomposition of $f$ while $g$ and $h$ are the DC components of $f$. The construction of DCA involves DC components $g$ and $h$ but not the function $f$ itself: each iteration $k$ of DCA consists of computing

$$y^k \in \partial h(x^k), x^{k+1} \in \arg\min\{g(x) - h(x^k) - \langle x - x^k, y^k\rangle : x \in \mathbb{R}^p\} \quad (P_k).$$

Hence, for a DC program, each DC decomposition corresponds to a different version of DCA. Since a DC function $f$ has an infinite number of DC decompositions which have crucial impacts on the qualities (speed of convergence, robustness, efficiency, globality of computed solutions,...) of DCA, the search for a "good" DC decomposition is important from an algorithmic point of view. Moreover, despite its local character, DCA with a good initial point can converge to global solutions. Finding a "good" initial point is then also an important stage of DCA. How to develop an efficient algorithm based on

the generic DCA scheme for a practical problem is thus a judicious question to be studied, and the answer depends on the specific structure of the problem being considered. In the current work, using an appropriate DC decomposition we propose a DCA scheme that is very inexpensive in term of CPU time thanks to the rapidity of the algorithm for solving the subproblem $(P_k)$.

The paper is organized as follows. The problem statement is presented in Section 2. Section 3 is devoted to the reformulation of the considered problem in terms of a deterministic continuous optimization problem. The solution method via DC programming and DCA is investigated in Section 4. The numerical results are reported in Section 5 while some conclusions and perspectives are discussed in Section 6.

## 2. Problem statement

First, let us introduce the notation employed in the paper.
$E$ : space of search, $Z$ : number of zones, $S$ : number of sensors
$z$ : zone index
$i$ : cell index
$s$ : sensor index
$\alpha$ : prior on the initial location of the target
$\Phi_s$ : quantity of resource available for sensor $s$ to search the space
$w_{i,z,s}$ : coefficient that characterizes the acuity of sensor $s$ over cell $i$ of the zone $z$ (visibility coefficient).

We state below the problem as described in [17] (see [17] for more details):

*The space of search*: the search space, named $E$, is a large space with spatially variable search characteristics. The search space $E$ is divided into $Z$ search zones, denoted $E_z, z = 1, 2, ..., Z$, each of them is partitioned into $C_z$ cells, denoted $\{c_{i,z}\}_{i=1}^{C_z}$ so that:

$$E = \bigcup_{z=1}^{Z} E_z, \quad E_z \cap E_{z'} = \emptyset, \forall z \neq z',$$

$$E_z = \bigcup_{i=1}^{C_z} c_{i,z}, \quad c_{i,z} \cap c_{j,z} = \emptyset, \forall i \neq j.$$

A cell $c_{i,z}$ represents the smallest search area in which the search parameters are constant. For example, it can be a part of land with constant character-

istics (latitude, landscape). Each zone must have a reasonable size in order to be explored by a sensor within a fixed time interval.

*The target*: the target is hidden in one unit of the search space. Its location is characterized by a prior $\alpha_{i,z}$. Thus, we have

$$\sum_{z=1}^{Z}\sum_{i=1}^{C_z}\alpha_{i,z}=1.$$

*The means of search*: means of search can be passive (e.g. IRST, ESM) or active sensors (radars). We will consider that searching the target will be carried out by $S$ sensors. Due to operational constraints, each sensor $s \in S$ must be allotted to a unique search zone. For example, it could be the exploration time to share between units of a zone. At the lower level, for the sensor $s$ allotted to the zone $E_z$, the amount of search resource allocated to the cell $c_{i,z}$ is denoted $x_{i,z,s}$. It can represent the time spent on searching the cell $c_{i,z}$ (passive sensor), the intensity of emissions or the number of pulses (active sensors), etc. Furthermore, each sensor $s$ has a search amount $\Phi_s$, it means that if the sensor $s$ is allotted to the zone $E_z$ then:

$$\sum_{i=1}^{C_z} x_{i,z,s} \leq \Phi_s.$$

To characterize the effectiveness of the search at the cell level, we consider the conditional non-detection probability $\bar{P}_s(x_{i,z,s})$ which represents the probability of not detecting the target given that the target is hidden in $c_{i,z}$ and that we apply an elementary search effort $x_{i,z,s}$ on $c_{i,z}$. Some assumptioms are made to model $\bar{P}_s(x_{i,z,s})$. For all sensors, the function $x_{i,z,s} \mapsto \bar{P}_s(x_{i,z,s})$ is convex and non-increasing (law of diminishing return). Assuming independence of elementary detections, a usual model is $\bar{P}_s(x_{i,z,s}) = \exp(-w_{i,z,s}x_{i,z,s})$, where $w_{i,z,s}$ is a (visibility) coefficient which characterizes the reward for the search effort put in $c_{i,z}$ by the sensor $s$.

An additional assumption is that sensors act independently at the cell level. This means that if $S$ sensors are allotted to $c_{i,z}$ the probability of not detecting a target hidden in $c_{i,z}$ is simply $\prod_{s=1}^{S}\bar{P}_s(x_{i,z,s})$.

Let $m : \{1,2,...,S\} \rightarrow \{1,2,...,Z\}$ be a mapping allotting sensors to search zones. Our aim is to find both the optimal mapping $m$ (i.e., the best

sensor-to-zone allotment) and the optimal local distributions $x$ in order to minimize the non-detection probability. The objective function is then

$$f(m,x) = \sum_{z=1}^{Z} \sum_{i=1}^{C_z} \alpha_{i,z} \prod_{s \in m^{-1}(z)} \bar{P}_s(x_{i,z,s}),$$

which leads to the following problem [17]:

$$\begin{cases} \min\limits_{m,x} f(m,x) = \sum\limits_{z=1}^{Z} \sum\limits_{i=1}^{C_z} \alpha_{i,z} \prod\limits_{s \in m^{-1}(z)} \bar{P}_s(x_{i,z,s}) \\ \text{s.t.} \quad \sum\limits_{i=1}^{C_z} x_{i,z,s} \leq \Phi_s, \forall z, \forall s \in m^{-1}(z), \\ \qquad x_{i,z,s} \geq 0, \forall i \in C_z, \forall s = 1, ..., S, \forall z = 1, ..., Z, \\ \qquad m \text{ mapping} : \{1, 2, ..., S\} \to \{1, 2, ..., Z\}. \end{cases} \tag{2}$$

This problem is hierarchical:

- At upper level: finding the best allotment of sensors to search zones (a sensor is allotted to a unique zone);
- At lower level: determining the best resource sharing for every sensor.

## 3. A deterministic formulation of Problem (2)

Let us introduce the allocation variable $u_{z,s}$ defined by

$$u_{z,s} = \begin{cases} 1 & \text{if the sensor } s \text{ is allotted to the zone } z, \\ 0 & \text{otherwise.} \end{cases}$$

Let variable $x_{i,z,s}$ be the quantity of resource of the sensor $s$ allotted to the cell $c_{i,z}$ in the zone $z$. We can rewrite (2) in the following form:

$$\begin{cases} \min\limits_{x,u} (f(x,u) = \sum\limits_{z=1}^{Z} \sum\limits_{i=1}^{C_z} \alpha_{i,z} \prod\limits_{s=1}^{S} \bar{P}_s(x_{i,z,s} u_{z,s})) \\ \text{s.t.} \sum\limits_{i=1}^{C_z} x_{i,z,s} \leq \Phi_s, z = 1, ..., Z, s = 1, ..., S, \\ \qquad \sum\limits_{z=1}^{Z} u_{z,s} = 1, s = 1, ..., S, \\ \qquad u_{z,s} \in \{0,1\}, x_{i,z,s} \geq 0, s = 1, ..., S, z = 1, ..., Z, i = 1, ..., C_z. \end{cases} \tag{3}$$

The constraints $\sum_{z=1}^{Z} u_{z,s} = 1, \forall s = 1, 2, ..., S$ mean that each sensor $s \in S$ must be allotted to a unique search zone.

With the usual model $\bar{P}_s$ [10]:

$$\bar{P}_s(x_{i,z,s}) = \exp(-w_{i,z,s}x_{i,z,s}),$$

the objective function $f$ becomes:

$$f(x,u) = \sum_{z=1}^{Z}\sum_{i=1}^{C_z} \alpha_{i,z} \exp(-\sum_{s=1}^{S} w_{i,z,s}x_{i,z,s}u_{z,s}).$$

Set

$$d = S.(C_1 + C_2 + ... + C_z), n = Z.S.$$

Let

$$D = \{x = (x_{i,z,s}) \in \mathbb{R}^d_+ : \sum_{i=1}^{C_z} x_{i,z,s} \leq \Phi_s, z = 1, ..., Z, s = 1, ..., S\}$$

and

$$M = \{u = (u_{z,s}) \in \{0,1\}^{Z.S} : \sum_{z=1}^{Z} u_{z,s} = 1, s = 1, ..., S\},$$

then we can write the problem (3) in the form:

$$(P) \quad \begin{cases} \min\limits_{x,u} \sum\limits_{z=1}^{Z}\sum\limits_{i=1}^{C_z} \alpha_{i,z} \exp(-\sum\limits_{s=1}^{S} w_{i,z,s}x_{i,z,s}u_{z,s}) \\ \text{s.t. } x \in D, u \in M, \end{cases}$$

which is a nonlinear mixed 0-1 programming problem. It is easy to see that the objective function of $(P)$, say $f$, is convex in $x$ for each fixed $u$, and similarly, it is convex in $u$ for each fixed $x$. Moreover, $f$ is infinitely differentiable.

## 4. Solution Method

We first reformulate (P) as a DC program by using a penalty technique.

*4.1. DC Reformulation*

Consider the function $p$ and the bounded polyhedral convex set $K$ defined, respectively, by:

$$p(u) = \sum_{z=1}^{Z} \sum_{s=1}^{S} u_{z,s}(1 - u_{z,s}),$$

and

$$K = \{u = (u_{z,s}) \in [0,1]^{Z.S} : \sum_{z=1}^{Z} u_{z,s} = 1, s = 1, ..., S\}.$$

We notice that $p$ is finite and concave on $\mathbb{R}^{Z.S}$, non-negative on $K$ and

$$M = \{u \in K : p(u) \leq 0\}.$$

Hence Problem $(P)$ can be rewritten as

$$\alpha = \min\{f(x,u) : x \in D,\ u \in K,\ p(u) \leq 0\}.$$

The exact penalty result is given by the following theorem.

**Theorem 1.** *(i) Let*

$$t^0 = \max\{||\nabla_u^2(f(x,u))|| : \quad u \in [0,1]^n, x \in D\},$$

*then* $\forall t > t^0$ *the problem (P)*

$$\min\{f(x,u) : x \in D, u \in M\} \quad (P)$$

*is equivalent to the problem*

$$\min\{f(x,u) + tp(u) : x \in D, u \in K\}. \quad (P_t)$$

*in the following sense: they have the same optimal value of the objective function and the same optimal solution set.*

*(ii) If* $(x^*, u^*)$ *is a local minimum to problem* $(P_t)$ *then* $(x^*, u^*)$ *is a feasible solution to the problem* $(P)$*.*

Proof. Let $\varphi(u) := \min_{x \in D}(f(x,u) + tp(u))$. By definition of $t^0$, for each fixed $x$ in $D$, the function $f(x,u) + t^0 p(u)$ is concave in $u$, thus the function

$\min_{x\in D}(f(x,u)+t^0p(u))$ is concave in $u$, and consequently, for any $t>t^0$, $\varphi$ is strongly concave in $u$.

If $(x^*,u^*)$ is an optimal solution to the problem $(P)$, then $p(u^*)=0$ and

$$f(x,u)\geq f(x^*,u^*),\forall x\in D,\forall u\in M. \tag{4}$$

Hence

$$f(x,u)+tp(u)\geq f(x,u)\geq f(x^*,u^*)+tp(u^*),\forall x\in D,\forall u\in M \tag{5}$$

which implies that

$$\varphi(u)\geq\varphi(u^*),\forall u\in M.$$

Since $extr(K)=M$ (see **Lemma 2** below**)**, and $\varphi$ is concave, we have

$$\min\{\varphi(u):u\in K\}=\min\{\varphi(u):u\in M\}. \tag{6}$$

Combining (5) and (6) we obtain

$$f(x,u)+tp(u)\geq\ f(x^*,u^*)+tp(u^*),\forall x\in D,\forall u\in K. \tag{7}$$

So $(x^*,u^*)$ is also an optimal solution to the problem $(P_t)$.

Conversely, suppose that $(x^*,u^*)$ is an optimal solution of the problem $(P_t)$, i.e. (7) holds. Thus $\varphi(u)\geq\varphi(u^*),\forall u\in K$. Since $\varphi$ is strongly concave, $u^*$ is an extreme point of $K$ and therefore $u^*\in M$. It means that $(x^*,u^*)$ is a feasible solution of the problem $(P)$. This and (7) imply (4), i.e., $(x^*,u^*)$ is an optimal solution to $(P)$.

(ii) If $(x^*,u^*)$ is a local minimum of problem $(P_t)$, then there exist a neighbourhood $V$ of $x^*$, $U$ of $u^*$, with $U$ is convex and compact such that $f(x,u)+tp(u)\geq f(x^*,u^*)+tp(u^*),\forall x\in D\cap V,\forall u\in K\cap U$. Let $\psi$ be the function defined by

$$\psi(u)=\min_{x\in D\cap V}(f(x,u)+tp(u)).$$

It is clear that $\psi$ is strongly concave in $u\in K\cap U$, and $\psi(u)\geq\psi(u^*),\forall u\in K\cap U$. So $u^*$ is an extreme point of the convex set $K\cap U$, thus $u^*$ is an extreme point of the convex set $K$, say $u^*\in M$. Therefore $(x^*,u^*)$ is a feasible solution to the problem $(P)$. □

Note that the part (ii) in this theorem is very useful for DCA applied to $(P_t)$ because DCA usually produces a local minimum.

**Lemma 2.** *We have $extr(K) = M$, where $extr(K)$ is the set of extreme points of $K$. Hence $co(M) = K$.*

PROOF. The set $K$ can be expressed as the product of $S$ simplex $\Delta$, say

$$K = \Delta \times \Delta \times ... \times \Delta \text{ with } \Delta := \left\{ u^s = (u_{z,s})_z \in [0,1]^Z : \sum_{z=1}^{Z} u_{z,s} = 1 \right\}.$$

It is easy to see that

$$extr(\Delta) = \left\{ u^s = (u_{z,s})_z \in \{0,1\}^Z : \sum_{z=1}^{Z} u_{z,s} = 1 \right\}.$$

Hence

$$extr(K) = extr(\Delta) \times extr(\Delta) \times ... \times extr(\Delta) = M.$$

Since $D$ is compact and $f \in C^\infty$, $\max\{||\nabla_u^2(f(x,u))|| : \quad u \in [0,1]^n, x \in D\}$ exists. The following proposition gives an estimation of $t^0$ :

**Proposition 3.** *We have*

$$t^0 := \max\{||\nabla_u^2(f(x,u))|| : \quad u \in [0,1]^n, x \in D\} \leq \Phi\alpha w^2 \sum_{s=1}^{S} \Phi_s,$$

*where*

$$\alpha = \max\{\alpha_{i,z} : z = 1, ..., Z, i = 1, ..., C_z\}, \Phi = \max\{\Phi_s : 1, ..., S\},$$

$$w = \max\{w_{i,z,s} : z = 1, ..., Z, i = 1, ..., C_z, s = 1, ..., S\}.$$

PROOF. see appendix.

In the following sections we will investigate DCA for solving problem $(P_t)$ with $t^0 = \Phi\alpha w^2 \sum_{s=1}^{S} \Phi_s$.

*4.2. DC Algorithm (DCA)*

*4.2.1. Outline of DC Programming and DCA*

Pham Dinh Tao in 1985 introduced DC programming and DCA, which constitute the backbone of (smooth or nonsmooth) nonconvex programming and global optimization. Le Thi Hoai An and Pham Dinh Tao since 1994 extensively developed them. And now DCA is a classic and increasingly popular approach (see [13–15] and references therein). DC programming and DCA address the problem of minimizing a function $f$ which is a difference of convex functions on the whole space $\mathbb{R}^p$ or on a convex set $C \subset \mathbb{R}^p$. Generally speaking, a DC program takes the form

$$\alpha = \inf\{f(x) := g(x) - h(x) : x \in \mathbb{R}^p\} \quad (P_{dc}) \tag{8}$$

where $g, h$ are lower semicontinuous proper convex functions on $\mathbb{R}^p$. Such a function $f$ is called DC function, and $g - h$, DC decomposition of $f$ while $g$ and $h$ are DC components of $f$. The convex constraint $x \in C$ can be incorporated in the objective function of $(P_{dc})$ by using the indicator function on $C$ denoted $\chi_C$ which is defined by

$$\chi_C(x) = \begin{cases} 0 \text{ if } x \in C \\ +\infty, \text{ otherwise.} \end{cases}$$

If either $g$ or $h$ is a polyhedral convex function then $(P_{dc})$ is called a polyhedral DC program.

Let

$$g^*(y) := \sup\{\langle x, y\rangle - g(x) : x \in \mathbb{R}^p\}$$

be the conjugate function of $g$. Then, the following program is called the dual program of $(P_{dc})$:

$$\alpha_D = \inf\{h^*(y) - g^*(y) : y \in \mathbb{R}^p\}. \quad (D_{dc}) \tag{9}$$

One can prove that $\alpha = \alpha_D$, (see e.g. [13, 14]) and there is the perfect symmetry between primal and dual DC programs: the dual to $(D_{dc})$ is exactly $(P_{dc})$.

DCA is based on the local optimality conditions of $(P_{dc})$, namely

$$\partial h(x^*) \cap \partial g(x^*) \neq \emptyset \tag{10}$$

(such a point $x^*$ is called a *critical point* of $g - h$), and

$$\emptyset \neq \partial h(x^*) \subset \partial g(x^*). \tag{11}$$

The condition (11) is necessary for local optimality of $(P_{dc})$ (or the generalized KKT condition). It is also sufficient for many classes of DC programs that include some cases quite often encountered in practice (see e.g. [13, 14]).

The idea of DCA is simple: each iteration of DCA approximates the concave part $-h$ by its affine majorization (that corresponds to taking $y^k \in \partial h(x^k)$) and minimizes the resulting convex function (that is equivalent to determining $x^{k+1} \in \partial g^*(y^k)$).

**Generic DCA scheme**

**Initialization:** Let $x^0 \in \mathbb{R}^p$ be a best guess, $0 \leftarrow k$.

**Repeat**

Calculate $y^k \in \partial h(x^k)$

Calculate $x^{k+1} \in \arg\min\{g(x) - h(x^k) - \langle x - x^k, y^k\rangle : x \in \mathbb{R}^p\} \quad (P_k)$

$k + 1 \leftarrow k$

**Until** convergence of $x^k$.

It is important to mention the following main convergence properties of DCA:

- DCA is a descent method (the sequences $\{g(x^k) - h(x^k)\}$ and $\{h^*(y^k) - g^*(y^k)\}$ are decreasing) *without linesearch*;
- If the optimal value $\alpha$ of the problem $(P_{dc})$ is finite and the infinite sequences $\{x^k\}$ and $\{y^k\}$ are bounded then every limit point $x^*$ (resp. $y^*$) of the sequence $\{x^k\}$ (resp. $\{y^k\}$) is a critical point of $g - h$ (resp. $h^* - g^*$).
- DCA has a *linear convergence* for general DC programs.
- DCA has a finite convergence for polyhedral DC programs.

The solution of a practical nonconvex program by DCA must have two stages: the search of an *appropriate* DC decomposition and the search of a *good* initial point. An appropriate DC decomposition, in our sense, is the one that corresponds to a DCA which is not expensive and has interesting convergence properties.

In the next subsection we will develop an instance of DCA to solve the problem $(P_t)$, and study the convergence properties of the proposed algorithm.

*4.2.2. Description of the DCA applied to* $(P_t)$

From the computations in the Appendix we have

$$||H(f)||_\infty \leq \max\{S\alpha w^2 + S\alpha w^2\Phi + \alpha w, 2\Phi\alpha w^2 \sum_{s=1}^{S} \Phi_s + Nw\alpha\}.$$

where

$$N = \max\{C_z : z = 1, ..., Z\}.$$

Hence, let $\rho := \max\{S\alpha w^2 + S\alpha w^2\Phi + \alpha w, 2\Phi\alpha w^2 \sum_{s=1}^{S} \Phi_s + Nw\alpha\}$, a DC formulation of $(P_t)$ can be

$$\min\left\{g(x,u) - h(x,u) : (x,u) \in D \times K\right\}, \tag{12}$$

where

$$g(x,u) := \frac{\rho}{2}||(x,u)||^2 \text{ and } h(x,u) := \frac{\rho}{2}||(x,u)||^2 - f(x,u) - tp(u).$$

DCA applied to DC program (12) consists of computing, at each iteration $k$, the two sequences $\left\{(y^k, v^k)\right\}$ and $\left\{(x^k, u^k)\right\}$ such that $(y^k, v^k) \in \partial h(x^k, u^k)$ and $(x^{k+1}, y^{k+1})$ is an optimal solution of the following convex quadratic program:

$$\min\left\{\frac{\rho}{2}||(x,u)||^2 - \langle(x,u),(y^k,v^k)\rangle : (x,u) \in D \times K\right\}$$

which can be decomposed into two subproblems

$$\min\left\{\frac{\rho}{2}||x||^2 - \langle x, y^k\rangle : x \in D\right\} \tag{13}$$

and

$$\min\left\{\frac{\rho}{2}||u||^2 - \langle u, v^k\rangle : u \in K\right\}. \tag{14}$$

We are now in a position to summarize the DCA for solving Problem $(P_t)$

**Step 1.** Initialization: let $(x^0, u^0)$ satisfy the constraints of the problem. Choose $\epsilon_1 > 0, \epsilon_2 > 0$ and $k = 0$.

**Step 2.** Compute $(y^k, v^k) = \nabla h(x^k, u^k)$, with

$$y^k = \rho x^k - \nabla_x f(x^k, u^k),\ v^k = \rho u^k - \nabla_u f(x^k, u^k) + t(2u^k - e).$$

**Step 3.** Compute $(x^{k+1}, u^{k+1})$ by solving the two convex quadratic problems (13) and (14)

**Step 4.** Iterate Step 2 and 3 until

$$|f(x^{k+1}, u^{k+1}) - f(x^k, u^k)| \leq \epsilon_1(1 + |f(x^{k+1}, u^{k+1})|)$$
$$\text{or } ||(x^{k+1}, u^{k+1}) - (x^k, u^k)||_\infty \leq \epsilon_2(1 + ||(x^{k+1}, u^{k+1})||_\infty).$$

**Theorem 4.** *(Convergence properties of Algorithm DCA. For simplicity's sake, we omit the dual part of these properties).*

*i) DCA generates the sequence* $\left\{(x^k, u^k)\right\}$ *such that the sequence* $\left\{f(x^k, u^k)\right\}$ *is decreasing.*

*ii) The point* $\left\{(x^k, u^k)\right\}$ *verifies the generalized condition of the problem (12).*

Proof. Direct consequences of the convergence properties of DCA for general DC programs and the fact that the function $h$ is differentiable.

**Efficient ways for solving problems (13) and (14)**

In fact, we can continuously divide the problems (13) and (14) into some smaller problems with unique constraints (except positive constraints). By denoting $y^k = (y_{i,z,s}) \in \mathbb{R}^d, v^k = (v_{z,s}) \in \mathbb{R}^{Z.S}$, solving the problem (13) is replaced by solving $S.Z$ subproblems with unique inequality constraints

$$\begin{cases} \min\limits_{x_{i,z,s}} \sum\limits_{i=1}^{C_z} \left( x_{i,z,s}^2 - \dfrac{2y_{i,z,s}^k}{\rho} x_{i,z,s} \right) \\ s.t.\ \sum\limits_{i=1}^{C_z} x_{i,z,s} \leq \Phi_s, \\ \quad x_{i,z,s} \geq 0, \forall i = 1, 2, ..., C_z, \end{cases} \tag{15}$$

and solving the problem (14) is replaced by solving $S$ subproblems with unique equality constraints

$$\begin{cases} \min\limits_{u_{z,s}} \sum\limits_{z=1}^{Z} \left( u_{z,s}^2 - \dfrac{2v_{z,s}^k}{\rho} u_{z,s} \right) \\ s.t.\ \sum\limits_{z=1}^{Z} u_{z,s} = 1, \\ \quad u_{z,s} \geq 0, \forall z = 1, 2, ..., Z. \end{cases} \tag{16}$$

The problem (16) can be solved efficiently by the Block Pivotal Principal Pivoting Algorithm [9]. We develop below an efficient algorithm for solving the problem (15).

**An efficient algorithm for solving (15)**
We consider the quadratic convex problem:

$$\begin{cases} \min\limits_{x} \sum\limits_{i=1}^{n} \left( \frac{1}{2} x_i^2 - a_i x_i \right) \\ s.t. \ \sum\limits_{i=1}^{n} x_i \leq \Phi, \\ \qquad x_i \geq 0, \forall i = 1, 2, ..., n. \end{cases} \tag{17}$$

Set

$$I = \{i \in \{1, 2, ..., n\} : a_i > 0\}.$$

We firstly see that if $\sum\limits_{i \in I} a_i \leq \Phi$ then $x^* = (x_1^*, ..., x_n^*)$ with $x_i^* = a_i$ if $i \in I$ else $x_i^* = 0$, is an optimal solution for Problem (17).

In contrast, i.e.,

$$\sum_{i \in I} a_i > \Phi, \tag{18}$$

then there exists $\lambda \geq 0$ such that

$$S(\lambda) := \sum_{i \in I_\lambda} (a_i - \lambda) > \Phi,$$

where

$$I_\lambda = \{i \in I : a_i > \lambda\}.$$

Indeed, we can choose, for instance,

$$0 \leq \lambda < \min \left\{ \frac{1}{|I|} \left( \sum_{i \in I} a_i - \Phi \right), a_i, i \in I \right\}. \tag{19}$$

We consider Karush-Kuhn-Tucker condition for Problem (17)

$$(KKT) \begin{cases} x_i - a_i + \lambda - t_i = 0, i = 1, 2, ..., n, \\ \lambda(\sum\limits_{i=1}^{n} x_i - \Phi) = 0, \\ t_i x_i = 0, i = 1, 2, ..., n, \\ \lambda \geq 0, \\ t_i \geq 0, i = 1, 2, ..., n. \end{cases}$$

Suppose that $x = (x_1, ..., x_n)$ and $\lambda \geq 0$ is the solution for $(KKT)$ equations. If $x_i > 0$, by the first condition, then

$$x_i - a_i + \lambda = 0, \quad i \in J, \tag{20}$$

where

$$J = \{i \in \{1, 2, ..., n\} : x_i > 0\}.$$

Thus

$$x_i = a_i - \lambda, \ a_i > \lambda, \quad i \in J.$$

With the condition (18), we have also $\lambda > 0$. The second condition of $(KKT)$ leads to:

$$\sum_{i \in J} x_i = \Phi, \tag{21}$$

From (20) and (21), we have

$$\lambda = \frac{1}{|J|} \left( \sum_{i \in J} a_i - \Phi \right).$$

In this case, we present the following algorithm to solve this problem:

**Step 1. Initiation**: let $\lambda_1 = \max\{a_i, i \in I\} + \alpha, \alpha > 0$ arbitrary, $\lambda_2$ satisfying the condition (19), and $\epsilon > 0$.

**Step 2. Iteration**
**Repeat**

- Compute $\lambda = \dfrac{\lambda_1 + \lambda_2}{2}$,
- Set

$$\begin{cases} x_i = a_i - \lambda & \text{if } i \in I_\lambda, \\ x_i = 0 & \text{otherwise,} \end{cases}$$

  where

$$I_\lambda = \{i \in I : a_i > \lambda\}.$$

- Compute $S(\lambda) = \sum_{i \in I_\lambda} x_i$.
- Compare $S(\lambda)$ with $\Phi$, and set

$$\begin{cases} \lambda_2 = \lambda & \text{if } S(\lambda) > \Phi + \epsilon, \\ \lambda_1 = \lambda & \text{if } S(\lambda) < \Phi - \epsilon. \end{cases}$$

**Until** $(|S - \Phi| \leq \epsilon)$.
Note that by choosing $\lambda_1, \lambda_2$ in Step 1, we have $S(\lambda_1) = 0 < \Phi < S(\lambda_2)$.

## 5. Numerical experimentation

The search space is the lake of Laouzas in France [17]. The number of zones is $Z = 4$ and there are $n = 30$ cells in each zone. We have $S = 6$ sensors. All sensors have the same amount of resource $\Phi$. The coefficients are given in Table 1.

The program is written in C using Microsoft Visual C++ 2008, and implemented on a notebook with chipset Intel(R) Core(TM) Duo CPU 2.0 GHz, RAM 3GB.

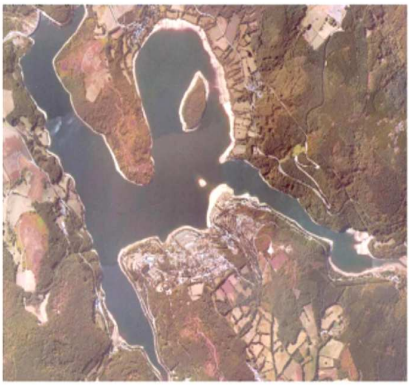

Figure 1: An aerial photograph of the lake of Laouzas

| Type of Cell | Prior of target | Visibility of sensors | | | | | |
|---|---|---|---|---|---|---|---|
| | | Sensor 1 | Sensor 2 | Sensor 3 | Sensor 4 | Sensor 5 | Sensor 6 |
| Forest | 0.0085 | 0.4 | 0.5 | 0.6 | 0.8 | 0.5 | 0.1 |
| Water | 0.001 | 0.9 | 0.1 | 0.1 | 0.1 | 0.3 | 0.5 |
| Plat land | 0.0115 | 0.3 | 0.1 | 0.4 | 0.6 | 0.5 | 0.2 |
| Rough land | 0.013 | 0.2 | 0.7 | 0.8 | 0.2 | 0.4 | 0.6 |
| Very rough land | 0.014 | 0.1 | 0.6 | 0.7 | 0.1 | 0.3 | 0.5 |
| Town | 0 | 0.8 | 0.9 | 0.1 | 0.7 | 0.6 | 0.2 |

Table 1: Data

To find a starting point of DCA, we take $(x^0, u^0)$, where $x^0 = (x^0_{i,z,s})$,

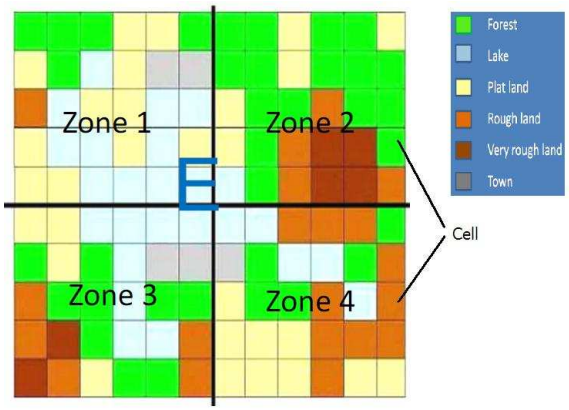


Figure 2: Partition of the lake Laouzas

$u^0 = (u^0_{z,s})$ are as follows:

$$x^0_{i,z,s} = \frac{\Phi}{n},\ u^0_{z,s} = \frac{1}{Z}, s = 1, ..., S, z = 1, ..., Z, i = 1, ..., C_z,$$

and run DCA for solving $(P_t)$ with the parameters $\rho$ and *penalty* quite small (depending on the parameters $\Phi$). We will get a new pair $(x^1, u^1)$ which becomes the starting point of the main DCA. In the implementation of the main DCA, we choose $\epsilon_1 = \epsilon_2 = 10^{-7}$, and some values of parameters $\rho$ and *penalty* are given in Table 2. In Table 3 we compare the results obtained by

| Amount of | The first DCA | | The main DCA | |
|---|---|---|---|---|
| resource $\Phi$ | Rho | Penalty | Rho | Penalty |
| $\Phi = 5$ | 0.378 | 0.001701 | 3.78 | 1.701 |
| $\Phi = 10$ | 1.3986 | 0.006804 | 13.986 | 6.804 |
| $\Phi = 15$ | 1.5498 | 0.007655 | 30.996 | 15.309 |
| $\Phi = 20$ | 1.0962 | 0.013698 | 54.81 | 27.216 |
| $\Phi = 25$ | 0.85428 | 0.004253 | 85.428 | 42.525 |
| $\Phi = 30$ | 0.61425 | 0.006124 | 122.85 | 61.236 |
| $\Phi = 35$ | 0.83538 | 0.008335 | 167.076 | 83.349 |
| $\Phi = 40$ | 0.72702 | 0.010886 | 218.106 | 108.864 |

Table 2: Parameters

DCA and the CE algorithm [17]. The main idea of CE algorithm is to generate particular allotments of sensors to search zones that will be evaluated and then selected, in order to obtain a drawing law which will converge toward the optimal allotment. The steps of this CE algorithm can be described as follows:

**Step 1.** Initialize $M = M_0 = (p^{M_0}(z|s))$ a uniform distribution, i.e.,

$$p^{M_0}(z|s) = \frac{1}{Z}, s = 1, ..., S, z = 1, ..., Z,$$

and choose $\theta \in (0, 1)$. (Note that $p(z|s)$ represents the probability to assign sensor $s$ to zone $z$.)
**Step 2.** Generate $N$ allotment mappings $m_1, m_2, ..., m_N$ according to $M$, and compute $f(m_k) := \min_x f(m_k, x), k = 1, 2, ..., N$, where $f(m, x)$ is objective function of (2). (In other words, we obtain $f(m_k)$ by solving Problem (2) when $m_k$ is known.)
**Step 3.** Sort the sequence $\{f(m_k)\}_{k=1}^N$ in the increasing orders. Let $f(m_{\sigma(1)}) \leq f(m_{\sigma(2)}) \leq ... \leq f(m_{\sigma(N)})$, where $\sigma$ is a permutation of the set $\{1, 2, ..., N\}$. Set $T = \lfloor \theta N \rfloor$, then choose $T$ best draws $X^{\sigma(1)}, X^{\sigma(2)}, ..., X^{\sigma(T)}$.
**Step 4.** Update $M$ by the formula

$$p^M(z|s) := \frac{\text{card}\{k \in \{\sigma(1), \sigma(2), ..., \sigma(T)\} : \ x_s^{\sigma(k)} = z\}}{T},$$

**Step 5.** Iterate step 2, 3, 4 until convergence.

Since CE is a heuristic method, for each parameter $\Phi$ we run the CE algorithm 10 times and then take the average for the 10 results. The parameters for CE are as follows: for each iteration, number of samples is $N = 50$, $\theta = 0.3$, and the number of iterations is limited to 15. We observe that the DCA produced better solutions than the CE while the CPU time is shorter.

Figure 3 shows the results (non-detection probability) obtained by DCA when the amount of resource, $\Phi$, varies. We can see that there is a negative relationship between $\Phi$ and the non-detection probability. When $\Phi = 110$, the non-detection probability is 0.060488.

## 6. Conclusion

We have presented a new approach for solving the problem of planning a multisensor multizone search for a target. This is the first time a determin-

| Amount of resource $\Phi$ | DCA method Objective function | Time | CE method Objective function | Time |
|---|---|---|---|---|
| $\Phi = 5$ | 0.818115 | 1.406s | 0.818577 | 1.192s |
| $\Phi = 10$ | 0.684478 | 1.281s | 0.684701 | 1.427s |
| $\Phi = 15$ | 0.579858 | 1.281s | 0.580673 | 1.925s |
| $\Phi = 20$ | 0.495232 | 2.500s | 0.500550 | 2.284s |
| $\Phi = 25$ | 0.426436 | 2.437s | 0.428900 | 2.689s |
| $\Phi = 30$ | 0.368320 | 2.265s | 0.372219 | 2.620s |
| $\Phi = 35$ | 0.322050 | 2.422s | 0.327574 | 2.285s |
| $\Phi = 40$ | 0.282366 | 2.421s | 0.292742 | 3.688s |

Table 3: Comparative results between DCA and CE

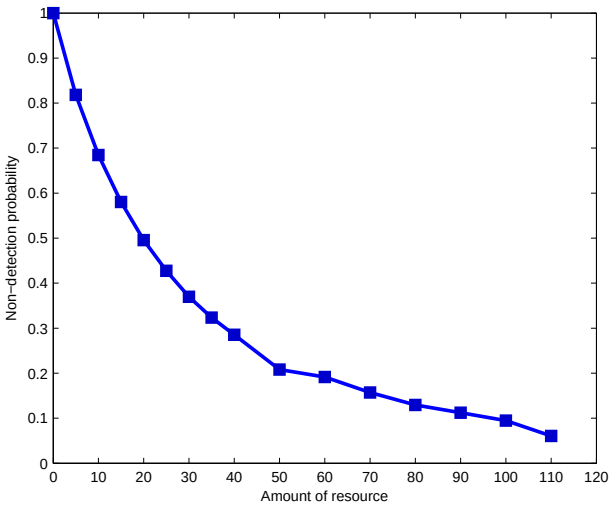


Figure 3: Relation of non-detection probability given by DCA and amount of resource

istic optimization model is introduced in the literature for solving this problem. This constitutes an interesting contribution of the paper. For solving the combinatorial optimization problem by DCA, an innovative continuous approach in nonconvex programming framework, we first reformulated the original problem in the form of a DC program. We prove an exact penalty result in which the penalty parameter can be estimated. That is our second important contribution. The third contribution deals with the development of an efficient DCA scheme for solving the resulting problem. In this scheme, the subconvex programs is decomposed on some simpler convex programs that can be solved by fast algorithms. Numerical results show that our approach can solve this problem more efficiently than the CE method, an efficient algorithm for the problem. Whenever there is a good starting point, we can get a better local solution or even a global solution. In future, we will investigate DCA for solving the moving target case of this problem.

*Appendix*

**Estimate the norm of Hessian matrix of function $f(x,u)$ and** $\max\{||\nabla_u^2(f(x,u))||_\infty : \ u \in [0,1]^n, x \in D\}$.

$x = \{x_{i,z,s}\}_{i,z,s} = (x_{1,1,1}, ..., x_{C_1,1,1}, x_{1,2,1}, ..., x_{C_2,2,1}, ..., x_{1,Z,1}, ..., x_{C_Z,Z,1}, x_{1,1,2}, ..., x_{C_1,1,2}, ..., x_{1,Z,2}, ..., x_{C_Z,Z,2}, ..., x_{1,1,S}, ..., x_{C_1,1,S}, ..., x_{1,Z,2}, ..., x_{C_Z,Z,S}) \in \mathbb{R}^p$.

$u = \{u_{z,s}\}_{z,s} = (u_{1,1}, u_{2,1}, ..., u_{Z,1}, u_{1,2}, u_{2,2}, ..., u_{Z,2}, u_{1,S}, u_{2,S}, ..., u_{Z,S}) \in \mathbb{R}^{Z.S}$.

We have

$$\frac{\partial f}{\partial x_{i_1,z_1,s_1}}(x,u) = -\alpha_{i_1,z_1} w_{i_1,z_1,s_1} u_{z_1,s_1} \exp(-\sum_{s=1}^{S} w_{i_1,z_1,s} x_{i_1,z_1,s} u_{z_1,s}),$$

$$\frac{\partial^2 f}{\partial x_{i_1,z_1,s_1} \partial x_{i_2,z_2,s_2}}(x,u) =$$

$$\begin{cases} 0 \quad \text{if } (i_2,z_2) \neq (i_1,z_1) \\ \alpha_{i_1,z_1} w_{i_1,z_1,s_1} u_{z_1,s_1} w_{i_1,z_1,s_2} u_{z_1,s_2} \exp(-\sum_{s=1}^{S} w_{i_1,z_1,s} x_{i_1,z_1,s} u_{z_1,s}) \\ \qquad\qquad\qquad\qquad\qquad\qquad \text{if } (i_2,z_2) = (i_1,z_1) \end{cases}$$

$$\frac{\partial^2 f}{\partial x_{i_1,z_1,s_1}\partial u_{z_3,s_3}}(x,u) =$$

$$\begin{cases} 0 \text{ if } z_3 \neq z_1 \\ \alpha_{i_1,z_1} w_{i_1,z_1,s_1} u_{z_1,s_1} w_{i_1,z_1,s_3} x_{i_1,z_1,s_3} \exp(-\sum_{s=1}^{S} w_{i_1,z_1,s} x_{i_1,z_1,s} u_{z_1,s}) \\ \qquad\qquad \text{if } z_3 = z_1, s_3 \neq s_1 \\ (-\alpha_{i_1,z_1} w_{i_1,z_1,s_1} + \alpha_{i_1,z_1} w_{i_1,z_1,s_1}^2 u_{z_1,s_1} x_{i_1,z_1,s_1}) \exp(-\sum_{s=1}^{S} w_{i_1,z_1,s} x_{i_1,z_1,s} u_{z_1,s}) \\ \qquad\qquad \text{if } z_3 = z_1, s_3 = s_1. \end{cases}$$

On the line $(i_1, z_1, s_1)$ of the matrix Hessian of the function $f$, we have the sum of absolute of all elements:

$$S_{i_1,z_1,s_1} = \sum_{s_2=1}^{S} |\alpha_{i_1,z_1} w_{i_1,z_1,s_1} u_{z_1,s_1} w_{i_1,z_1,s_2} u_{z_1,s_2}| \exp(-\sum_{s=1}^{S} w_{i_1,z_1,s} x_{i_1,z_1,s} u_{z_1,s}) +$$

$$\sum_{s_3=1}^{S} (|\alpha_{i_1,z_1} w_{i_1,z_1,s_1} u_{z_1,s_1} w_{i_1,z_1,s_3} x_{i_1,z_1,s_3}| + |\alpha_{i_1,z_1} w_{i_1,z_1,s_1}|) \exp(-\sum_{s=1}^{S} w_{i_1,z_1,s} x_{i_1,z_1,s} u_{z_1,s})$$

$$= (\sum_{s_2=1}^{S} |\alpha_{i_1,z_1} w_{i_1,z_1,s_1} u_{z_1,s_1} w_{i_1,z_1,s_2} u_{z_1,s_2}| + \sum_{s_3=1}^{S} |\alpha_{i_1,z_1} w_{i_1,z_1,s_1} u_{z_1,s_1} w_{i_1,z_1,s_3} x_{i_1,z_1,s_3}| +$$

$$|\alpha_{i_1,z_1} w_{i_1,z_1,s_1}|) \exp(-\sum_{s=1}^{S} w_{i_1,z_1,s} x_{i_1,z_1,s} u_{z_1,s}).$$

Let

$$\alpha = \max\{\alpha_{i,z} : z = 1, 2, ..., Z, i = 1, 2, ..., C_z\}, \Phi = \max\{\Phi_s : 1, 2, ..., S\},$$

$$w = \max\{w_{i,z,s} : z = 1, 2, ..., Z, i = 1, 2, ..., C_z, s = 1, 2, ..., S\},$$

$$N = \max\{C_z : z = 1, 2, ..., Z\}.$$

Because $0 \leq u_{z,s} \leq 1, \forall s \in S, z \in Z$, and $\exp(-\sum_{s=1}^{S} w_{i_1,z_1,s} x_{i_1,z_1,s} u_{z_1,s}) \leq 1$, we have

$$S_{i_1,z_1,s_1} \leq S\alpha w^2 + S\alpha w^2 \Phi + \alpha w$$
$$= S\alpha w^2(\Phi + 1) + \alpha w, \forall z_1 = 1, 2, ..., Z, i_1 = 1, 2, ..., C_{z_1}, s_1 = 1, 2, ..., S.$$

We have

$$\frac{\partial f}{\partial u_{z_1,s_1}}(x,u) = -\sum_{i=1}^{C_{z_1}} \alpha_{i,z_1} w_{i,z_1,s_1} x_{i,z_1,s_1} \exp(-\sum_{s=1}^{S} w_{i,z_1,s} x_{i,z_1,s} u_{z_1,s}),$$

$$\frac{\partial^2 f}{\partial u_{z_1,s_1} \partial u_{z_2,s_2}}(x,u) =$$

$$\begin{cases} 0 \quad \text{if } z_2 \neq z_1, \\ \sum_{i=1}^{C_{z_1}} \alpha_{i,z_1} w_{i,z_1,s_1} x_{i,z_1,s_1} w_{i,z_1,s_2} x_{i,z_1,s_2} \exp(-\sum_{s=1}^{S} w_{i,z_1,s} x_{i,z_1,s} u_{z_1,s}) \text{ if } z_2 = z_1. \end{cases}$$

$$\frac{\partial^2 f}{\partial u_{z_1,s_1} \partial x_{i_2,z_2,s_2}}(x,u) =$$

$$\begin{cases} 0 \quad \text{if } z_2 \neq z_1, \\ \alpha_{i_2,z_1} w_{i_2,z_1,s_1} x_{i_2,z_1,s_1} w_{i_2,z_1,s_2} u_{z_1,s_2} \exp(-\sum_{s=1}^{S} w_{i_2,z_1,s} x_{i_2,z_1,s} u_{z_1,s}) \\ \qquad\qquad \text{if } z_2 = z_1, s_2 \neq s_1, \forall i_2 \in z_1, \\ (-w_{i_2,z_1,s_1} \alpha_{i_2,z_1} + \alpha_{i_2,z_1} w_{i_2,z_1,s_1} x_{i_2,z_1,s_1} w_{i_2,z_1,s_1} u_{z_1,s_1}) \exp(-\sum_{s=1}^{S} w_{i_2,z_1,s} x_{i_2,z_1,s} u_{z_1,s}) \\ \qquad\qquad \text{if } z_2 = z_1, s_2 = s_1, \forall i_2 \in z_1. \end{cases}$$

On the line $(z_1, s_1)$ of the Hessian matrix of the function $f$, we have the sum of absolute of all elements:

$$\begin{aligned} S_{z_1,s_1} &= \sum_{s_2=1}^{S} \sum_{i=1}^{C_{z_1}} |\alpha_{i,z_1} w_{i,z_1,s_1} x_{i,z_1,s_1} w_{i,z_1,s_2} x_{i,z_1,s_2}| \exp(-\sum_{s=1}^{S} w_{i_2,z_1,s} x_{i_2,z_1,s} u_{z_1,s}) + \\ &\quad \sum_{s_2=1}^{S} \sum_{i_2=1}^{C_{z_1}} |\alpha_{i_2,z_1} w_{i_2,z_1,s_1} x_{i_2,z_1,s_1} w_{i_2,z_1,s_2} x_{i_2,z_1,s_2}| \exp(-\sum_{s=1}^{S} w_{i_2,z_1,s} x_{i_2,z_1,s} u_{z_1,s}) + \\ &\quad \sum_{i_2=1}^{C_{z_1}} |w_{i_2,z_1,s_1} \alpha_{i_2,z_1}| \exp(-\sum_{s=1}^{S} w_{i_2,z_1,s} x_{i_2,z_1,s} u_{z_1,s}) \\ &\leq \sum_{s_2=1}^{S} \sum_{i=1}^{C_{z_1}} \Phi \alpha w^2 x_{i,z_1,s_2} + \sum_{s_2=1}^{S} \sum_{i_2=1}^{C_{z_1}} \Phi \alpha w^2 x_{i_2,z_1,s_2} \ + Nw\alpha \\ &\leq 2\Phi \alpha w^2 \sum_{s=1}^{S} \Phi_s + Nw\alpha. \end{aligned}$$

The infinity norm of Hessian matrix of function $f$ satisfies

$$||H(f)||_\infty \leq \max\{S\alpha w^2 + S\alpha w^2\Phi + \alpha w, 2\Phi\alpha w^2 \sum_{s=1}^{S} \Phi_s + Nw\alpha\} = \rho.$$

On the line $(z_1, s_1)$ of the Hessian matrix $\nabla_u f(x,u)$ of the function $f(x,.)$, we have the sum of absolute of all elements

$$\begin{aligned} T_{z_1,s_1} &= \sum_{s_2=1}^{S} \sum_{i=1}^{C_{z_1}} |\alpha_{i,z_1} w_{i,z_1,s_1} x_{i,z_1,s_1} w_{i,z_1,s_2} x_{i,z_1,s_2}| \exp(-\sum_{s=1}^{S} w_{i_2,z_1,s} x_{i_2,z_1,s} u_{z_1,s}) \\ &\leq \sum_{s_2=1}^{S} \sum_{i=1}^{C_{z_1}} \Phi\alpha w^2 x_{i,z_1,s_2} \leq \Phi\alpha w^2 \sum_{s=1}^{S} \Phi_s, \forall x \in D, \forall u \in [0,1]^n. \end{aligned}$$

Thus, we have

$$\sup\{||\nabla_u^2(f(x,u))|| : \quad u \in [0,1]^n, x \in D\} \leq \Phi\alpha w^2 \sum_{s=1}^{S} \Phi_s.$$

## References


[1] deBoer PT, Kroese DP, Mannor S, Rubinstein RY. A Tutorial on The Cross-Entropy Method. Annals of Operations Research 2005;134:19-67.

[2] Brown SS. Optimal search for a moving target in discrete time and space. Operations Research 1979;32(5):1107-1115.

[3] Cadre JP, Souris G. Searching Tracks. IEEE Transactions on Aerospace and Electronic systems 2000;36(4):1149-1166.

[4] Costa A, Jones OD, Kroese D. Convergence properties of the cross-entropy method for discrete optimization. Operations Research Letters 2007;35(5):573-580.

[5] Dobbie JM. Transfer of detection contacts to tracking contacts in surveillance. Operations Research 1966;14:791-800.

[6] Frost JR. Principles of search theory, part III: Probability density distributions. Response 1999;17(3):1-10.

[7] deGuenin J. Optimum distribution of effort: an extension of the Koopman theory. Operations Research 1961;9(1):1-7.

[8] Haley KB, Stone LD (Eds.). Search Theory and Applications, New York: Plenum Press, 1980.

[9] Júdice J, Raydan M, et al. On the solution of the symmetric eigenvalue complementarity problem by the spectral projected gradient algorithm. Numerical Algorithms 2008;47(4):391-407.

[10] Koopman BO. Search and its optimization. Mathematical Monthly 1979;7:527-540.

[11] Koopman BO. Search and Screening: General Principles with Historical Applications. Pergamon Press; New York; 1980.

[12] Koopman BO. Search and Screening: General Principle with Historical Applications. Alexandria, VA: MORS Heritage Series, 1999.

[13] Le Thi HA, Pham Dinh T. The DC (difference of convex functions) Programming and DCA revisited with DC models of real world nonconvex optimization problems. Annals of Operations Research 2005;133:23-46.

[14] Pham Dinh T, Le Thi HA. Convex analysis approach to DC programming: Theory, Algorithms and Applications (dedicated to Professor Hoang Tuy on the occasion of his 70th birthday). Acta Mathematica Vietnamica 1997;22:289-355.

[15] Pham Dinh T, Le Thi HA. DC optimization algorihms for solving the trust region subproblem. SIAM J.Optimization 1998;8:476-505.

[16] Rubinstein RY, Kroese D. The cross-entropy method: a unified approach to combinatorial optimization, Monté Carlo simulation, and machine learning. Berlin: Springer; 2004.

[17] Simonin C, Le Cadre JP, Dambreville F. A hierarchical approach for planning a multisensor multizone search for a moving target. Computers and Operations Research 2009;36(7):2179-2192.

[18] Stone LD. What's happened in search theory since the 1975 Lanchester prize? Operations Research 1989;37(3):501-506.

[19] Stone LD. Theory of Optimal Search (2nd ed.). Arlington, VA: Operations Research Society of America. ORSA Books; 1989.

[20] Van Keuk G. Sequential track extraction. IEEE Transactions on Aerospace and Electronic Systems 1998;34(4):1135-1148.

[21] Washburn AR. Search for a moving target, The FAB algorithm. Operations Research 1983;31(4):739-751.

# A deterministic optimization approach for planning a multisensor multizone search for a moving target

LE THI Hoai An, NGUYEN Duc Manh and PHAM DINH Tao

**Abstract**

In this paper, we consider a well-known problem in the general area of search theory: planning a multisensor in multizone search so as to minimize the probability of non-detection of a moving target under a given resource effort to be shared. The solution method is based on a combination of the forward-backward split technique and a deterministic resolution for the general subproblem. Numerical experiments demonstrate the efficiency of the proposed algorithm in comparing with a standard method.

## 1 Introduction

Search theory is defined by Cadre and Soiris [3] as a discipline that treats the problem of how a missing object can be searched optimally, when the amount of searching time is limited and only probabilities of the possible position of the missing object are given. The theory of how to search for missing objects has been a subject of serious scientific research for more than 50 years. It is a branch of the broader applied science known as operations research [6].

Search theory was first established during World War II by the work of B. O. Koopman and his colleagues [13] in the Antisubmarine Warfare Operations Research Group (ASWORG). The applications of search theory were firstly made on military operations [22]. Koopman [12] stated that the principles of search theory could be applied effectively to any situation where the objective is to find a person or object contained in some restricted geographic area. After military applications, it was also applied to different problems such as; surveillance, explorations, medicine, industry and search and rescue operations [9]. The aim of searching in the context of Aeronautical Search and Rescue (ASAR), for instance, is to find the missing aircraft effectively and as quickly as possible with the available resources [21].

Stone [21] summarized major steps in the development of search theory, for instance, stationary target problems, moving target problems, optimal searcher path algorithms, and dynamic search games. The last item (search games) is the primary focus of recent researches, including numerous sub-domains such as mobile evaders, avoiding target, ambush

games, inspection games, and tactical games.

A search theory problem, in general, is characterized by three pieces of data ([3]):

- the probabilities of the searched object (the "target") being in various possible locations;
- the local detection probability that a particular amount of local search effort could detect the target;
- the total amount of searching effort available.

The problem is to find the optimal distribution of this total effort that maximizes the probability of detection.

In this paper we consider an instance of search theory problems that can be described as follows: suppose that a space of search is partitioned into zones of reasonable size. A unique sensor must be able to explore efficiently a whole zone. Each zone is itself partitioned into cells. A cell is an area in which every points have the same properties, according to the difficulty of detection (altitude, vegetation, etc.). Each sensor has its own coefficient of visibility over a cell. The visibility coefficients depend also on the kind of target that is searched. Here, there is a unique target to detect. This study considers a multi period search of a moving target. This means that information about the sensors and the target will be now indexed by time (the period index). The target prior is now trajectorial and we shall consider here a Markovian (target) prior. Furthermore, assuming that sensors act independently at the cell level. The target is said undetected for this multiperiod search if it has not been detected at any period of the search. The objective is alloting sensors to search zones and finding the search resources sharing of multisensor in multizone at each time period so as to mimimize the probability of non-detection of a target.

This problem, in general, is very complicated because of the huge number of possible target trajectories. For a unique sensor, the problem has been theoretically solved in [20, 23]; while extensions to double layered constraints have been considered in [8]. In practice, all feasible algorithms are based on a forward-backward split introduced by Brown [2]. Similar procedures are also much employed in order to estimate Hidden Markov Models parameters (see e.g. [5]). In our case, although the forward-backward split technique allows us to simplify the main problem, the obtained general subproblem is still very hard since it is hierarchical:

- At upper level: finding the best allotment of sensors to search zones (a sensor is allotted to a unique zone);
- At lower level: determining the best resource sharing for every sensor, in order to have an optimal surveillance over the allotted zone.

In [19], Simonin et al. have proposed a hierarchical approach for solving the general subproblem, where a cross-entropy (CE) algorithm [1, 4, 18] has been developed for the

upper level while an optimization method based on the algorithm of de Guenin [7] for detecting a stationary target has been used in the lower level. Besides this paper, we do not find in the literature the works considering the general subproblem. *We can say that, up to now, there is no deterministic models and methods for it.*

In [15], we efficiently applied a new method for solving a problem, which has the same structure as the general subproblem. In this method, we developed a *deterministic continuous optimization approach* based on DC (Difference of Convex functions) programming and DCA (DC optimization Algorithms). Specifically, we proposed a new optimization model that is a nonlinear mixed 0-1 programming problem. This problem was then reformulated as a DC (Difference of Convex functions) program via a penalty technique. DC programming and DCA (DC algorithm) ([14, 16, 17]) have been investigated for solving the resulting DC program. This motivates us to investigate the combination of the forward-backward split technique and our proposed method for solving the considered problem.

The paper is organized as follows. In Section 2, the problem statement and the classic "forward-backward split" are presented. In Section 3, we present our approach developed in [15] for solving the general subproblem, then introduce a schema combining the forward-backward split and DCA. The numerical results are reported in Section 4 while some conclusions and perspectives are discussed in Section 5.

# 2 Problem statement

First, let us introduce the notations employed in the remainder of the paper.
The search periods are indexed by $t \in \{1, 2, ..., T\}$
$E$ : space of search, $Z$ : number of zones, $S$ : number of sensors
$z$ : zone index
$i$ : cell index
$s$ : sensor index
$\alpha$ : prior on the initial location of the target
$\varphi_s^t(c_{i,z})$ : quantity of resource of sensor $s$ allotted to cell $i$ of the zone $z$ at the time $t$
$\Phi_s^t$ : quantity of resource available for sensor $s$ to search the space at the time $t$
$w_{i,z,s}$ : coefficient that characterizes the acuity of sensor $s$ over cell $i$ of the zone $z$ (visibility coefficient)

We report below the problem statement described in [19] (see [19] for more details).
*The space of search*: the search space, named $E$, is a large space with spatially variable search characteristics. The search space $E$ is divided into $Z$ search zones, denoted $E_z, z = 1, 2, ..., Z$, each of them is partitioned into $C_z$ cells, denoted $\{c_{i,z}\}_{i=1}^{C_z}$ so that:

$$E = \bigcup_{z=1}^{Z} E_z, E_z \cap Z_{z'} = \emptyset, \forall z \neq z',$$

$$E_z = \bigcup_{i=1}^{C_z} c_{i,z}, c_{i,z} \cap c_{j,z}, \forall i \neq j.$$

A cell $c_{i,z}$ represents the smallest search area in which the search parameters are constant. For example, it can be a part of land with constant characteristics (latitude, landscape). Each zone must have a reasonable size in order to be explored by a sensor within a fixed time interval.

*The target*: the target is hidden in one unit of the search space. Its location is characterized by a prior $\alpha_{i,z}$. Thus, we have

$$\sum_{z=1}^{Z} \sum_{i=1}^{C_z} \alpha_{i,z} = 1.$$

*The means of search*: means of search can be passive (e.g. IRST, ESM) or active sensors (radars). We will consider that searching the target will be carried out by $S$ sensors. Due to operational constraints, each sensor $s \in S$ must be allotted to a unique search zone. For example, it could be the exploration time to share between units of a zone. At the lower level the amount of search resource allocated to the cell $c_{i,z}$ for the sensor $s$ at the time $t$ -if sensor $s$ is allotted to zone $E_z$ -is denoted $\varphi_s^t(c_{i,z})$. It can represent the time spent on searching the cell $c_{i,z}$ (passive sensor), the intensity of emissions or the number of pulses (active sensors), etc. Furthermore, each sensor $s$ has at the time $t$ a search amount $\Phi_s^t$, it means that if sensor $s$ is allotted to the zone $E_z$, we have the constraint:

$$\sum_{i=1}^{C_z} \varphi_s^t(c_{i,z}) \leq \Phi_s^t.$$

To characterize the effectiveness of the search at the cell level, we consider the conditional non-detection probability $\bar{P}_s(\varphi_s^t(c_{i,z}))$ which represents the probability of not detecting the target given that the target is hidden in $c_{i,z}$ and that we apply an elementary search effort $\varphi_s^t(c_{i,z})$ on $c_{i,z}$. Some hypotheses are made to model $\bar{P}_s(\varphi_s^t(c_{i,z}))$. For all sensors, $\varphi_s^t(c_{i,z}) \mapsto \bar{P}_s(\varphi_s^t(c_{i,z}))$ is convex and non-increasing (law of diminishing return). Assuming independence of elementary detections, a usual model is $\bar{P}_s(\varphi_s^t(c_{i,z})) = \exp(-w_{i,z,s}\varphi_s^t(c_{i,z}))$, where $w_{i,z,s}$ is a (visibility) coefficient which characterizes the reward for the search effort put in $c_{i,z}$ by sensor $s$.

An additional assumption is that sensors act independently at the cell level which means that at the time period $t$ if $S$ sensors are allotted to $c_{i,z}$ the probability of not detecting a target hidden in $c_{i,z}$ is simply $\prod_{s=1}^{S} \bar{P}_s(\varphi_s^t(c_{i,z}))$.

At each time period $t$, let $m_t : \{1, 2, ..., S\} \to \{1, 2, ..., Z\}$ be a mapping allotting sensors to search zones. Our aim is to find both the optimal mappings $m_t$ and the optimal local distributions $\varphi_s^t$ in order to minimize the non-detection probability, i.e.,

$$F((m_t,\varphi_s^t)_{t=1}^T) = \sum_{\overrightarrow{\omega}\in\Omega} \alpha(\overrightarrow{\omega}) \prod_{t=1}^{T} \prod_{s\in m_t^{-1}(z)} \bar{P}_s\left(\varphi_s^t(\overrightarrow{\omega}(t))\right),$$

where $\Omega$ denotes the set of target trajectories, $\overrightarrow{\omega}$ a target trajectory in $\Omega$, and $\overrightarrow{\omega}(t)$ is the cell of the target trajectory $\overrightarrow{\omega}$ at the time $t$. That leads to solve the following constrained problem [19]:

$$\begin{cases} \min\limits_{(m_t,\varphi_s^t)_{t=1}^T} F((m_t,\varphi_s^t)_{t=1}^T) \\ \text{s.t.} \quad \forall t,\forall z,\forall s\in m^{-1}(z), \sum\limits_{i=1}^{C_z} \varphi_s^t(c_{i,z}) \leq \Phi_s, \\ \qquad \forall i\in z, \varphi_s^t(c_{i,z}) \geq 0, \\ \qquad \forall t, m_t \text{ mapping} : \{1,2,...,S\} \to \{1,2,...,Z\}. \end{cases} \tag{1}$$

**Forward-backward split**

We can rewrite the objective function $F$ as follows:

$$F((m_t,\varphi_s^t)_{t=1}^T) = \sum_{z=1}^{Z}\sum_{i=1}^{C_z} \beta_{i,z}^{\tau} \prod_{s\in m_\tau^{-1}(z)} \bar{P}_s(\varphi_s^\tau(c_{i,z})),$$

where

$$\beta_{i,z}^{\tau} = \sum_{\overrightarrow{\omega}\in\overrightarrow{\omega}_{i,z,\tau}} \alpha(\overrightarrow{\omega}) \prod_{1\leq t\leq T}^{t\neq\tau} \prod_{s\in m_t^{-1}(z)} \bar{P}_s\left(\varphi_s^t(c_{i_t,z_t})\right),$$

$$\overrightarrow{\omega}_{i,z,\tau} = \{\overrightarrow{\omega}\in\Omega : \overrightarrow{\omega}(\tau) = c_{i,z}\},$$

$$\overrightarrow{\omega} = (c_{i_1,z_1},...,c_{i_\tau,z_\tau},...,c_{i_T,z_T}),$$

$$\alpha(\overrightarrow{\omega}) = \alpha_{i_1,z_1} \prod_{t}^{T-1} \alpha_{t,t+1}(c_{i_t,z_t},c_{i_{t+1},z_{t+1}}).$$

Here, $\alpha_{t,t+1}(c_{i_t,z_t},c_{i_{t+1},z_{t+1}})$ is probability the target move from cell $c_{i_t,z_t}$ to the cell $c_{i_{t+1},z_{t+1}}$.

It remains to have a mean to calculate efficiently the $\beta_{i,z}^{\tau}$. To that aim, the trajectory Markov hypothesis is instrumental and we consider the following splitting of the $\beta_{i,z}^{\tau}$:

$$\beta_{i,z}^{\tau} = U_{i,z}^{\tau}.D_{i,z}^{\tau},$$

where $U$ and $D$ are recursively defined by

$$U^\tau(i,z) = \sum_{j\in\widetilde{z}} \alpha_{\tau-1,\tau}(j,i) \prod_{s\in m_{\tau-1}^{-1}(z)} \bar{P}_s\left(\varphi_s^{\tau-1}(c_{j,\widetilde{z}})\right) U^{\tau-1}(j,\widetilde{z}),$$

$$D^\tau(i,z) = \sum_{j\in\widetilde{z}} \alpha_{\tau,\tau+1}(j,i) \prod_{s\in m_{\tau+1}^{-1}(z)} \bar{P}_s\left(\varphi_s^{\tau+1}(c_{j,\widetilde{z}})\right) D^{\tau+1}(j,\widetilde{z}).$$

In the above equation, we denote by $\widetilde{z}$, the zones which can be attained conditionally to the hypothesis that the target is in the cell $i$ of the zone $z$ at the period $\tau$ and that it has a Markovian prior $\alpha$. Such a forward-backward split was introduced by Brown [2].

Now, for a given $\tau$, and considering that the $\beta_{i,z}^{\tau}$ are known, the multiperiod search problem is put in the situation: the target is "static" with prior $\beta_{i,z}^{\tau}$. The general subproblem can be written as follows:

$$\begin{cases} \min\limits_{\{m,\varphi_s(c_{i,z})\}} \sum\limits_{z=1}^{Z}\sum\limits_{i=1}^{C_z} \beta_{i,z} \prod\limits_{s\in m^{-1}(z)} \bar{P}_s(\varphi_s(c_{i,z})) \\ \text{s.t.} \quad \forall z, \forall s \in m^{-1}(z), \sum\limits_{i=1}^{C_z} \varphi_s(c_{i,z}) \leq \Phi_s, \\ \qquad \forall i \in z, \varphi_s(c_{i,z}) \geq 0, \\ \qquad m \text{ mapping} : \{1,2,...,S\} \to \{1,2,...,Z\}. \end{cases} \tag{2}$$

# 3 Solution Method

## 3.1 A deterministic formulation of Problem (2)

Let us introduce the allocation variable $u_{z,s}$ defined by

$$u_{z,s} = \begin{cases} 1 & \text{if the sensor } s \text{ is allotted to the zone } z, \\ 0 & \text{otherwise.} \end{cases}$$

Let variable $x_{i,z,s} = \varphi_s(c_{i,z})$ be the quantity of resource of the sensor $s$ allotted to the cell $c_{i,z}$ in the zone $z$. We can rewrite (2) in the following form:

$$\begin{cases} \min\limits_{x,u}(f(x,u) = \sum\limits_{z=1}^{Z}\sum\limits_{i=1}^{C_z} \beta_{i,z} \prod\limits_{s=1}^{S} \bar{P}_s(x_{i,z,s}u_{z,s})) \\ \text{s.t.} \sum\limits_{i=1}^{C_z} x_{i,z,s} \leq \Phi_s, z = 1,...,Z, s = 1,...,S, \\ \quad \sum\limits_{z=1}^{Z} u_{z,s} = 1, s = 1,...,S, \\ \quad u_{z,s} \in \{0,1\}, x_{i,z,s} \geq 0, s = 1,...,S, z = 1,...,Z, i = 1,...,C_z. \end{cases} \tag{3}$$

The constraints $\sum\limits_{z=1}^{Z} u_{z,s} = 1, \forall s = 1,2,...,S$ mean that each sensor $s \in S$ must be allotted to a unique search zone.
With the usual model $\bar{P}_s$ [11]:

$$\bar{P}_s(x_{i,z,s}) = \exp(-w_{i,z,s}x_{i,z,s}),$$

the objective function $f$ becomes:

$$f(x,u) = \sum_{z=1}^{Z}\sum_{i=1}^{C_z} \beta_{i,z} \exp(-\sum_{s=1}^{S} w_{i,z,s}x_{i,z,s}u_{z,s}).$$

Set

$$d = S.(C_1 + C_2 + ... + C_z), n = Z.S.$$

Denoting by

$$D = \{x = (x_{i,z,s}) \in \mathbb{R}^d_+ : \sum_{i=1}^{C_z} x_{i,z,s} \leq \Phi_s, z = 1, ..., Z, s = 1, ..., S, \}$$

$$M = \{u = (u_{z,s}) \in \{0,1\}^{Z.S} : \sum_{z=1}^{Z} u_{z,s} = 1, s = 1, ..., S\},$$

we can writte the problem (3) in the form:

$$(P) \begin{cases} \min\limits_{x,u} \sum\limits_{z=1}^{Z} \sum\limits_{i=1}^{C_z} \beta_{i,z} \exp(-\sum\limits_{s=1}^{S} w_{i,z,s} x_{i,z,s} u_{z,s}) \\ \text{s.t. } x \in D, u \in M, \end{cases}$$

which is a nonlinear mixed 0-1 programming problem. It is easy to see that the objective function of $(P)$, say $f$, is convex in $x$ for each fixed $u$, and similarly, it is convex in $u$ for each fixed $x$. Moreover, $f$ is infinitely differentiable.

## 3.2 DC Reformulation

Consider the function $p$ and the bounded polyhedral convex set $K$ defined, respectively, by:

$$p(u) = \sum_{z=1}^{Z} \sum_{s=1}^{S} u_{z,s}(1 - u_{z,s}),$$

and

$$K = \{u = (u_{z,s}) \in [0,1]^{Z.S} : \sum_{z=1}^{Z} u_{z,s} = 1, s = 1, ..., S\}.$$

We notice that $p$ is finite and concave on $\mathbb{R}^{Z.S}$, non-negative on $K$ and

$$M = \{u \in K : p(u) \leq 0\}.$$

Hence Problem $(P)$ can be rewritten as

$$\alpha = \min\{f(x,u) : x \in D,\ u \in K,\ p(u) \leq 0\}.$$

The exact penalty result is given in the following theorem.

**Theorem 1** *(i) Let*

$$t^0 = \max\{||\nabla^2_u(f(x,u))|| : \quad u \in [0,1]^n, x \in D\},$$

*then* $\forall t > t^0$ *the problem (P)*

$$\min\{f(x,u) : x \in D, u \in M\} \quad (P)$$

*is equivalent to the next problem*

$$\min\{f(x,u) + tp(u) : x \in D, u \in K\}. \quad (P_t)$$

*(ii) If* $(x^*, u^*)$ *is a local solution to problem* $(P_t)$ *then* $(x^*, u^*)$ *is a feasible solution to the problem* $(P)$*.*

**Proof** see [15].

Note that the part (ii) in this theorem is very useful for DCA applied to $(P_t)$ because DCA usually produces a local solution.

Since $D$ is compact and $f \in C^\infty$, $\max\{||\nabla_u^2(f(x,u))|| : \quad u \in [0,1]^n, x \in D\}$ exists. The following proposition gives an estimation of $t^0$ :

**Proposition 2** *We have*

$$t^0 := \max\{||\nabla_u^2(f(x,u))|| : \quad u \in [0,1]^n, x \in D\} \leq \Phi\beta w^2 \sum_{s=1}^{S} \Phi_s,$$

*where*

$$\beta = \max\{\beta_{i,z} : z = 1, ..., Z, i = 1, ..., C_z\}, \Phi = \max\{\Phi_s : 1, ..., S\},$$
$$w = \max\{w_{i,z,s} : z = 1, ..., Z, i = 1, ..., C_z, s = 1, ..., S\}.$$

**Proof** see [15].

In the sequel we will investigate DCA for solving problem $(P_t)$ with $t^0 = \Phi\beta w^2 \sum_{s=1}^{S} \Phi_s$.

## 3.3 DC Algorithm (DCA)

### 3.3.1 Outline of DC Programming and DCA

DC programming and DCA, which constitute the backbone of smooth/nonsmooth nonconvex programming and global optimization, have been introduced by Pham Dinh Tao in 1985 and extensively developed by Le Thi Hoai An and Pham Dinh Tao since 1994 to become now classic and increasingly popular ([14, 16, 17] and references therein). They address the problem of minimizing a function $f$ which is a difference of convex functions on the whole space $\mathbb{R}^p$ or on a convex set $C \subset \mathbb{R}^p$. Generally speaking, a DC program takes the form

$$\alpha = \inf\{f(x) := g(x) - h(x) : x \in \mathbb{R}^p\} \quad (P_{dc}) \tag{4}$$

where $g, h$ are lower semicontinuous proper convex functions on $\mathbb{R}^p$. Such a function $f$ is called DC function, and $g - h$, DC decomposition of $f$ while $g$ and $h$ are DC components of $f$. The convex constraint $x \in C$ can be incorporated in the objective function of $(P_{dc})$ by using the indicator function on $C$ denoted $\chi_C$ which is defined by

$$\chi_C(x) = \begin{cases} 0 \text{ if } x \in C \\ +\infty, \text{ otherwise.} \end{cases}$$

If either $g$ or $h$ is polyhedral convex function then $(P_{dc})$ is called a polyhedral DC program. Let

$$g^*(y) := \sup\{\langle x, y\rangle - g(x) : x \in \mathbb{R}^p\}$$

be the conjugate function of $g$. Then, the following program is called the dual program of $(P_{dc})$:

$$\alpha_D = \inf\{h^*(y) - g^*(y) : y \in \mathbb{R}^p\}. \quad (D_{dc}) \tag{5}$$

One can prove that $\alpha = \alpha_D$, (see e.g. [14, 16]) and there is the perfect symmetry between primal and dual DC programs: the dual to $(D_{dc})$ is exactly $(P_{dc})$.
DCA is based on the local optimality conditions of $(P_{dc})$, namely

$$\partial h(x^*) \cap \partial g(x^*) \neq \emptyset \tag{6}$$

(such a point $x^*$ is called *critical point* of $g - h$), and

$$\emptyset \neq \partial h(x^*) \subset \partial g(x^*). \tag{7}$$

The condition (7) is necessary local optimality of $(P_{dc})$ (or the generalized KKT condition). It is also sufficient for many classes of DC programs. In particular it is sufficient for the next cases quite often encountered in practice (see e.g. [14, 16]):
The idea of DCA is simple: each iteration of DCA approximates the concave part $-h$ by its affine majorization (that corresponds to taking $y^k \in \partial h(x^k)$) and minimizes the resulting convex function (that is equivalent to determining $x^{k+1} \in \partial g^*(y^k)$).
**Generic DCA scheme**
**Initialization:** Let $x^0 \in \mathbb{R}^p$ be a best guess, $0 \leftarrow k$.
**Repeat**
Calculate $y^k \in \partial h(x^k)$
Calculate $x^{k+1} \in \arg\min\{g(x) - h(x^k) - \langle x - x^k, y^k\rangle : x \in \mathbb{R}^p\} \quad (P_k)$
$k + 1 \leftarrow k$
**Until** convergence of $x^k$.
It is important to mention the following main convergence properties of DCA:

- DCA is a descent method (the sequences $\{g(x^k) - h(x^k)\}$ and $\{h^*(y^k) - g^*(y^k)\}$ are decreasing) *without linesearch*;
- If the optimal value $\alpha$ of the problem $(P_{dc})$ is finite and the infinite sequences $\{x^k\}$ and $\{y^k\}$ are bounded then every limit point $x^*$ (resp. $y^*$) of the sequence $\{x^k\}$ (resp. $\{y^k\}$) is a critical point of $g - h$ (resp. $h^* - g^*$).

- DCA has a *linear convergence* for general DC programs.
- DCA has a finite convergence for polyhedral DC programs.

Moreover, it is worth noting that the construction of DCA involves DC components $g$ and $h$ but not the function $f$ itself. Hence, for a DC program, each DC decomposition corresponds to a different version of DCA. Since a DC function $f$ has an infinite number of DC decompositions which have crucial impacts on the qualities (speed of convergence, robustness, efficiency, globality of computed solutions,...) of DCA, the search for a "good" DC decomposition is important from algorithmic point of views. Furthemover, despite its local character, DCA with a good initial point can converge to global solutions. Finding a "good" initial point is then also an important stage of DCA. How to develop an efficient algorithm based on the generic DCA scheme for a practical problem is thus a judicious question to be studied, and the answer depends on the specific structure of the problem being considered. The solution of a practical nonconvex program by DCA must be composed of two stages: the search of an *appropriate* DC decomposition and the search of a *good* initial point. An appropriate DC decomposition, in our sense, is the one that corresponds to a DCA which is not expensive and has interesting convergence properties.
In the next subsection we will develop an instance of DCA to solve the problem $(P_t)$, and study the convergence properties of the proposed algorithm.

### 3.3.2 Description of the DCA applied to $(P_t)$

From the computations, we have

$$||H(f)||_\infty \leq \max\{S\beta w^2 + S\beta w^2\Phi + \beta w, 2\Phi\beta w^2 \sum_{s=1}^{S} \Phi_s + Nw\beta\}.$$

where

$$N = \max\{C_z : z = 1, ..., Z\}.$$

Hence, let $\rho := \max\{S\beta w^2 + S\beta w^2\Phi + \beta w, 2\Phi\beta w^2 \sum\limits_{s=1}^{S} \Phi_s + Nw\beta\}$, a DC formulation of $(P_t)$ can be

$$\min\left\{g(x,u) - h(x,u) : (x,u) \in D \times K\right\}, \tag{8}$$

where

$$g(x,u) := \frac{\rho}{2}||(x,u)||^2 \text{ and } h(x,u) := \frac{\rho}{2}||(x,u)||^2 - f(x,u) - tp(u).$$

DCA applied to DC program (8) consists of computing, at each iteration $k$, the two sequences $\left\{(y^k, v^k)\right\}$ and $\left\{(x^k, u^k)\right\}$ such that $(y^k, v^k) \in \partial h(x^k, u^k)$ and $(x^{k+1}, y^{k+1})$ is an optimal solution of the next convex quadratic program

$$\min\left\{\frac{\rho}{2}||(x,u)||^2 - \langle(x,u),(y^k,v^k)\rangle : (x,u) \in D \times K\right\}$$

which can be decomposed into two subproblems

$$\min\left\{\frac{\rho}{2}||x||^2 - \langle x, y^k\rangle : x \in D\right\} \tag{9}$$

and

$$\min\left\{\frac{\rho}{2}||u||^2 - \langle u, v^k\rangle : u \in K\right\}. \tag{10}$$

We are now in a position to summarize the DCA for solving Problem $(P_t)$

**Algorithm 1:**
**Step 1.** Initialization: let $(x^0, u^0)$ satisfy the constraints of the problem. Choose $\epsilon_1 > 0, \epsilon_2 > 0$ and $k = 0$.

**Step 2.** Compute $(y^k, v^k) = \nabla h(x^k, u^k)$, with

$$y^k = \rho x^k - \nabla_x f(x^k, u^k),\ v^k = \rho u^k - \nabla_u f(x^k, u^k) + t(2u^k - e).$$

**Step 3.** Compute $(x^{k+1}, u^{k+1})$ by solving the two convex quadratic problems (9) and (10)
**Step 4.** Iterate Step 2 and 3 until

$$|f(x^{k+1}, u^{k+1}) - f(x^k, u^k)| \leq \epsilon_1(1 + |f(x^{k+1}, u^{k+1})|)$$

$$\text{or}\ \ ||(x^{k+1}, u^{k+1}) - (x^k, u^k)||_\infty \leq \epsilon_2(1 + ||(x^{k+1}, u^{k+1})||_\infty).$$

**Theorem 3** *(Convergence properties of Algorithm DCA, for simplicity's sake, we omit here the dual part of these properties)*
*i) DCA generates the sequence* $\left\{(x^k, u^k)\right\}$ *such that the sequence* $\left\{f(x^k, u^k)\right\}$ *is decreasing.*
*ii) The point* $\left\{(x^k, u^k)\right\}$ *verifies the generalized condition of the problem (8).*

**Proof** Direct consequences of the convergence properties of DCA for general DC programs and the fact that the function $h$ is differentiable.

In order to speed up DCA, we proposed an efficient way for computing $(x^{k+1}, u^{k+1})$ in [15].

## 3.4 Combining forward-backward split technique and DCA (FAB&DCA)

The multisensor multizone moving target algorithm takes the following form:
**Algorithm 3:**

1. **Initialization:**

   $\forall \tau, \forall z, \forall i, D_1^\tau(i, z) = 1,$
   $\forall k, \forall z, \forall i, U_1^k(i, z) = \alpha_{i,z}.$

2. **Iteration** (k index):

**Iteration**($\tau$ index)

- $\forall z, \forall i$, compute the optimal allotment and resource sharing by DCA with prior

$$\beta_{i,z}^{\tau} = U_k^{\tau}(i,z).D_k^{\tau}(i,z);$$

- $\forall z, \forall i$, compute $U_k^{\tau+1}(i,z)$;

$\forall z, \forall i$, compute $D_{k+1}(\tau)(i,z)$;

**Stop**: when the search plan is no more improved.

# 4 Numerical result

Suppose that the search space is the lake of Laouzas in France [19]. Number of zone is $Z = 4$ and there are $n = 30$ cells in each zone. We assume that the target is Markovian and moves south east direction. The transition matrix describing the target motion is given in Fig. 3 and is assumed to be constant over time. The search is carried out over four time periods by means of the six sensors. All sensors has the same amount of resource $\Phi$ for all time period. The coefficients are given in Table 1. We take five search plans.

The program is written by language C on Microsoft Visual C++ 2008, and the implementation takes place on a notebook with chipset Intel(R) Core(TM) Duo CPU 2.0 GHz, RAM 3GB.

To find a starting point of DCA, we firstly apply DCA on $(P_t)$ with the parameters $\rho$ and *penalty* quite small (depending on the parameters $\Phi$) form the starting point $(x^0, u^0)$, where $x^0 = (x_{i,z,s}^0)$ and $u^0 = (u_{z,s}^0)$ is determined as follows:

$$x_{i,z,s}^0 = \Phi/n, \ \forall i, z, s; \ u_{z,s}^0 = 1/Z, \ \forall z, s.$$

Then we get a new pair $(x^1, u^1)$ which is chosen as the starting point of the main algorithm. In two stages of DCA, we choose $\epsilon_1 = \epsilon_2 = 10^{-7}$.

In Table 2, we compare the result obtained by FAB&DCA and the method of combining the forward-backward split technique and the Cross-Entropy method (FAB&CE)[19]. We choose $\Phi = 5$. The parameters for CE as follows: for each iteration, number of samples is $N = 50$, $\theta = 0.3$ and the number of iteration is limited to 15. We observe that the FAB&DCA produced the better solutions than the FAB&CE while the CPU is shorter. Moreover, although we only care of the results at the time 4 of the last search plan

(iteration $k = 5$), FAB&DCA works better than FAB&CE does in all time of this search plan.

Table 3 presents additional results with $\Phi = 10$ and $\Phi = 15$. FAB&DCA produced once again the better solutions than the FAB&CE while CPU time is shorter. Here, there is a negative relationship between $\Phi$ and the non-detection probability. When $\Phi = 15$, the probability of non-detection obtained by FAB&DCA is 0.078225.

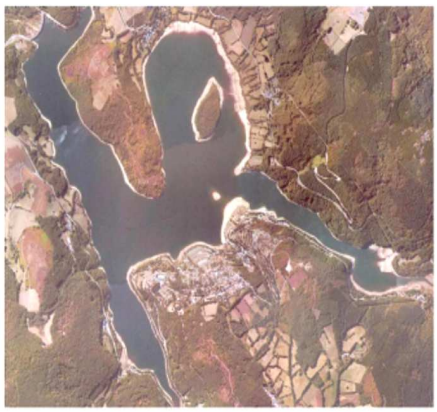

Figure 1: An aerial photograph of the lake of Laouzas

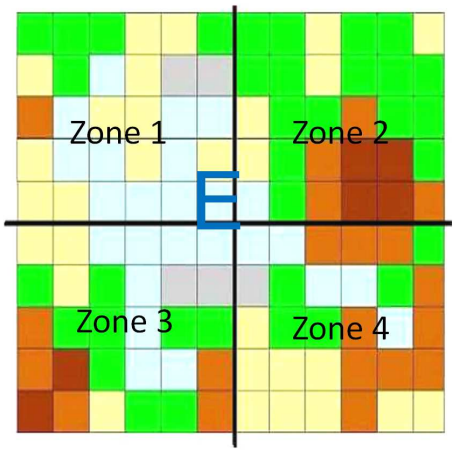


Figure 2: Partition of the lake Laouzas

# 5 Conclusion

We have presented a new approach for solving the problem planning a multisensor multizone search for a moving target. Actually, this is an extension of our developed approach in the case "static" target. In other words, that is combination of the forward-backward split

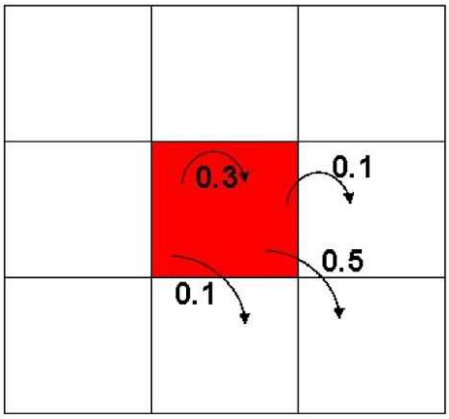


Figure 3: The target transition probability

| Type of Cell | Prior of target | Visibility of sensors | | | | | |
|---|---|---|---|---|---|---|---|
| | | Sensor 1 | Sensor 2 | Sensor 3 | Sensor 4 | Sensor 5 | Sensor 6 |
| Forest | 0.0085 | 0.4 | 0.5 | 0.6 | 0.8 | 0.5 | 0.1 |
| Water | 0.001 | 0.9 | 0.1 | 0.1 | 0.1 | 0.3 | 0.5 |
| Plat Plan | 0.0115 | 0.3 | 0.1 | 0.4 | 0.6 | 0.5 | 0.2 |
| Rough plan | 0.013 | 0.2 | 0.7 | 0.8 | 0.2 | 0.4 | 0.6 |
| Very rough plan | 0.014 | 0.1 | 0.6 | 0.7 | 0.1 | 0.3 | 0.5 |
| Town | 0 | 0.8 | 0.9 | 0.1 | 0.7 | 0.6 | 0.2 |

Table 1: Parameters

| Time | Sensors | FAB&CE method | | | | | | FAB&DCA method | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Allotment S = 6 | | | | Non-detection probability | CPU time | Allotment S = 6 | | | | Non-detection probability | CPU time |
| Time 1 | Sensor 1 | 0 | 1 | 0 | 0 | 0.468681 | 18.57s | 0 | 1 | 0 | 0 | 0.440446 | 9.36s |
| | Sensor 2 | 1 | 0 | 0 | 0 | | | 0 | 0 | 1 | 0 | | |
| | Sensor 3 | 0 | 0 | 1 | 0 | | | 0 | 0 | 1 | 0 | | |
| | Sensor 4 | 1 | 0 | 0 | 0 | | | 1 | 0 | 0 | 0 | | |
| | Sensor 5 | 1 | 0 | 0 | 0 | | | 1 | 0 | 0 | 0 | | |
| | Sensor 6 | 0 | 0 | 0 | 1 | | | 0 | 0 | 0 | 1 | | |
| Time 2 | Sensor 1 | 1 | 0 | 0 | 0 | 0.386112 | | 1 | 0 | 0 | 0 | 0.367013 | |
| | Sensor 2 | 0 | 1 | 0 | 0 | | | 0 | 1 | 0 | 0 | | |
| | Sensor 3 | 0 | 0 | 1 | 0 | | | 0 | 1 | 0 | 0 | | |
| | Sensor 4 | 0 | 1 | 0 | 0 | | | 0 | 1 | 0 | 0 | | |
| | Sensor 5 | 0 | 1 | 0 | 0 | | | 0 | 0 | 0 | 1 | | |
| | Sensor 6 | 0 | 1 | 0 | 0 | | | 0 | 0 | 0 | 1 | | |
| Time 3 | Sensor 1 | 0 | 0 | 0 | 1 | 0.326479 | | 1 | 0 | 0 | 0 | 0.310935 | |
| | Sensor 2 | 0 | 1 | 0 | 0 | | | 0 | 1 | 0 | 0 | | |
| | Sensor 3 | 0 | 0 | 1 | 0 | | | 0 | 1 | 0 | 0 | | |
| | Sensor 4 | 0 | 1 | 0 | 0 | | | 0 | 0 | 1 | 0 | | |
| | Sensor 5 | 0 | 0 | 0 | 1 | | | 0 | 0 | 0 | 1 | | |
| | Sensor 6 | 0 | 0 | 0 | 1 | | | 0 | 0 | 0 | 1 | | |
| Time 4 | Sensor 1 | 0 | 0 | 1 | 0 | **0.272548** | | 0 | 0 | 1 | 0 | **0.270107** | |
| | Sensor 2 | 0 | 0 | 0 | 1 | | | 0 | 0 | 0 | 1 | | |
| | Sensor 3 | 0 | 0 | 0 | 1 | | | 0 | 0 | 0 | 1 | | |
| | Sensor 4 | 0 | 0 | 1 | 0 | | | 0 | 0 | 1 | 0 | | |
| | Sensor 5 | 0 | 0 | 0 | 1 | | | 0 | 0 | 0 | 1 | | |
| | Sensor 6 | 0 | 0 | 0 | 1 | | | 0 | 0 | 0 | 1 | | |

Table 2: Non-detection probability with $\Phi = 5$.

| Amount of resource Φ | Time | FAB&CE method | | FAB&DCA method | |
|---|---|---|---|---|---|
| | | Non-detection probability | CPU time | Non-detection probability | CPU time |
| 5 | Time 1 | 0.468681 | 18.57s | 0.440446 | 9.36s |
| | Time 2 | 0.386112 | | 0.367013 | |
| | Time 3 | 0.326479 | | 0.310935 | |
| | Time 4 | **0.272548** | | **0.270107** | |
| 10 | Time 1 | 0.254927 | 18.43s | 0.228051 | 9.00s |
| | Time 2 | 0.205480 | | 0.191276 | |
| | Time 3 | 0.173844 | | 0.160008 | |
| | Time 4 | **0.149615** | | **0.137433** | |
| 15 | Time 1 | 0.139648 | 18.41s | 0.134255 | 14.17s |
| | Time 2 | 0.113025 | | 0.111196 | |
| | Time 3 | 0.098799 | | 0.091861 | |
| | Time 4 | **0.090995** | | **0.078225** | |

Table 3: Relation of non-detection probability and amount of resource.

technique and DCA. The table 2 and 3 show that our approach can solve this problem more effectively than the FAB&CE method, an efficient algorithm for the considered problem.

# References


[1] P.-T. de Boer, D. P. Kroese, S. Mannor, and R.Y. Rubinstein, A Tutorial on The Cross-Entropy Method, Annals of Operations Research 134, 19-67, 2005.

[2] S.S. Brown, Optimal search for a moving target in discrete time and space, Operations Research 32 (5), pp. 1107-1115, 1979.

[3] J.P. Cadre, and G. Souris, Searching Tracks, IEEE Transactions on Aerospace and Electronic systems, Vol. 36, No.4, 1149-1166, 2000.

[4] A. Costa, O.D. Jones, and D. Kroese, Convergence properties of the cross-entropy method for discrete optimization, Operations Research Letters 35(5): 573-580, 2007.

[5] Y. Ephraim and N. Merhav, Hidden Markov processes, IEEE Transactionson Information Theory 48(2002), pp.1518-1569

[6] J.R. Frost (1999c). Principles of search theory, part III: Probability density distributions, Response, 17(3), pp. 1-10.

[7] J. de Guenin, Optimum distribution of effort: an extension of the Koopman theory, Operations Research 9 (1), pp. 1-7, 1961.

[8] R. Hohzaki and K. Iida, A concave minimization problem with double layers of constraints on the total amount of resources, Journal of the Operations Research Society of Japan 43(1)(2000), pp.109-127.

[9] K.B. Haley, and L.D. Stone (Eds.), Search Theory and Applications, New York: Plenum Press, 1980.

[10] J. Júdice, M. Raydan, et al., On the solution of the symmetric eigenvalue complementarity problem by the spectral projected gradient algorithm. Numerical Algorithms 47(4): 391-407, 2008.

[11] B.O. Koopman, Search and its optimization, Mathematical Monthly 7, pp. 527-540, 1979.

[12] B.O. Koopman, Search and Screening: General Principles with Historical Applications, Pergamon Press, New York, 1980.

[13] B.O. Koopman, Search and Screening: General Principle with Historical Applications. Alexandria, VA: MORS Heritage Series, 1999.

[14] H.A. Le Thi, and T. Pham Dinh, The DC (difference of convex functions) Programming and DCA revisited with DC models of real world nonconvex optimization problems, Annals of Operations Research, Vol 133, pp. 23-46, 2005.

[15] H.A. Le Thi, D.M. Nguyen, and T. Pham Dinh, A deterministic optimization approach for planning a multisensor multizone search for a target, submitted in Computers & Operations Research (2011).

[16] T. Pham Dinh, and H.A. Le Thi, Convex analysis approach to DC programming: Theory, Algorithms and Applications (dedicated to Professor Hoang Tuy on the occasion of his 70th birthday), Acta Mathematica Vietnamica, 22, pp. 289-355, 1997.

[17] T. Pham Dinh and H.A. Le Thi, DC optimization algorihms for solving the trust region subproblem, SIAM J.Optimization **8** (1998), 476-505.

[18] R.Y. Rubinstein, and D. Kroese, The cross-entropy method: a unified approach to combinatorial optimization, Monté Carlo simulation, and machine learning, Berlin: Springer, 2004.

[19] C. Simonin, J.P. Le Cadre, and F. Dambreville, A hierarchical approach for planning a multisensor multizone search for a moving target, Computers and Operations Research 36(7): 2179-2192.

[20] L.D. Stone, Necessary and suffcient conditions for optimal search plans for moving targets, Mathematics of Operations Research4 (4)(1979), pp. 431-440. MathSciNet.

[21] L.D. Stone, What's happened in search theory since the 1975 Lanchester prize? Operations Research, 37, 3 (May-June 1989), 501-506.

[22] L.D. Stone, Theory of Optimal Search (2nd ed.). Arlington, VA: Operations Research Society of America, ORSA Books, 1989.

[23] W.R. Stromquist and L.D. Stone, Constrained optimization of functionals with search theory applications, Mathematics of Operations Research 6(4)(1981), pp. 518-527.

# Assignment Problem

Optim Lett (2012) 6:315–329
DOI 10.1007/s11590-010-0259-x

ORIGINAL PAPER

# Globally solving a nonlinear UAV task assignment problem by stochastic and deterministic optimization approaches

**Hoai An Le Thi · Duc Manh Nguyen · Tao Pham Dinh**

Received: 11 October 2010 / Accepted: 25 October 2010 / Published online: 25 November 2010


**Abstract** In this paper, we consider a task allocation model that consists of assigning a set of $m$ unmanned aerial vehicles (UAVs) to a set of $n$ tasks in an optimal way. The optimality is quantified by target scores. The mission is to maximize the target score while satisfying capacity constraints of both the UAVs and the tasks. This problem is known to be NP-hard. Existing algorithms are not suitable for the large scale setting. Scalability and robustness are recognized as two main issues. We deal with these issues by two optimization approaches. The first approach is the Cross-Entropy (CE) method, a generic and practical tool of stochastic optimization for solving NP-hard problem. The second one is Branch and Bound algorithm, an efficient classical tool of global deterministic optimization. The numerical results show the efficiency of our approaches, in particular the CE method for very large scale setting.

**Keywords** UAV · Task assignment problem · Stochastic programming · Binary nonlinear programming · Cross-entropy (CE) method · Brand and bound algorithm

H. A. Le Thi (✉)
Laboratory of Theoretical and Applied Computer Science, UFR MIM,
Paul Verlaine University, Metz, Ile du Saulcy, 57045 Metz, France
e-mail: lethi@univ-metz.fr

D. M. Nguyen · T. Pham Dinh
Laboratory of Modelling, Optimization and Operations Research,
National Institute for Applied Sciences, Rouen, 76801 Saint-Etienne-du-Rouvray, France
e-mail: duc.nguyen@insa-rouen.fr

T. Pham Dinh
e-mail: pham@insa-rouen.fr

## 1 Introduction

The use of unmanned aerial vehicles (UAVs) for various military missions has received growing attention in the past years. Apart from the obvious advantage of not placing human life at risk, the lack of a human pilot enables considerable weight savings and lower costs. On the other hand, UAVs provide an opportunity for new operational paradigms. However, to realize these advantages, UAVs must have a high level of autonomy and capacity to work cooperatively in groups. In this context, several algorithms dealing with the problem of commanding multiple UAVs to cooperatively perform multiple tasks have been developed. The aim is to assign specific tasks and flyable trajectories to each vehicle to maximize the group performance. The intrinsic uncertainty imbedded in military operations makes the problem more challenging. Scalability and robustness are recognized as two main issues. Also, to allow implementation, the developed algorithms must be solved in real time.

Extensive research has been done recently in this field [4,7,9,12–18,20,21,27–30]. In [4,17,20], task allocation has been formulated in the form of Mixed-Integer Linear Programming (MILP). In this approach, the problem is solved as a deterministic optimization problem with known parameters. Since the MILP is NP-hard, it suffers from poor scalability although the solutions preserve global optimality [19]. Moreover, military situations are in general dynamic and uncertain because of the UAV's sensing limitation and adversarial strategies. Thus, replanning is necessary whenever the information is updated. Heuristics and ad-hoc methods have been considered during replanning in [14,17]. On the other hand, uncertainty is considered via optimization parameters, and risk management techniques in finance are utilized (see e.g. [21,28]). In [21], a nonlinear integer programming problem is formulated where a risk measure by conditional value-at-risk is considered as constraint. In [28], a robust approach using the Soyster formulation on the expectation of the target scores is investigated. These approaches are based on solving hard combinatorial optimization problems and then scalability is still a big issue. An alternative approach dealing with uncertainties consists of formulating a stochastic optimal control problem by using the method of Model Predictive Control (MPC) [6,27].

In this paper, we are interested in task allocation models where we seek to assign a set of $m$ UAVs to a set of $n$ tasks in an optimal way. The optimality is quantified by target scores. The mission is to maximize the target score while satisfying capacity constraints of both the UAVs and the tasks.

The scoring scheme defining effectiveness in our work is a nonlinear function. More precisely, our considered problem is a mixed integer nonlinear programming problem for which the classical MILP solution method can not be used. We propose two approaches to tackle it. The first approach is the Cross-Entropy (CE) method, a simple generic and practical tool of stochastic optimization for solving NP-hard problem. The second one is the Branch and Bound algorithm, an efficient classical tool of global deterministic optimization.

The CE method was originally developed in [22] for an adaptive networks, where an adaptive variance minimization algorithm for estimating probabilities of rare events for stochastic networks was presented. It was modified in [23,24] to solve optimization problems. Several recent publications demonstrate the power of the CE method

as simple and efficient approach for many applications such as telecommunication systems, buffer allocation, vehicle routing, DNA sequence alignment, Machine Learning, etc. It has been proved that this method is particularly relevant for solving "hard" combinatorial optimization problems. In fact, when deterministic methods failed to find the optimal solution within a reasonable time, in most cases the CE method allows to find a fairly good solution more quickly. This motivates us to investigate the CE method for large scale UAV Task Assignment Problem. For measuring the efficiency of the CE method and globally solving the considered problem, we develop a Brand and Bound (B&B) algorithm and compare the two methods.

The rest of paper is organized as follows. In Sect. 2, we describe the problem and give its mathematical formulation. Section 3 is dedicated to the description of CE method and its application for solving the considered problem. The B&B is presented in Sect. 4 while the numerical experiments are reported in Sect. 5. Finally we conclude the paper by Sect. 6.

## 2 Problem statement

Let $V$ and $T$ be the sets of $m$ UAVs and $n$ targets, respectively. The scoring scheme defining effectiveness is based on the definition of target score. Each target $j$ has an associated score based on the task success probability $r_j$ and a weight $w_j$ measuring the importance of the target. The probability that the task will be successfully carried out for that target depends on $y_j$, the number of UAVs which have been assigned to the target $j$, in the following way:

$$1 - (1 - r_j)^{y_j}.$$

A target score is computed as the product of the success probability and its weight:

$$g_j(y_j) = w_j(1 - (1 - r_j)^{y_j}), \tag{1}$$

and the UAVs group effectiveness is simply the sum of all individual target scores: $\sum_{j\in T} g_j(y_j)$. Then the goal is to maximize the UAVs group effectiveness.

Let $z_{ij}$, for $i \in V = \{1, \ldots, m\}$ and $j \in T = \{1, \ldots, n\}$, be the decision variable defined by: $z_{ij}$ is equal to 1 when the UAV $i$ is assigned to the target $j$, and 0 otherwise. An entry of $m \times n$ adjacency matrix A, $a_{ij}$, indicates which target each UAV can be assigned. So, the number $y_j$ can be computed as $y_j = \sum_{i\in V} a_{ij} z_{ij},\ j \in T$.

The mathematical model of this problem can be written as follows:

$$\begin{cases} \max\limits_{z,y} \sum\limits_{j\in T} w_j(1 - (1 - r_j)^{y_j}) \\ s.t.\quad y_j = \sum\limits_{i\in V} a_{ij} z_{ij}, \quad j \in T, \\ \sum\limits_{j\in T} z_{ij} = 1, \quad i \in V, \\ z_{ij} \in \{0, 1\}, \quad i \in V,\ j \in T. \end{cases} \tag{2}$$

The second constraints ensure that each UAV $i$ is used for only one task. This problem is an integer nonlinear programming which is known to be very hard.

## 3 A Cross-Entropy algorithm for solving the UAV task assignment problem (2)

### 3.1 An introduction to Cross-Entropy method

The CE method is a relatively new method for solving both continuous multi-extremal and combinatorial optimization problems. It was originally developed in the rare-event estimation framework [22] as an adaptive importance sampling scheme for estimating rare event probabilities via simulation. This approach was afterward modified in [23,24] for solving both continuous multi-extremal and combinatorial optimization problems. The main idea of the CE method is the construction of a random sequence of solutions which converges probabilistically to the optimal or near-optimal solution. It involves the following two iterative phases:

1. Generation of a sample of random data (trajectories, vectors, etc.) according to a specified random mechanism.
2. Updating the parameters of the random mechanism, typically parameters of pdfs (probability density functions), on the basis of the data, to produce a "better" sample in the next iteration.

Unlike most of the stochastic algorithms for optimization which are based on local search, the CE method is a global random search procedure. The CE method was successfully applied to various problems such as the traveling salesman problem [23], the bipartition problem [23], the maximal cut problem [25], the image matching [10], the image segmentation [11], etc.

For a comprehensive overview and history of the CE method, the reader is referred to [26]. For the sake of completion we present below the generic CE scheme for combinatorial optimization problems.

Consider the problem of minimizing the function $S$ over a finite set $\mathscr{X}$, say

$$\gamma^* = \min_{x \in \mathscr{X}} S(x). \tag{3}$$

The starting point in the methodology of the CE method applied to (3) is to associate an estimation problem with the optimization problem (3). To this end one defines a collection of indicator functions $I_{\{S(x) \le \gamma\}}$ on $\mathscr{X}$ for various thresholds or levels $\gamma \in \mathbb{R}$. Next, let $\{f(\cdot; v), v \in V\}$ be a family of (discrete) probability density functions (pdfs) on $\mathscr{X}$, parameterized by a real-valued (vector) $v$.

For some $u \in V$, we consider the *Associated Stochastic Problem* (ASP):

$$\ell(\gamma) = \mathbb{P}_u(S(x) \le \gamma) = \sum_{x \in \mathscr{X}} I_{\{S(x) \le \gamma\}} f(x; u) = \mathbb{E}_u I_{\{S(x) \le \gamma\}}, \tag{4}$$

where $\mathbb{P}_u$ is the probability measure under which the random state $\mathscr{X}$ has the pdf $f(\cdot; u)$, and $\mathbb{E}_u$ denotes the corresponding expectation operator. The idea of CE method

is to construct simultaneously two sequences of levels $\widehat{\gamma}_1, \widehat{\gamma}_2, \ldots, \widehat{\gamma}_T$ and parameters (vectors) $\widehat{v}_1, \widehat{v}_2, \ldots, \widehat{v}_T$ such that $\widehat{\gamma}_T$ is close to the optimal $\gamma^*$, and $\widehat{v}_T$ is such that the corresponding density assigns high probability mass to the collection of states that give a low value. More specifically, one initializes by setting $v_0 = u$ and choosing a not very small quantity $\theta$, and then proceeds as follows:

1. **Adaptive updating of $\gamma_t$.** For a fixed $v_t$, let $\gamma_t$ be the $\theta$-quantile of $S(X)$ under $v_{t-1}$. That is, $\gamma_t$ satisfies

$$\mathbb{P}_{v_{t-1}}(S(X) \geq \gamma_t) \geq 1 - \theta, \tag{5}$$

$$\mathbb{P}_{v_{t-1}}(S(X) \leq \gamma_t) \geq \theta, \tag{6}$$

   where $X \sim f(\cdot; v_{t-1})$.
   A simple estimator of $\gamma_t$, denotes $\widehat{\gamma}_t$, can be obtained by drawing a random sample $X^1, X^2, \ldots, X^N$ from $f(\cdot; v_{t-1})$. Suppose that $S(X^{\sigma(1)}) \leq S(X^{\sigma(2)}) \leq \cdots \leq S(X^{\sigma(N)})$, where $\sigma$ is a permutation of the set $\{1, \ldots, N\}$. Estimate the $\theta$-quantile of $S(X)$ as

$$\widehat{\gamma}_t = S_{\lfloor \theta N \rfloor}. \tag{7}$$

2. **Adaptive updating of $v_t$.** For a fixed $\gamma_t$ and $v_{t-1}$, derive $v_t$ by minimizing the Kullback-Leibler distance, or equivalent to solving the program

$$\max_v \mathbb{E}_{v_{t-1}} I_{\{S(X) \leq \gamma_t\}} \ln f(X; v). \tag{8}$$

   The stochastic counterpart of (8) is as follows: for fixed $\widehat{\gamma}_t$ and $\widehat{v}_{t-1}$ the estimate of $v_{t-1}$, derive $\widehat{v}_t$ from the solution of following program

$$\max_v D(v) := \frac{1}{N} \sum_{i=1}^{N} I_{\{S(X^i) \leq \gamma_t\}} \ln f(X^i; v). \tag{9}$$

   In typical applications, the function $D$ is concave and differentiable with respect to $v$, and thus updating Eq. (9) is equivalent to solving the following system of equations:

$$\frac{1}{N} \sum_{i=1}^{N} I_{\{S(X^i) \leq \gamma_t\}} \nabla \ln f(X^i; v) = 0, \tag{10}$$

   where the gradient is with respect to $v$.

**CE Algorithm for combinatorial optimization**

1. Choose $\widehat{v}_0$, and $0 < \theta < 1$. Set $t = 1$.
2. Generate $N$ samples $X^1, X^2, \ldots, X^N$ according to $f(\cdot; \widehat{v}_{t-1})$, and compute $\theta$-quantile $\widehat{\gamma}_t$ of $S$ according to (7).

3. Use the same samples $X^1, X^2, \ldots, X^N$ to solve the stochastic programming problem (9). Denote the solution by $\widehat{v}_t$.
4. If for some $t \geq d$, say $d = 5$ such that

$$\widehat{\gamma}_t = \widehat{\gamma}_{t-1} = \cdots = \widehat{\gamma}_{t-d},$$

then **stop**; otherwise set $t = t + 1$, reiterate from step 2.

For the convergence analysis of the CE method we refer to [5,8,26].

### 3.2 A Cross-Entropy method for solving problem (2)

We first rewrite the problem (2) in the form:

$$\begin{cases} \min\limits_{z} f(z) := \sum\limits_{j \in T} w_j \left(1 - r_j\right)^{\sum_{i \in V} a_{ij} z_{ij}} \\ s.t. \quad \sum\limits_{j \in T} z_{ij} = 1, \quad \forall i \in V, \\ z_{ij} \in \{0, 1\}, \quad \forall i \in V, j \in T. \end{cases} \tag{11}$$

Denote by $Z$ the feasible set of (11), say

$$Z := \left\{ z = (z_{ij}) \in \{0, 1\}^{mn} : \sum_{j \in T} z_{ij} = 1, i \in V \right\}.$$

It is clear that each variable $z \in Z$ is identical to an assignment mapping $m_z : \{1, \ldots, m\} \longmapsto \{1, \ldots, n\}$, i.e., $m_z(i) = j$ iff $z_{ij} = 1$. Then, the set $Z$ is identical to the set:

$$\mathscr{X} = \{x = (x_1, \ldots, x_m) : x_i \in \{1, \ldots, n\} \text{ is the target assigned to UAV } i\}. \tag{12}$$

The CE algorithm draws particular assignment of UAVs to targets that will be evaluated and then selected, in order to obtain a drawing law which will converge toward the optimal assignment. First, we must choose a family of pdf, $f(\cdot; v)$, describing a probability choice of $x$.

A discrete probability law $p(j|i)$ is associated to each UAV $i$. It represents the probability to assign the UAV $i$ to the target $j$. These probabilities are summarized by the matrix $M$:

$$M = \begin{pmatrix} p(1|1) & p(2|1) & \cdots & p(n|1) \\ p(1|2) & p(2|2) & \cdots & p(n|2) \\ & & \cdots & \\ p(1|m) & p(2|m) & \cdots & p(n|m) \end{pmatrix}.$$

Note that, we must have

$$\sum_{j=1}^{n} p^M(j|i) = 1.$$

Let $X = (x_1, x_2, \ldots, x_m)$ be the random assignment vector of UAVs to targets, $x_i$ is the target assigned to UAV $i$ for the draw. The probability of drawing the vector according to $M$ is

$$p(X) = \prod_{i=1}^{m} p(x_i|i),$$

where $p(x_i|i)$ is the coefficient in the column $x_i$ and the row $i$ of matrix $M$.

In each iteration, suppose that $X^k$, $X^k = (x_1^k, \ldots, x_i^k, \ldots, x_m^k)$, $k = 1, 2, \ldots, N$ are the samples drawn. The $H = \lfloor \theta N \rfloor$ best samples, according to the objective function $f$, are selected to update $M$. Denoting $\{X^1, X^2, \ldots, X^H\}$ as the $H$ "best" vectors among the draws $\{X^1, X^2, \ldots, X^N\}$. Minimizing the Kullback–Leibler distance leads to the following optimization problem:

$$\begin{cases} \max\limits_{p(j|i)} \mathscr{K} := \sum\limits_{h=1}^{H} \ln\left(\prod\limits_{i=1}^{m} p(x_i^h|i)\right) \\ s.t. \quad \sum\limits_{j=1}^{n} p(j|i) = 1, \quad i = 1, \ldots, m, \\ p(j|i) \geq 0, \quad i = 1, \ldots, m, \ j = 1, \ldots, n. \end{cases} \tag{13}$$

We first rewrite the objective function as follows

$$\begin{aligned} \mathscr{K} &= \sum_{h=1}^{H} \ln\left(\prod_{i=1}^{m} p\left(x_i^h|i\right)\right) = \sum_{i=1}^{m}\sum_{h=1}^{H} \ln\left(p\left(x_i^h|i\right)\right) \\ &= \sum_{i=1}^{m}\sum_{j=1}^{n} \mathrm{card}\{h \in \{1, \ldots, H\} : x_i^h = j\} \ln(p(j|i)) \end{aligned}$$

Denoting

$$u_{ji} = p(j|i), b_{ji} = \mathrm{card}\{h \in \{1, \ldots, H\} : x_i^h = j\}.$$

The program (13) is equivalent to

$$\begin{cases} \min\limits_{u_{ji}} \left(-\sum\limits_{i=1}^{m}\sum\limits_{j=1}^{n} b_{ji} \ln(u_{ji})\right) \\ s.t. \quad \sum\limits_{j=1}^{n} u_{ji} = 1, \quad i = 1, \ldots, m, \\ u_{ji} \geq 0, \quad i = 1, \ldots, m, \ j = 1, \ldots, n. \end{cases} \tag{14}$$

Since it is a convex programming problem, we consider the Karusk–Kuhn-Tucker (KKT) condition of the program (14):

$$\begin{cases} -\frac{b_{ji}}{u_{ji}} + \lambda_i - \mu_{ji} = 0, \quad i = 1, \ldots, m, \ j = 1, \ldots, n, \\ \lambda_i \left( \sum_{j=1}^{n} u_{ji} - 1 \right) = 0, \quad i = 1, \ldots, m, \\ \mu_{ji} u_{ji} = 0, \quad i = 1, \ldots, m, \ j = 1, \ldots, n, \\ \lambda_i \geq 0, \quad i = 1, \ldots, m, \\ \mu_{ji} \geq 0, \quad i = 1, \ldots, m, \ j = 1, \ldots, n. \end{cases}$$

Then by solving the KKT condition, we get the updating formula of matrix $M$ as follows

$$p(j|i) := \frac{\text{card}\{h \in \{1, \ldots, H\} : x_i^h = j\}}{H}.$$

**The CE algorithm for solving the Problem (11):**

*Step 1.* Initialize $M = M_0 = (p_0(j|i))_{m \times n}$ a uniform distribution, i. e.,

$$p_0(j|i) = \frac{1}{n}, \quad i = 1, \ldots, m, \ j = 1, \ldots, n,$$

and choose $\theta \in (0, 1)$,

*Step 2.* Draw $N$ samples $X^1, X^2, \ldots, X^N$ according to $M$. Compute $f(X^k)$, $k = 1, 2, \ldots, N$,

*Step 3.* Sort the sequence $\{f(X^k)\}_{k=1}^N$ in the increasing orders. Let $f(X^{\sigma(1)}) \leq f(X^{\sigma(2)}) \leq \cdots \leq f(X^{\sigma(N)})$, where $\sigma$ is a permutation of the set $\{1, 2, \ldots, N\}$. Set $H = \lfloor \theta N \rfloor$, then choose $H$ best draws $X^{\sigma(1)}, X^{\sigma(2)}, \ldots, X^{\sigma(H)}$,

*Step 4.* Update $M$ by the formula

$$p(j|i) := \frac{\text{card}\{h \in \{1, 2, \ldots, H\} : \quad x_i^{\sigma(h)} = j\}}{H},$$

*Step 5.* Iterate step 2, 3, 4 until convergence.

In practice, we should choose $M_0$ in the Step 1 as follows:

$$p_0(j|i) = \begin{cases} \frac{1}{n_i} & \text{if } a_{ij} = 1, \\ 0 & \text{if } a_{ij} = 0, \end{cases}$$

where $n_i = \sum_{j=1}^{n} a_{ij}, i = 1, \ldots, m$.

## 4 A Branch and Bound algorithm

For globally solving the considered problem, and for measuring the quality of our CE algorithm, we develop a global approach based on classical Branch and Bound (B&B) scheme. The lower bounds are computed by solving the relaxed problem:

$$\min \left\{ f(z) = \sum_{j=1}^{n} w_j (1 - r_j)^{\sum_{i=1}^{m} a_{ij} z_{ij}} : z \in K \right\}, \tag{15}$$

where $K$ is nonempty bounded polyhedral convex set in $\mathbb{R}^{mn}$.

Since this problem is convex, it can be solved by standard solvers for convex programming. Here, we use DCA [1–3] to solve it. DCA is an efficient approach for DC (Difference of Convex functions) programming problems. It addresses a general DC program of the form

$$\alpha := \inf \left\{ f(z) := g(z) - h(z) : z \in \mathbb{R}^n \right\} \quad (P_{dc})$$

where $g$ and $h$ are convex, lower semicontinuous proper functions. DC programs with closed convex set constraints $C$ can be cast into $(P_{dc})$ by adding $\chi_C$, the indicator function on $C$, with $g$. ($\chi_C(z) = 0$ if $z \in C$, $+\infty$ otherwise).

The DC (Difference of Convex functions) programming and DCA (DC Algorithms) constitute the backbone of Nonconvex Programming and Global Optimization. They were introduced by Tao in 1985 and extensively developed by An and Tao since 1994 to become now classic and increasingly popular. It is clear that convex programs are *false* DC programs for which DCA can be used. On the other hand, with suitable DC decompositions, DCA applied to convex programs permits to find again standard optimization methods for convex programming.

The idea of DCA is quite simple: each iteration $k$ one linearizes the concave part $-h$ and then solve the resulting convex program. More precisely, DCA consists of computing at each iteration $k$

$$y^k \in \partial h(z^k), \quad z^{k+1} \in \arg\min_{z \in \mathbb{R}^n} \left\{ g(z) - h(z^k) - \langle z - z^k, y^k \rangle \right\} \ (P_k).$$

For solving (15) we use the following DC decomposition:

$$f(z) = \sum_{j=1}^{n} w_j (1 - r_j)^{\sum_{i=1}^{m} a_{ij} z_{ij}} = g(z) - h(z), \tag{16}$$

where

$$g(z) := \frac{\lambda}{2}\|z\|^2; \quad h(z) := \frac{\lambda}{2}\|z\|^2 - \sum_{j=1}^{n} w_j (1 - r_j)^{\sum_{i=1}^{m} a_{ij} z_{ij}}. \tag{17}$$

Here, $\lambda$ takes the associated value such that $h$ is convex.

**Description of the DCA for problem (15)**

*Step 1.* Initialization: Choose $z^0 \in K$, $\varepsilon_1 > 0$, and $\varepsilon_2 > 0$. Set $k = 0$.

*Step 2.* Compute $y^k = \nabla h(z^k)$, with

$$y_{ij}^k = \lambda z_{ij}^k - w_j \log(1 - r_j) a_{ij} (1 - r_j)^{\sum_{i=1}^{m} a_{ij} z_{ij}^k}$$

for $i = 1, \ldots, m$, $j = 1, \ldots, n$.

*Step 3.* Compute $z^{k+1}$ by solving the convex quadratic problem

$$\begin{cases} \min\limits_{z} \left( g(z) - \langle z, y^k \rangle \right) \\ s.t. \quad z \in K. \end{cases}$$

*Step 4.* Iterate Step 2 and 3 until

$$\left| f\left(z^{k+1}\right) - f(z^k) \right| \leq \varepsilon_1 \left(1 + \left| f\left(z^{k+1}\right) \right| \right)$$
$$\text{or} \quad \left\| z^{k+1} - z^k \right\| \leq \varepsilon_2 \left(1 + \left\| z^{k+1} \right\| \right).$$

Let $\Omega$ be the set defined by

$$\Omega := \left\{ z = (z_{ij}) \in [0,1]^{mn} : \sum_{j \in T} z_{ij} = 1, i \in V \right\}.$$

**B&B Algorithm**

Let $R_0 := [0,1]^{m.n}$ and $\varepsilon$ be a sufficiently small positive number. Set *restart := true*; Solve the convex problem (15) with $K \leftarrow K_{R_0} = \Omega \cap R_0$ to obtain a solution $z^{R_0}$ and the first lower bound $\beta_0 := \beta(R_0)$;

**If** $z^{R_0}$ is feasible to (11) **then**

set $\gamma_0 := f(z^{R_0})$, $z^0 := z^{R_0}$, *restart := false* **else** $\gamma_0 := +\infty$;

**Endif**

**If** $(\gamma_0 - \beta_0) \leq \varepsilon |\gamma_0|)$ **then** STOP, $z^0$ is an $\varepsilon$-optimal solution of (11) **else** set $\Re \leftarrow \{R_0\}, k \leftarrow 0$;

**Endif**

**While** (STOP = false) **do**

Select a rectangle $R_k$ such that $\beta_k = \beta(R_k) = \min\{\beta(R) : R \in \Re\}$.

Let $j^* \in \{1, \ldots, m.n\}$ be the index such that $z_{j^*}^{R_k} \notin \{0, 1\}$. Divide $R_k$ into two sub-rectangles $R_{k_0}$ and $R_{k_1}$ via the index $j^*$:

$$R_{k_i} = \{z \in R_k : z_{j^*} = i; i = 0, 1\}. \tag{18}$$

Solve the subproblems $(P_{k_i})$ to obtain $\beta(R_{k_i})$ and $(z^{R_{k_i}})$:

$$(P_{k_i}) \quad \beta(R_{k_i}) = \min\{f(z) : z \in \Omega, z \in R_{k_i}\}. \tag{19}$$

**For** $i = 0, 1$
  **If** $z^{R_{k_i}}$ is feasible to (11) **then**
    update $\gamma_k$ and the best feasible solution $z^k$;
  **Endif**
**Endfor**
Set $\Re \leftarrow \Re \cup \{R_{k_i} : \beta(R_{k_i}) < \gamma_k - \varepsilon, i = 0, 1\} \setminus R_k$.
**If** $\Re = \emptyset$ **then** STOP, $z^k$ is an $\varepsilon$-optimal solution **else** set $k \leftarrow k + 1$.
**Endwhile**

## 5 Numerical results

The algorithms are written in language C on Microsoft Visual C++ 2008. The implementation takes place on a notebook with chipset Intel(R) Core(TM) Duo CPU 2.0 GHz, RAM 3GB. The commercial software CPLEX 11.2 is used as a convex quadratic programming solver.

The following notations are used in these tables:

- $Pb$: Problem,
- $Bin$: number of the binary variables,
- $Ctrs$: number of the constraints,
- $Obj$: value of the objective function obtained by each algorithm,
- $Time$: CPU time in seconds of each algorithm,
- $Gap\% = \frac{Obj - Lastlowerbound}{Obj}$,
- $GapCE\% = \frac{Obj - Firstlowerbound}{Obj}$.

In Table 1, we give a comparison between CE and B&B. The results have demonstrated that with small dimension, CE gives good solutions as the same as B&B does, but CPU time in our approach is very better. The ratio of time consumed varies from 114 to 10,218. Especially, for Problem 4, B&B runs for more than one hour and does not produce a solution, whereas CE gives a good solution. In this problem, we compute GapCE by using the first lower bound. Figures 1 and 2 show the result for nine out of ten problems. In Table 2, we continue to compare in larger dimension, with $m = 20, n = 10$. We can see that CE still works very well while B&B runs for more than one hour and does not produce a solution, for all problems. In these experiments CE was run with number of samples, $N = 50$, and $\theta = 0.4$. The number of iterations is limited to 20.

Table 3 gives the results with 10 instances with large dimensions having $50 \times 30 = 1{,}500$ binary variables. The weights $\{w_j\}$ are random integers chosen uniformly from 1 to 10, and the task success probabilities $\{r_j\}$ are random numbers chosen uniformly in interval [0, 1]. The parameters are chosen as follows: number of samples is $N = 100, \theta = 0.4$. Table 4 presents the results for very large dimensions. In this case we take $\theta = 0.04$. The maximum of number of iterations is 50.

**Table 1** CE compares with B&B, $m = 10, n = 10$

| Data | | | CE method | | | B&B | | |
|---|---|---|---|---|---|---|---|---|
| Pb | Bin | Ctrs | Obj | Time(s) | Gap | Obj | Time(s) | Gap |
| 1 | 100 | 10 | 29.425828 | 0.078 | 0.1999 | 29.425858 | 170.890 | 0.2 |
| 2 | 100 | 10 | 40.899408 | 1.156 | 0.5556 | 40.790448 | 1919.812 | 0.29 |
| 3 | 100 | 10 | 30.256247 | 0.078 | 1.2553 | 29.927309 | 776.766 | 0.17 |
| 4 | 100 | 10 | 26.001069 | 1.140 | 2.85 | NA | >1 h | NA |
| 5 | 100 | 10 | 34.964398 | 0.078 | 0.06 | 34.964398 | 8.937 | 0.06 |
| 6 | 100 | 10 | 28.286803 | 2.890 | 1.03 | 28.286803 | 2887.265 | 1.03 |
| 7 | 100 | 10 | 35.185966 | 0.078 | 0.3965 | 35.454166 | 47.859 | 1.15 |
| 8 | 100 | 10 | 29.528116 | 0.078 | 0.33 | 29.528116 | 23.110 | 0.33 |
| 9 | 100 | 10 | 19.188199 | 0.078 | 0.46 | 19.188199 | 38.578 | 0.46 |
| 10 | 100 | 10 | 15.761578 | 0.078 | 2.08 | 15.761578 | 797.063 | 2.08 |

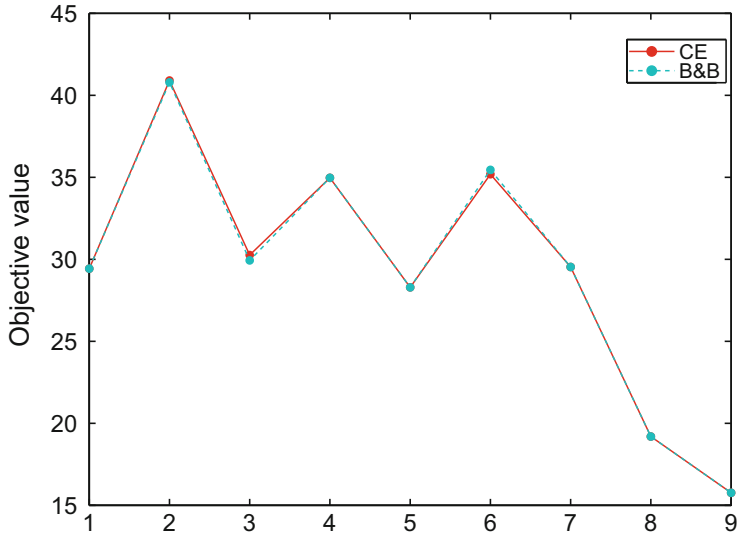


**Fig. 1** Objective function, $m = 10, n = 10$

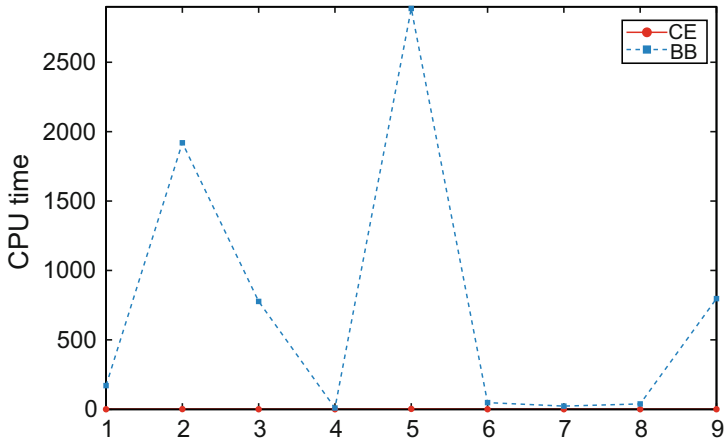


**Fig. 2** CPU time, $m = 10, n = 10$

**Table 2** CE compares with B&B, $m = 20, n = 10$

| Data | | | CE method | | | B&B | |
|---|---|---|---|---|---|---|---|
| Pb | Bin | Ctrs | Obj | Time(s) | GapCE | Obj | Time (h) |
| 1 | 200 | 20 | 23.765072 | 0.203 | 0.2 | NA | >1 |
| 2 | 200 | 20 | 21.351345 | 0.218 | 0.56 | NA | >1 |
| 3 | 200 | 20 | 22.799311 | 0.187 | 1.26 | NA | >1 |
| 4 | 200 | 20 | 23.388861 | 0.203 | 2.85 | NA | >1 |
| 5 | 200 | 20 | 18.416153 | 0.187 | 0.06 | NA | >1 |
| 6 | 200 | 20 | 25.096867 | 0.203 | 1.03 | NA | >1 |
| 7 | 200 | 20 | 22.698760 | 0.187 | 0.4 | NA | >1 |
| 8 | 200 | 20 | 19.459113 | 0.218 | 0.33 | NA | >1 |
| 9 | 200 | 20 | 13.077463 | 0.187 | 0.46 | NA | >1 |
| 10 | 200 | 20 | 23.539310 | 0.203 | 2.08 | NA | >1 |

**Table 3** Results with large dimensions: $m = 50, n = 30$

| Instance | Bin | Ctrs | Time(s) | GapCE |
|---|---|---|---|---|
| 1 | 1,500 | 50 | 4.766 | 2.562583 |
| 2 | 1,500 | 50 | 4.750 | 3.315747 |
| 3 | 1,500 | 50 | 4.797 | 2.816054 |
| 4 | 1,500 | 50 | 4.797 | 1.868424 |
| 5 | 1,500 | 50 | 5.047 | 3.784732 |
| 6 | 1,500 | 50 | 4.875 | 2.093580 |
| 7 | 1,500 | 50 | 4.968 | 3.010455 |
| 8 | 1,500 | 50 | 4.797 | 1.401817 |
| 9 | 1,500 | 50 | 4.937 | 3.711038 |
| 10 | 1,500 | 50 | 4.860 | 2.658826 |

**Table 4** Results with very large dimensions

| Instance | $m$ | $n$ | Bin | Ctrs | Time (s) | GapCE | Samples |
|---|---|---|---|---|---|---|---|
| 1 | 300 | 100 | 30,000 | 300 | 141.078 | 1.270807 | 1,000 |
| 2 | 500 | 100 | 50,000 | 500 | 229.593 | 1.829299 | 1,000 |
| 3 | 500 | 300 | 150,000 | 500 | 1190.437 | 3.139713 | 2,000 |
| 4 | 700 | 500 | 350,000 | 700 | 2306.141 | 4.031592 | 2,000 |
| 5 | 1,000 | 500 | 500,000 | 1,000 | 3649.781 | 5.114315 | 2,000 |

## 6 Conclusion

In this work, we have firstly proposed an approach based on the CE method for solving UAV Task Assignment problem, and then presented a global approach based on B&B algorithm for measuring the quality of our CE algorithm. The results have shown the efficiency of this approach not only with small dimensions but also with very large dimensions. This approach can overcome very well the barrier of binary variables, that

the standard methods are usually very difficult for treating. Other non-linear models of task assignment problem will be studied in our future work.

## References


1. Pham Dinh, T., Le Thi, H.A.: Convex analysis approach to DC programming: theory, algorithms and applications (dedicated to Professor Hoang Tuy on the occasion of his 70th birthday). Acta Math. Vietnamica **22**, 289–355 (1997)
2. Le Thi, H.A., Pham Dinh, T.: Large scale global molecular optimization from exact distance matrices by a DC optimization approach. SIAM J. Optim. **14**(1), 77–114 (2003)
3. Le Thi, H.A., Pham Dinh, T.: The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems. Ann. Oper. Res. **133**, 23–46 (2005)
4. Chandler, P.R., Pachter, M., Rasmussen, S.R., Schumacher, C.: Multiple Task Assignment for a UAV Team. AIAA Guidance, navigation, and control conference, Monterey, Aug. (2002)
5. Costa, A., Dafydd, O., Kroese, D.: Convergence properties of the cross-entropy method for discrete optimization. Oper. Res. Lett. **35**(5), 57380 (2007)
6. Cruz, J.B. Jr., Chen, G., Li D., Wang, X.: Particle swarm optimization for resource allocation in UAV cooperative control. AIAA Guidance, Navigation, and Control Conference, Providence, Aug. (2004)
7. Fontes, D.B.M.M., Fontes, F.A.C.C.: Minimal switching time formations with collision avoidance. Dyn. Inf. Syst. Springer Optim. Appl. **40**, 305–321 (2010)
8. Dambreville, F.: Cross-entropy method: convergence issues for extended implementation. In: Proceedings of the Rare Event Simulation Conference (RESIM 2006), Bamberg, Germany (2006)
9. Walker, D.H., McLain, T.W., Howlett, J.K. : Coordinated UAV target assignment using distributed tour calculation. In: Grundel, D., Murphy, R., Pardalos, P.M. (eds.) Theory and Algorithms for Cooperative Systems, Series on Computers and Operations Research, vol. 4, pp. 327–333. Kluwer, Dordrecht (2004)
10. Dubin, U.: The cross-entropy method for combinatorial optimization with applications. Master's thesis, The Technion, Israel Institute of Technology, Haifa, June (2002)
11. Dubin, U.: Application of the cross-entropy method for image segmentation. Ann. Oper. Res. (2004) (submitted)
12. Pasiliao, E.L.: Local neighborhoods for the multidimensional assignment problem. In: Hirsch, M.J., Pardalos, P.M., Murphey, R. (eds.) Dynamics of Information Systems, Springer Optimization and its Applications, vol. 40, pp. 353–371. Springer, Berlin (2010)
13. Krokhmal, P., Murphey, R., Pardalos, P., Uryasev, S. : Use of conditional value-at-risk in stochastic programs with poorly defined distributions. In: Butenko, S., Murphey, R., Pardalos, P. (eds.) Recent Developments in Cooperative Control and Optimization, pp. 225–242. Kluwer, Dordrecht (2004)
14. Le Ny, J., Feron, E.: An Approximation Algorithm for the Curvature-Constrained Travelling Salesman Problem. 43rd Annual Allerton Conference on Communications, Control and Computing, Monticello, Sept. (2005)
15. Murray, R.M.: Recent research in cooperative control of multivehicle systems. J. Dyn. Syst. Measure. Control **129**(5), 571–583 (2007)
16. Nygard, K.E., Altenburg, K., Tang, K., Schesvold, D.: A decentralized swarm approach to asset patrolling with unmanned air vehicles. In: Grundel, D., Murphy, R., Pardalos, P.M. (eds.) Theory and Algorithms for Cooperative Systems, Series on Computers and Operations Research, vol. 4, pp. 327–338. Kluwer, Dordrecht (2004)
17. Papadimitriou, C.H., Steiglitz, K.: Combinatorial Optimization: Algorithms and Complexity. Prentice-Hall, Englewood Cliffs (1982)
18. Pardalos, P.M., Pitsoulis, L.S. (eds.): Nonlinear Assignment Problems: Algorithms and Applications. Combinatorial Optimization, vol. 7. Kluwer, Dordrecht (2000)
19. Protvin, J.Y.: Genetic algorithms for the travelling salesman problem. Ann. Oper. Res. **63**(3), 339–370 (1996)
20. Richards, A., Bellingham, J., Tillerson, M., How, J.P.: Coordination and control of multiple UAVs. AIAA Guidance, Navigation and Control Conference, Monterey, Aug. (2002)
21. Richards, A., How, J.P.: Aircraft Trajectory Planning With Collision Avoidance Using Mixed Integer Linear Progrmming. American Control Conference, Anchorage, AK, May (2002)

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

# Solving the Multidimentional Assignment Problem via the Cross-Entropy method

**Nguyen Duc Manh · Le Thi Hoai An · Pham Dinh Tao**

**Abstract** The multidimensional assignment problem (MAP) is a higher dimensional version of the linear assignment problem, where we find tuples of elements from given sets, such that the total cost of the tuples is minimal. The MAP has many recognized applications such as data association, target tracking, and resource planning. While the linear assignment problem is solvable in polynomial time, the MAP is NP-hard. In this work, we propose a new approach based on the Cross-Entropy (CE) method for solving the MAP. Due to the structure of the MAP, we design an appropriate family of discrete distributions on the feasible set of the MAP and apply CE method. The efficiency of our method will be shown by testing for the large-scale problems, for instance, the MAP with 5 dimensions and there are 20 elements in each dimension, which is equivalent to a 0-1 linear program with 3.2 millions binary variables and 100 constraints.

**Keywords** Multidimensional Assignment Problem, Combinatorial Optimization, Cross-Entropy.

**Mathematics Subject Classification (2000)** 90

## 1 Introduction

The most basic type of assignment problem is linear assignment, which is considered to be the job-to-worker problem. Suppose that we have a set of $n_1$ jobs and a set of $n_2$ workers ($n_1 \leq n_2$), and $c_{ij}$ is the cost of worker $j$ performing job $i$. The objective is to assign each job to a worker so that all the jobs are done with minimum total cost.

Nguyen Duc Manh · Pham Dinh Tao
LMI, National Institute for Applied Sciences (INSA) - Rouen,
BP08-Avenue de l'Université, 76801, Saint-Étienne-du-Rouvray, France
E-mail: duc.nguyen@insa-rouen.fr, pham@insa-rouen.fr

Le Thi Hoai An
Laboratory of Theoretical and Applied Computer Science (LITA),
Paul Verlaine University of Metz, Ile du Saulcy, F-57045 Metz Cedex, France
E-mail: lethi@univ-metz.fr

Let $x_{ij}$ be decision variable defined as

$$x_{ij} = \begin{cases} 1 & \text{if worker } j \text{ is assigned job } i, \\ 0 & \text{otherwise,} \end{cases}$$

The 0-1 linear program formulation of this problem is described as follows

$$\begin{cases} \min \sum\limits_{i=1}^{n_1} \sum\limits_{j=1}^{n_2} c_{ij} x_{ij} \\ \text{s.t.} \quad \sum\limits_{j=1}^{n_2} x_{ij} = 1, i = 1, 2, ..., n_1, \\ \qquad \sum\limits_{i=1}^{n_1} x_{ij} \leq 1, j = 1, 2, ..., n_2, \\ \qquad x_{ij} \in \{0, 1\}. \end{cases}$$

The constraints guarantee that each worker performs at most one job, and that all jobs are accomplished.

The multidimensional assignment problem (MAP) is simply a higher dimensional version of the linear assignment problem. Denoting $n_k$ the number of elements in dimension $k$ of $d$ dimensions, $n_1 \leq n_k, k = 2, ..., d$, the MAP can be expressed as an integer program as follows

$$\begin{cases} \min \sum\limits_{i_1=1}^{n_1} ... \sum\limits_{i_d=1}^{n_d} c_{i_1 ... i_d} x_{i_1 ... i_d} \\ \text{s.t.} \quad \sum\limits_{i_2=1}^{n_2} ... \sum\limits_{i_d=1}^{n_d} x_{i_1 ... i_d} = 1, i_1 = 1, 2, ..., n_1, \\ \qquad \sum\limits_{i_1=1}^{n_1} ... \sum\limits_{i_{k-1}=1}^{n_{k-1}} \sum\limits_{i_{k+1}=1}^{n_{k+1}} ... \sum\limits_{i_d=1}^{n_d} x_{i_1 ... i_d} \leq 1, i_k = 1, 2, ..., n_k, k = 2, ..., d-1, \\ \qquad \sum\limits_{i_1=1}^{n_1} ... \sum\limits_{i_{d-1}=1}^{n_{d-1}} x_{i_1 ... i_d} \leq 1, i_d = 1, 2, ..., n_d, \\ \qquad x_{i_1 ... i_d} \in \{0, 1\}. \end{cases} \tag{1}$$

An equivalent formulation of this problem, which offers us several approaches to find an optimal assignment is injection formulation

$$\begin{cases} \min \sum\limits_{i=1}^{n_1} c_{i, \pi_1(i), ..., \pi_{d-1}(i)} \\ \text{s.t.} \quad \pi_k : \{1, 2, ..., n_1\} \to \{1, 2, ..., n_{k+1}\} \text{ is injective}, \forall k = 1, 2, .., d-1. \end{cases} \tag{2}$$

The MAP was first introduced by Pierskalla (1968) [24], and since then has found numerous applications in the areas of data association [2], image recognition [36], multisensor multitarget tracking [19,26], tracking of elementary particles [28], etc. For a survey of the MAP and its applications, see [5,6,8,14,21,34].

It is worth to note that while the linear assignment problem is solvable in polynomial time, the MAP is known to be NP-hard. The computational time to find an optimal solution of an MAP grows exponentially with the number of dimensions and factorially with the dimension size [23]. Indeed, the total number cost coefficients, and the number of feasible solutions for a fully dense problem are given by the following expressions [9]:

$$\text{Total number cost coefficients} = \prod_{k=1}^{d} n_k;$$

$$\text{Number of feasible solutions } = \prod_{k=2}^{d} \frac{n_k!}{(n_k - n_1)!}.$$

Despite its inherent difficulty, several exact and heuristic algorithms have been proposed in the literature [1,3,4,9,17,18,20,22–25,27,35]. Most of them are heuristic approaches, such as the GRASP (greedy randomized adaptive search procedure) with Path Relinking for Three-Index Assignment [1], the Simulated Annealing [9], the Local Search Heuristics [17], Local Search Neighborhoods [23], etc. The exact solution methods are the Branch and Bound procedures [16,22,24], the Lagrangian Relaxation Based Algorithms [25,27].

In this paper, we propose a new approach based on the Cross-Entropy (CE) method to tackle the problem (2). The CE method was originally developed in [29] for an adaptive networks, where an adaptive variance minimization algorithm for estimating probabilities of rare events for stochastic networks was presented. It was modified in [30,31] to solve optimization problems. Several recent publications demonstrate the power of the CE method as simple and efficient approach for many applications such as telecommunication systems, buffer allocation, vehicle routing, DNA sequence alignment, Machine Learning, etc. It has been proved that this method is particularly relevant for solving "hard" combinatorial optimization problems. In fact, when deterministic methods failed to find the optimal solution within a reasonable time, in most cases the CE method allows to find a fairly good solution more quickly. This motivates us to investigate the CE method for the MAP.

To apply CE method, the difficulty is how to find an appropriate family of distributions on the feasible set of the optimization problem such that updating the parameters could be done as easily as possible. Due to the formulation (2) of the MAP, we construct a family of discrete distributions, that allows us to apply CE method. It is the main contribution in this paper.

The rest of paper is organized as follows. In section 2, we give a short presentation of the CE method. In Section 3, we propose a CE algorithm for solving our problem. Numerical experiments are reported in Section 4 while some conclusions and perspectives are discussed in Section 5.

## 2 The Cross-Entropy method

The CE method is a relatively new method for solving both continuous multi-extremal and combinatorial optimization problems. It was originally developed in the rare-event estimation framework [29] as an adaptive importance sampling scheme for estimating rare event probabilities via simulation. This approach was afterward modified in [30,31] for solving both continuous multi-extremal and combinatorial optimization problems. The main idea of the CE method is the construction of a random sequence of solutions which converges probabilistically to the optimal or near-optimal solution. It involves the following two iterative phases:

1. Generation of a sample of random data (trajectories, vectors, etc.) according to a specified random mechanism.
2. Updating the parameters of the random mechanism, typically parameters of pdfs (probability density functions), on the basis of the data, to produce a "better" sample in the next iteration.

Unlike most of the stochastic algorithms for optimization which are based on local search, the CE method is a global random search procedure. The CE method was successfully applied to various problems such as the traveling salesman problem [30], the bipartition problem [30], the maximal cut problem [32], the image matching [12], the image segmentation [13], etc.

For a comprehensive overview and history of the CE method, the reader is referred to [33]. For the sake of completion we present below the generic CE scheme for combinatorial optimization problems.

Consider the problem of minimizing the function $S$ over a finite set $\mathcal{X}$, say

$$\gamma^* = \min_{x \in \mathcal{X}} S(x). \tag{3}$$

The starting point in the methodology of the CE method applied to (3) is to associate an estimation problem with the optimization problem (3). To this end one defines a collection of indicator functions $I_{\{S(x) \leq \gamma\}}$ on $\mathcal{X}$ for various thresholds or levels $\gamma \in \mathbb{R}$. Next, let $\{f(\cdot; v), v \in V\}$ be a family of (discrete) probability density functions (pdfs) on $\mathcal{X}$, parameterized by a real-valued (vector) $v$.

For some $u \in V$, we consider the *Associated Stochastic Problem* (ASP):

$$\ell(\gamma) = \mathbb{P}_u(S(x) \leq \gamma) = \sum_{x \in \mathcal{X}} I_{\{S(x) \leq \gamma\}} f(x; u) = \mathbb{E}_u I_{\{S(x) \leq \gamma\}}, \tag{4}$$

where $\mathbb{P}_u$ is the probability measure under which the random state $\mathcal{X}$ has the pdf $f(\cdot; u)$, and $\mathbb{E}_u$ denotes the corresponding expectation operator. The idea of CE method is to construct simultaneously two sequences of levels $\widehat{\gamma}_1, \widehat{\gamma}_2, ..., \widehat{\gamma}_T$ and parameters (vectors) $\widehat{v}_1, \widehat{v}_2, ..., \widehat{v}_T$ such that $\widehat{\gamma}_T$ is close to the optimal $\gamma^*$, and $\widehat{v}_T$ is such that the corresponding density assigns high probability mass to the collection of states that give a low value. More specifically, one initializes by setting $v_0 = u$ and choosing a not very small quantity $\theta$, and then proceeds as follows:

1. **Adaptive updating of $\gamma_t$.** For a fixed $v_{t-1}$, let $\gamma_t$ be the $\theta$-quantile of $S(X)$ under $v_{t-1}$. That is, $\gamma_t$ satisfies

$$\mathbb{P}_{v_{t-1}}(S(X) \geq \gamma_t) \geq 1 - \theta, \tag{5}$$

$$\mathbb{P}_{v_{t-1}}(S(X) \leq \gamma_t) \geq \theta, \tag{6}$$

   where $X \sim f(\cdot; v_{t-1})$.
   A simple estimator of $\gamma_t$, denotes $\widehat{\gamma}_t$, can be obtained by drawing a random sample $X^1, X^2, ..., X^N$ from $f(\cdot; v_{t-1})$. Suppose that $S(X^{\sigma(1)}) \leq S(X^{\sigma(2)}) \leq ... \leq S(X^{\sigma(N)})$, where $\sigma$ is a permutation of the set $\{1, ..., N\}$. Estimate the $\theta$-quantile of $S(X)$ as

$$\widehat{\gamma}_t = S_{\lfloor \theta N \rfloor}. \tag{7}$$

2. **Adaptive updating of $v_t$.** For a fixed $\gamma_t$ and $v_{t-1}$, derive $v_t$ by minimizing the Kullback-Leibler distance, or equivalent to solving the program

$$\max_v \mathbb{E}_{v_{t-1}} I_{\{S(X) \leq \gamma_t\}} \ln f(X; v). \tag{8}$$

   The stochastic counterpart of (8) is as follows: for fixed $\widehat{\gamma}_t$ and $\widehat{v}_{t-1}$ the estimate of $v_{t-1}$, derive $\widehat{v}_t$ from the solution of following program

$$\max_v D(v) := \frac{1}{N} \sum_{i=1}^{N} I_{\{S(X^i) \leq \widehat{\gamma}_t\}} \ln f(X^i; v). \tag{9}$$

In typical applications, the function $D$ is concave and differentiable with respect to $v$, and thus updating equation (9) is equivalent to solving the following system of equations:

$$\frac{1}{N}\sum_{i=1}^{N} I_{\{S(X^i)\leq\gamma_t\}}\nabla \ln f(X^i;v)=0, \tag{10}$$

where the gradient is with respect to $v$.

For the convergence analysis of the CE method we refer to [10,11,33].

*Remark 1* (**smooth parameter**): Instead of updating the parameter $v$ directly via the solution of (9) we use the following smooth version

$$\widehat{v}_t=\alpha\widetilde{v}_t+(1-\alpha)\widehat{v}_{t-1}, t=1,2,..., \tag{11}$$

where $\widetilde{v}_t$ is the parameter vector from the solution of (9), and $\alpha$ is called the smoothing parameter, with $0.7\leq\alpha\leq 1$. The reason for using this smooth is: (a) to smooth out the value of $\widehat{v}_t$, (b) to reduce the probability that some component $\widehat{v}_{t,i}$ of $\widehat{v}_t$ will be zero or one at the first few iterations. This is particularly important when $\widehat{v}_t$ is a vector or matrix of probabilities.

**CE Algorithm for Combinatorial Optimization**

1. Choose $\widehat{v}_0$, and $0<\theta<1$. Set $t=1$.
2. Generate $N$ samples $X^1,X^2,...,X^N$ according to $f(\cdot;\widehat{v}_{t-1})$, and compute $\theta$-quantile $\widehat{\gamma}_t$ of $S$ according to (7).
3. Use the same samples $X^1,X^2,...,X^N$ to solve the stochastic programming problem (9). Denote the solution by $\widetilde{v}_t$.
4. Applying (11) to smooth out the vector $\widetilde{v}_t$.
5. If for some $t\geq d$, say $d=5$ such that

$$\widehat{\gamma}_t=\widehat{\gamma}_{t-1}=...=\widehat{\gamma}_{t-d},$$

then **stop**; otherwise set $t=t+1$, reiterate from step 2.

## 3 Application CE method to Problem (2)

We denote by $X$ the set of the sequences of injections $\pi=(\pi_1,\pi_2,...,\pi_{d-1})$, where $\pi_k:\{1,2,...,n_1\}\to\{1,2,...,n_{k+1}\}$ is an injection, $k=1,2,...,d-1$. The objective function of the MAP is computed by:

$$S(\pi)=\sum_{i=1}^{n_1} c_{i,\pi_1(i),...,\pi_{d-1}(i)}.$$

Due to the special structure of $X$, it is very hard to directly construct a family of discrete distributions on $X$. We consider an extending set of $X$, denoted $\widetilde{X}$, that is the set of the sequences of mappings $\widetilde{\pi}=(\widetilde{\pi}_1,\widetilde{\pi}_2,...,\widetilde{\pi}_{d-1})$, where $\widetilde{\pi}_k:\{1,2,...,n_1\}\to\{1,2,...,n_{k+1}\}$, $k=1,2,...,d-1$. We define a function $\widetilde{S}$ on $\widetilde{X}$ as follows

$$\widetilde{S}(\widetilde{\pi})=\begin{cases} S(\widetilde{\pi}) & \text{if } \widetilde{\pi}\in X,\\ +\infty & \text{otherwise.}\end{cases}$$

Therefore

$$\min_{\pi\in X} S(\pi) = \min_{\widetilde{\pi}\in\widetilde{X}} \widetilde{S}(\widetilde{\pi}).$$

To construct a family of discrete distribution on $\widetilde{X}$, we choose a sequence of probability matrices $M = (M_1, M_2, ..., M_{d-1})$, where

$$M_k = \begin{pmatrix} p_k(1|1) & p_k(1|2) & \cdots & p_k(1|n_{k+1}) \\ p_k(2|1) & p_k(2|2) & \cdots & p_k(2|n_{k+1}) \\ & & \cdots & \\ p_k(n_1|1) & p_k(n_1|2) & \cdots & p_k(n_1|n_{k+1}) \end{pmatrix},$$

and $p_k(i|j)$ represents the probability to assign the element $i$ of dimension 1 to the element $j$ of dimension $k+1$, for all $k = 1, ..., d-1$.

We have

$$\sum_{j=1}^{n_{k+1}} p_k(i|j) = 1.$$

For each $\widetilde{\pi} = (\widetilde{\pi}_1, \widetilde{\pi}_2, ..., \widetilde{\pi}_{d-1}) \in \widetilde{X}$. The probability of drawing this sequence of mappings according to $M$ is

$$p(\widetilde{\pi}) = \prod_{k=1}^{d-1} \prod_{i=1}^{n_1} p_k(i|\widetilde{\pi}_k(i)).$$

In each iteration, suppose that $\widetilde{\pi}^l = (\widetilde{\pi}_1^l, \widetilde{\pi}_2^l, ..., \widetilde{\pi}_{d-1}^l)$, $l = 1, 2, ..., N$ are the samples drawn. The $H = \lfloor \theta N \rfloor$ best samples, according to the objective function $\widetilde{S}$, are selected to update $M$. Denoting $\{\widetilde{\pi}^1, \widetilde{\pi}^2, ..., \widetilde{\pi}^H\}$ as the $H$ "best" samples among the draws $\{\widetilde{\pi}^1, \widetilde{\pi}^2, ..., \widetilde{\pi}^N\}$, minimizing the Kullback-Leibler distance leads to the following optimization problem:

$$\begin{cases} \max\limits_{p_k(i|j)} \mathcal{K} := \sum\limits_{h=1}^{H} \ln(\prod\limits_{k=1}^{d-1} \prod\limits_{i=1}^{n_1} p_k(i|\widetilde{\pi}_k^h(i))) \\ s.t. \quad \sum\limits_{j=1}^{n_{k+1}} p_k(i|j) = 1, i = 1, ..., n_1, k = 1, ..., d-1, \\ \qquad p_k(i|j) \geq 0, i = 1, ..., n_1, j = 1, .., n_{k+1}, k = 1, ..., d-1. \end{cases} \tag{12}$$

Solving this program is equivalent to solving $(d-1)$ following programs

$$\begin{cases} \max\limits_{p_k(i|j)} \mathcal{K}_k := \sum\limits_{h=1}^{H} \ln(\prod\limits_{i=1}^{n_1} p_k(i|\widetilde{\pi}_k^h(i))) \\ s.t. \quad \sum\limits_{j=1}^{n_{k+1}} p_k(i|j) = 1, i = 1, ..., n_1, \\ \qquad p_k(i|j) \geq 0, i = 1, ..., n_1, j = 1, .., n_{k+1}. \end{cases} \tag{13}$$

We first rewrite the objective function of (13) as follows

$$\begin{aligned} \mathcal{K}_k &= \sum_{h=1}^{H} \ln(\prod_{i=1}^{n_1} p_k(i|\widetilde{\pi}_k^h(i)) = \sum_{i=1}^{n_1} \sum_{h=1}^{H} \ln p_k(i|\widetilde{\pi}_k^h(i)) \\ &= \sum_{i=1}^{n_1} \sum_{j=1}^{n_{k+1}} \text{card}\{h \in \{1, ..., H\} : \widetilde{\pi}_k^h(i) = j\} \ln(p(i|j)). \end{aligned}$$

Denoting

$$u_{ij} = p(i|j), b_{ij} = \text{card}\{h \in \{1, ..., H\} : \ \widetilde{\pi}_k^h(i) = j\}.$$

The program (13) is equivalent to

$$\begin{cases} \min\limits_{u_{ij}}(-\sum\limits_{i=1}^{n_1}\sum\limits_{j=1}^{n_{k+1}} b_{ij}\ln(u_{ij})) \\ s.t. \quad \sum\limits_{j=1}^{n_{k+1}} u_{ij} = 1, i = 1, ..., n_1, \\ \qquad u_{ij} \geq 0, i = 1, ..., n_1, j = 1, .., n_{k+1}. \end{cases} \tag{14}$$

The Karusk-Kuhn-Tucker (KKT) condition of the convex program (14) is:

$$\begin{cases} -\frac{b_{ij}}{u_{ij}} + \lambda_i - \mu_{ij} = 0, i = 1, ..., n_1, j = 1, ..., n_{k+1}, \\ \lambda_i(\sum\limits_{j=1}^{n_{k+1}} u_{ij} - 1) = 0, i = 1, ..., n_1, \\ \mu_{ij}u_{ij} = 0, i = 1, ..., n_1, j = 1, ..., n_{k+1}, \\ \lambda_i \geq 0, i = 1, ..., n_1, \\ \mu_{ij} \geq 0, i = 1, ..., n_1, j = 1, ..., n_{k+1}. \end{cases} \tag{15}$$

Then by solving (15), we get the updating formula of matrix $M_k, k = 1, ..., d-1$ as follows

$$p_k(i|j) := \frac{\text{card}\{h \in \{1, ..., H\} : \ \widetilde{\pi}_k^h(i) = j\}}{H}.$$

**The CE algorithm for solving the Problem (2):**
**Step 1.** Initialize $M = (M_1, M_2, ..., M_{d-1})$, where $M_k = (p_k(i|j))_{n_1 \times n_{k+1}}, k = 1, ..., d-1$, are a uniform distribution, i.e.,

$$p_k(i|j) = \frac{1}{n_{k+1}}, i = 1, ..., n_1, j = 1, ..., n_{k+1},$$

and choose $\theta \in ]0, 1[$.
**Step 2.** Draw $N$ samples $\widetilde{\pi}^1, \widetilde{\pi}^2, ..., \widetilde{\pi}^N$ according to $M$. Compute $\widetilde{S}(\widetilde{\pi}^l), l = 1, 2, ..., N$.
**Step 3.** Sort the sequence $\widetilde{S}(\widetilde{\pi}^l)_{l=1}^N$ in the increasing orders. Let $\widetilde{S}(\widetilde{\pi}^{\sigma(1)}) \leq \widetilde{S}(\widetilde{\pi}^{\sigma(2)}) \leq ... \leq \widetilde{S}(\widetilde{\pi}^{\sigma(N)})$, where $\sigma$ is a permutation of the set $\{1, 2, ..., N\}$. Set $H = \lfloor \theta N \rfloor$, then choose $H$ best draws $\widetilde{\pi}^{\sigma(1)}, \widetilde{\pi}^{\sigma(2)}, ..., \widetilde{\pi}^{\sigma(H)}$.
**Step 4.** Update $M$ by the formula

$$p_k(i|j) := \frac{\text{card}\{h \in \{1, 2, ..., H\} : \ \widetilde{\pi}_k^{\sigma(h)}(i) = j\}}{H},$$

$$i = 1, ..., n_1, j = 1, ..., n_{k+1}, k = 1, ..., d-1.$$

**Step 5.** Applying (11) to smooth out $M$.
**Step 6.** Iterate step 2-5 until convergence.

*Remark 2* : In **Step 2**, if we use the sequence of matrices $M = (M_1, ..., M_{d-1})$ to generate directly the sequence of mappings $\widetilde{\pi} = (\widetilde{\pi}_1, ..., \widetilde{\pi}_{d-1})$, then the generated mapping $\widetilde{\pi}_k$ may not be an injection, in this case $\widetilde{S}(\widetilde{\pi})$ would be $+\infty$. For avoiding the generation of irrelevant mappings, we propose the following procedure.

**Generation of sequence of injections in the MAP**

For $k = 1, ..., d-1$:

**1.** Set $\widehat{M_k} = M_k$, i.e., $\widehat{p}_k(i|j) = p_k(i|j)$. Let $r = 1$.

**2.** Generate a value of random variable $X$ from the following distribution

$$\text{Prob}\{X = j\} = \widehat{p}_k(r|j), j = 1, ..., n_{k+1}.$$

Assume that $X$ receives value $j^*$. Define

$$\pi_k(r) = j^*.$$

**3.** If $(r < n_1)$ update $M_k^{(r)}$

for $i = r + 1$ to $n_1$
$\widehat{p}_k(i|j^*) = 0$;
$sum = 0$;
for $j = 1$ to $n_{k+1}$
$sum = sum + \widehat{p}_k(i|j)$;
end for
for $j = 1$ to $n_{k+1}$
$\widehat{p}_k(i|j) := \widehat{p}_k(i|j)/sum$;
end for
end for
Set $r = r + 1$, go to **2**.
else stop

## 4 Numerical Results

In this section, we test our CE algorithm on a set of 58 MAPs. In the first experiment, we use the procedure given in [16] to generate the data. By this procedure, we know a priori optimal solution of MAP that allows us to evaluate the efficiency of our CE algorithm. The problems are fully dense and the cost variables for individual problems are approximately normally distributed [9,16].

The CE algorithm is written in language C of Microsoft Visual C++ 2008, and is tested on a notebook with chipset Intel(R) Core(TM) Duo CPU 2.0 GHz, 3GB of RAM.

The following notations are used in these tables:

- *Prob*: the problem,
- *Samples*: the number of the samples $N$,
- *Smooth*: the smooth parameter $\alpha$,
- $MaxIter$: the maximal number of iteration of CE algorithm,
- $Vars$: the number of the binary variables,
- $Ctrs$: the number of the constraints,
- *Average*: the average for the 10 results,
- *Best*: the best result among the 10 results,
- $Time$: the average time of 10 tests,
- *Objective value*: the objective value given by the CE algorithm,
- *Gap* % $= \frac{\text{Objective value - Optimal value}}{\text{Optimal value}}$.

Table 1 gives the value of parameters of the CE algorithm for each problem. In the "Prob" column, we list the structure of test problems. For instance, n5d3 represents the MAP with dimension $d = 3$, and each dimension has $n = 5$ elements.

| Prob | Samples | $\theta$ | Smooth | MaxIter |
|---|---|---|---|---|
| n5d3 | 500 | 0.1 | 0.3 | 10 |
| n8d3 | 2000 | 0.01 | 0.3 | 10 |
| n10d3 | 5000 | 0.01 | 0.3 | 15 |
| n15d3 | 8000 | 0.01 | 0.3 | 30 |
| n20d3 | 8000 | 0.01 | 0.3 | 30 |
| n8d5 | 3000 | 0.01 | 0.3 | 20 |
| n10d5 | 5000 | 0.01 | 0.3 | 20 |
| n12d5 | 5000 | 0.01 | 0.3 | 30 |
| n15d5 | 8000 | 0.01 | 0.3 | 30 |
| n20d5 | 8000 | 0.01 | 0.3 | 30 |

**Table 1** Parameters.

| Prob | Vars | Ctrs | Optimal | Objective value | | Gap | | Time (s) |
|---|---|---|---|---|---|---|---|---|
| | | | | Average | Best | Average | Best | |
| 1 | 125 | 15 | 29 | 29.9 | 29 | 3.10 | 0.00 | 0.21 |
| 2 | 125 | 15 | 24 | 24.4 | 24 | 1.67 | 0.00 | 0.20 |
| 3 | 125 | 15 | 17 | 19.8 | 17 | 16.47 | 0.00 | 0.20 |
| 4 | 125 | 15 | 18 | 18.9 | 18 | 5.00 | 0.00 | 0.20 |
| 5 | 125 | 15 | 14 | 16.0 | 14 | 14.29 | 0.00 | 0.22 |
| 6 | 125 | 15 | 24 | 25.8 | 24 | 7.50 | 0.00 | 0.20 |
| 7 | 125 | 15 | 23 | 25.4 | 23 | 10.43 | 0.00 | 0.20 |
| 8 | 125 | 15 | 24 | 24.0 | 24 | 0.00 | 0.00 | 0.21 |
| 9 | 125 | 15 | 31 | 31.0 | 31 | 0.00 | 0.00 | 0.21 |
| 10 | 125 | 15 | 30 | 30.0 | 30 | 0.00 | 0.00 | 0.21 |

**Table 2** Results with d = 3, n = 5.

For each test problem in these tables, we run CE algorithm 10 times, then take the average result ("Average") and the best result ("Best") in term of objective value and gap.

Table 2, 3, 4, 5, 6 present the results with $d = 3$, and $n = 5, 8, 10, 15, 20$.

From the Table 2 and 3, the best run of the CE method found an optimal solution in almost problems, except the two last problems in the case $n = 3, d = 8$. Moreover, the CE algorithm is fast: CPU time is less than 0.22 (resp. 4.5) seconds in the case $n = 5$ (resp. $n = 8$).

In Table 4, 5, 6, the CE algorithm also produced good solutions. The best gap varies from 0.00% to 11.63% (resp. from 7.87% to 12.82%, from 11.71% to 17.78%) in the case $n = 10$ (resp. $n = 15, n = 20$).

In the second experiment, we consider the data given on the website [37] corresponding to [16], where the optimal value and the optimal assignment are furnished.

| Prob | Vars | Ctrs | Optimal | Objective value | | Gap | | Time (s) |
|---|---|---|---|---|---|---|---|---|
| | | | | Average | Best | Average | Best | |
| 1 | 512 | 24 | 57 | 63.6 | 57 | 11.58 | 0.00 | 3.03 |
| 2 | 512 | 24 | 40 | 44.7 | 40 | 11.75 | 0.00 | 3.27 |
| 3 | 512 | 24 | 49 | 54.3 | 49 | 10.82 | 0.00 | 3.20 |
| 4 | 512 | 24 | 41 | 45.3 | 41 | 10.49 | 0.00 | 3.87 |
| 5 | 512 | 24 | 36 | 41.0 | 36 | 13.89 | 0.00 | 4.25 |
| 6 | 512 | 24 | 40 | 45.1 | 40 | 12.75 | 0.00 | 4.38 |
| 7 | 512 | 24 | 43 | 46.9 | 43 | 9.07 | 0.00 | 4.39 |
| 8 | 512 | 24 | 62 | 65.9 | 62 | 6.29 | 0.00 | 4.42 |
| 9 | 512 | 24 | 45 | 50.2 | 49 | 11.56 | 8.89 | 4.18 |
| 10 | 512 | 24 | 47 | 52.2 | 48 | 11.06 | 2.13 | 2.05 |

**Table 3** Results with d = 3, n = 8.

| Prob | Vars | Ctrs | Optimal | Objective value | | Gap | | Time (s) |
|---|---|---|---|---|---|---|---|---|
| | | | | Average | Best | Average | Best | |
| 1 | 1000 | 30 | 54 | 61.1 | 58 | 13.15 | 7.41 | 10.92 |
| 2 | 1000 | 30 | 47 | 53.6 | 51 | 14.04 | 8.51 | 11.55 |
| 3 | 1000 | 30 | 70 | 76.0 | 72 | 8.57 | 2.86 | 10.92 |
| 4 | 1000 | 30 | 66 | 73.9 | 71 | 11.97 | 7.58 | 11.45 |
| 5 | 1000 | 30 | 43 | 50.4 | 48 | 17.21 | 11.63 | 10.98 |
| 6 | 1000 | 30 | 59 | 66.0 | 64 | 11.86 | 8.47 | 12.94 |
| 7 | 1000 | 30 | 42 | 49.2 | 46 | 17.14 | 9.52 | 10.97 |
| 8 | 1000 | 30 | 56 | 64.5 | 61 | 15.18 | 8.93 | 13.22 |
| 9 | 1000 | 30 | 51 | 56.6 | 51 | 10.98 | 0.00 | 11.41 |
| 10 | 1000 | 30 | 51 | 58.5 | 56 | 14.71 | 9.80 | 11.78 |

**Table 4** Results with d = 3, n = 10.

| Prob | Vars | Ctrs | Optimal | Objective value | | Gap | | Time (s) |
|---|---|---|---|---|---|---|---|---|
| | | | | Average | Best | Average | Best | |
| 1 | 3375 | 45 | 68 | 79.6 | 75 | 17.06 | 10.29 | 195.03 |
| 2 | 3375 | 45 | 79 | 92.6 | 89 | 17.22 | 12.66 | 201.44 |
| 3 | 3375 | 45 | 89 | 99.8 | 96 | 12.13 | 7.87 | 201.97 |
| 4 | 3375 | 45 | 81 | 93.8 | 90 | 15.80 | 11.11 | 197.92 |
| 5 | 3375 | 45 | 79 | 92.5 | 89 | 17.09 | 12.66 | 208.25 |
| 6 | 3375 | 45 | 71 | 85.4 | 80 | 20.28 | 12.68 | 214.77 |
| 7 | 3375 | 45 | 86 | 99.2 | 96 | 15.35 | 11.63 | 211.83 |
| 8 | 3375 | 45 | 78 | 92.4 | 88 | 18.46 | 12.82 | 214.14 |
| 9 | 3375 | 45 | 83 | 96.1 | 92 | 15.78 | 10.84 | 213.78 |
| 10 | 3375 | 45 | 72 | 85.7 | 79 | 19.03 | 9.72 | 190.41 |

**Table 5** Results with d = 3, n = 15.

| Prob | Vars | Ctrs | Optimal | Objective value | | Gap | | Time (s) |
|---|---|---|---|---|---|---|---|---|
| | | | | Average | Best | Average | Best | |
| 1 | 8000 | 60 | 109 | 131.2 | 127 | 20.37 | 16.51 | 315.08 |
| 2 | 8000 | 60 | 118 | 140.2 | 135 | 18.81 | 14.41 | 315.08 |
| 3 | 8000 | 60 | 97 | 117.1 | 112 | 20.72 | 15.46 | 334.59 |
| 4 | 8000 | 60 | 103 | 123.6 | 119 | 20.00 | 15.53 | 341.13 |
| 5 | 8000 | 60 | 119 | 139.4 | 134 | 17.14 | 12.61 | 324.33 |
| 6 | 8000 | 60 | 111 | 129.1 | 124 | 16.31 | 11.71 | 335.38 |
| 7 | 8000 | 60 | 90 | 108.4 | 106 | 20.44 | 17.78 | 339.08 |
| 8 | 8000 | 60 | 107 | 126.7 | 122 | 18.41 | 14.02 | 356.17 |
| 9 | 8000 | 60 | 113 | 134.9 | 131 | 19.38 | 15.93 | 346.45 |
| 10 | 8000 | 60 | 100 | 120.0 | 116 | 20.00 | 16.00 | 226.31 |

**Table 6** Results with d = 3, n = 20.

The results are reported in Table 7 with $d = 5$, and $n = 8, 10, 12, 15, 20$ respectively. We use the same notations as in the previous tables.

We observe from the numerical results that the CE algorithm produced the good solutions in acceptable time. The best gap varies from 3.23% to 11.54%. On the other hand, the difference between the best gap and the average gap is from 3.83% to 6.73%, that shows the stability of our CE method.

| Prob | Vars | Ctrs | Optimal | Objective value | | Gap | | Time (s) |
|---|---|---|---|---|---|---|---|---|
| | | | | Average | Best | Average | Best | |
| n8d5prob1 | 32768 | 40 | 31 | 33.6 | 32 | 8.39 | 3.23 | 10.66 |
| n8d5prob2 | 32768 | 40 | 38 | 41.8 | 40 | 10.00 | 5.26 | 11.03 |
| n10d5prob1 | 100000 | 50 | 47 | 52.8 | 51 | 12.34 | 8.51 | 42.20 |
| n10d5prob2 | 100000 | 50 | 52 | 57.2 | 55 | 9.81 | 5.77 | 60.63 |
| n12d5prob1 | 248832 | 60 | 56 | 62.4 | 60 | 11.43 | 7.14 | 118.39 |
| n12d5prob2 | 248832 | 60 | 53 | 61.1 | 59 | 15.28 | 11.32 | 127.55 |
| n15d5prob1 | 759375 | 75 | 64 | 74.1 | 70 | 15.78 | 9.38 | 340.66 |
| n20d5prob1 | 3200000 | 100 | 104 | 123 | 116 | 18.27 | 11.54 | 718.04 |

**Table 7** Results with d = 5.

## 5 Conclusion

In this paper, we have proposed a new and efficient approach based on the CE method for solving the Multidimensional Assignment Problem. Exploiting the formulation (2) of the MAP, we constructed a family of discrete distributions such that the CE method could be applied. The computational results have shown the efficiency of our proposed approach. It found a near-optimal solution within an acceptable time for even large-scale problems. Our CE algorithm solves the MAP up to 3.2 millions binary variables

and 100 constraints, while the existing methods, for instance the GRASP with Path Relinking [1] and the simulated annealing [9], the maximal numbers of binary variables of the MAP are 287,496 and 1.0 million respectively. In the future work, we plan to combine our CE algorithm with a local method to improve the result.

## References


1. Aiex RM, Resende MGC, Pardalos PM, Toraldo G (2005) GRASP with Path Relinking for Three-Index Assignment. INFORMS J. on Computing 17(2):224-247.
2. Andrijich SM, Caccetta L (2001) Solving the multisensor data association problem. Nonlinear Analysis 47:5525-5536.
3. Balas E, Saltzman MJ (1991) An Algorithm for the Three-Index Assignment Problem. Operations Research 39(1):150-161.
4. Bandelt HJ, Maas A, Spieksma FCR (2004) Local Search Heuristics for Multi-Index Assignment Problems with Decomposable Costs. The Journal of the Operational Research Society 55(7):694-704.
5. Burkard RE, Çela E (1997) Quadratic and three-dimensional assignment problems. In: M. Dell'Amico, F. Maffioli, S. Martello (ed) Annotated Bibliographies in Combinatorial Optimization. Wiley, Chichester, pp. 373-392.
6. Burkard RE, Çela E (1999) Linear Assignment Problems and extensions. In: Du D, Pardalos PM (ed) Handbook of Combinatorial Optimization. Kluwer Academic, Dordrecht, Chap. 21, pp. 75-149.
7. de Boer PT, Kroese DP, Mannor S, Rubinstein RY (2005) A Tutorial on The Cross-Entropy Method. Annals of Operations Research 134:19-67.
8. Çela E (2002) Assignment problems. In: Pardalos PM, Resende MGC (ed) Handbook of Applied Optimization, pp. 661-678. Oxford University Press, New York. Chap. 17.9.
9. Clemons W, Grundel D, Jeffcoat D (2003) Applying simulated annealing on the multidimensional assignment problem, In: Proceeding of the 2nd Cooperative Control and Optimization Conference.
10. Costa A, Dafydd O, Kroese D (2007) Convergence properties of the cross-entropy method for discrete optimization. Operations Research Letters 35(5):573-580.
11. Dambreville F (2006) Cross-entropy method: convergence issues for extended implementation. In: Proceedings of the rare event simulation conference (RESIM 2006), Bamberg, Germany.
12. Dubin U (2002) The cross-entropy method for combinatorial optimization with applications. Master's thesis, The Technion, Israel Institute of Technology, Haifa.
13. Dubin U (2004) Application of the cross-entropy method for image segmentation. Annals of Operations Research, 2004. Submitted.
14. Gilbert KC, Hofstra RB (1988) Multidimensional assignment problems, Decision Sciences 19(2):306-321.
15. Grundel D, Oliveira CA, Pardalos PM, Pasiliao E (2005) Asymptotic Results for Random Multidimensional Assignment Problems. Comput. Optim. Appl. 31(3):275-293.
16. Grundel D, Pardalos PM (2005) Test Problem Generator for the Multidimensional Assignment Problem. Comput. Optim. Appl. 30(2):133-146.
17. Gutin G, Karapetyan D (2009) Local Search Heuristics for the Multidimensional Assignment Problem. In Graph Theory, Computational Intelligence and Thought, Lipshteyn M, Levit VE, RM Mcconnell RM (Eds.). Lecture Notes In Computer Science, Vol. 5420. Springer-Verlag, Berlin, Heidelberg 100-115
18. Kuroki Y, Matsui T (2009) An approximation algorithm for multidimensional assignment problems minimizing the sum of squared errors. Discrete Applied Mathematics 157(9):2124-2135.
19. Murphey R, Pardalos PM, Pitsoulis L (1998) A greedy randomized adaptive search procedure for the multitarget multisensor tracking problem. American Mathematical Society, DIMACSSeries 40:277-302.
20. Oliveira CAS, Pardalos PM (2004) Randomized parallel algorithms for the multidimensional assignment problem. Appl. Numer. Math. 49(1):117-133.
21. Pardalos PM, Pitsoulit LS (ed) (2000) Nonlinear Assignement Problems: Algorithms and Applications. Combinatorial Optimization, vol. 7, Kluwer Academic, Dordrecht.

22. Pasiliao EL (2003) Algorithms for Multidimensional Assignment Problems. Ph.D. thesis, Department of Industrial and Systems Engineering, University of Florida.
23. Pasiliao EL (2010) Local Neighborhoods for the Multidimensional Assignment Problem. In: Hirsch MJ, Pardalos PM, Murphey R (ed) Dynamics of Information Systems. Springer New York, 40:353-371.
24. Pierskalla W (1968) The multidimensional assignment problem. Operations Research 16:422-431.
25. Poore AB, Rijavec N (1993) A Lagrangian Relaxation Algorithm for Multidimensional Assignment Problems Arising from Multitarget Tracking. SIAM Journal of Optimization 3(3):544-563.
26. Poore AB (1994) Multidimensional assignment formulation of data association problems arising from multitarget and multisensor tracking. Computation Optimization and Applications 3:27-54.
27. Poore AB, Robertson III AJ (1997) A New Lagrangian Relaxation Based Algorithm for a Class of Multidimensional Assignment Problems. Computational Optimization and Applications 8(2):129-150.
28. Pusztaszeri JF, Rensing PE, Liebling TM (1996) Tracking elementary particles near their primary vertex: acombinatorial approach. Journal of Global Optimization 9(1):41-64.
29. Rubinstein RY (1997) Optimization of cumputer simulation models with rare events. European Journal of Operation Reseach 99:89-112.
30. Rubinstein RY (1999) The simulated entropy method for combinatorial and continuous optimization. Methodology and Computing in Applied Probability 2:127-190.
31. Rubinstein RY (2001) Combinatorial optimization, cross-entropy, ants and rare events. In: Uryasev S, Pardalos PM (ed) Stochastic Optimization: Algorithms and Application. Kluwer, pp. 304-358.
32. Rubinstein RY (2002) The cross-entropy method and rare-events for maximal cut and bipartition problems. ACM Transactions on Modelling and Computer Simulation 12(1):27-53.
33. Rubinstein RY, Kroese D (2004) The cross-entropy method: a unified approach to combinatorial optimization, Monté Carlo simulation, and machine learning. Springer, Berlin.
34. Spieksma FCR (2000) Multi Index Assignment Problems: Complexity, Approximation, Applications. In: Pardalos PM, Pitsoulit LS (ed) Nonlinear Assignement Problems: Algorithms and Applications. Combinatorial Optimization, vol. 7, pp. 1-12. Kluwer Academic, Dordrecht (2000). Chap. 1.
35. Storms PPA, Spieksma FCR (2003) An LP-based algorithm for the data association problem in multitarget tracking. Computers & Operations Research 30(7):1067-1085.
36. Veenman CJ, Hendriks EA, Reinders MJT (1998) A fast and robust point tracking algorithm. In: Proceedings of the Fifth IEEE International Conferenceon Image Processing, 653-657, Chicago.
37. http://camo.ici.ro/hco.htm.

# Value-at-Risk constrained Optimization

**Journal of Global Optimization manuscript No.**
(will be inserted by the editor)

# A fast and scalable Cross-Entropy Method for Value-at-Risk constrained Optimization

**Le Thi Hoai An, Nguyen Duc Manh, Pham Dinh Tao**

Received: date / Accepted: date

**Abstract** In this paper, we consider a portfolio selection problem that consists of maximizing a return distribution under a Value-at-Risk constraint. It is a nonconvex nonsmooth optimization problem which is very hard to solve. We propose an approach based on the Cross-Entropy (CE) method to tackle it. The numerical results show the efficiency of our approach
**Keywords**: Portfolio Optimization, Risk Management, Value at Risk, Cross-Entropy.

## 1 Introduction

Value-at-Risk, denoted *VaR*, is an important and popular risk measure that has been used extensively in recent years in portfolio selection and in risk analysis (see e.g. [12]). Let $X$ be the anticipated random returns, then the Value-at-Risk of $X$ is defined as

$$VaR_\alpha(X) = \inf\{u : F_X(u) \geq \alpha\} = F_X^{-1}(\alpha), \;\; 0 < \alpha < 1,$$

where $F_X$ is the distribution function of $X$. $VaR_\alpha$ is said to be an acceptability functional [15].

The *VaR* has the undesirable mathematical characteristics such as a lack of subadditivity and concavity [2,3]. Due to the non-concavity of *VaR*, the problem of maximizing Value-at-Risk or maximizing the return under a Value-at-Risk constrain is nonconvex. That is why, in practice *VaR* is often replaced by the Average Value-at-Risk (*AVaR*, also called Conditional Value-at-Risk, *CVaR*). Some reasons justifying the use of *CVaR* instead of *VaR* can be found in [25]. However, due to regulatory frameworks such as Basel II and Solvency II, Value-at-Risk remains to be widely used in portfolio management in most notably banks.

Le Thi Hoai An
Laboratory of Theoretical and Applied Computer Science (LITA),
Paul Verlaine University of Metz, Ile du Saulcy, F-57045 Metz Cedex, France
E-mail: lethi@univ-metz.fr

Nguyen Duc Manh and Pham Dinh Tao
LMI, National Institute for Applied Sciences (INSA) - Rouen,
BP08-Avenue de l'Université, 76801, Saint-Etienne-du-Rouvray, France
E-mail: nguyen@insa-rouen.fr, pham@insa-rouen.fr

In this paper, we consider the following non-convex portfolio selection under the Value-at-Risk constraint:

$$\begin{cases} \max\ \mathbb{E}(x^T\xi) \\ \text{s.t.}\ \sum_{i=1}^{n} x_i = 1, \\ \quad x_i \geq 0, 1 \leq i \leq n, \\ \quad VaR_\alpha(x^T\xi) \geq a, \end{cases} \tag{1}$$

where $x_i$ denotes the relative weight of asset $i$ in portfolio, $\xi_i$ is the random return of asset $i$, and $n$ is the number of assets. Because of the non-convexity of Value-at-Risk constrain, this problem is NP-hard (NP-complete in the strong sense) [4].

There are several approaches including deterministic and heuristic to solve the problem (1) in the literature (see e.g. [4–6, 9–11, 14] or [24] for an overview). Recently, Wozabal et al. [24] gave a representation of the $VaR$ as a DC function (Difference of Convex functions) in the case finite scenarios, and proposed a conical Branch-and-Bound algorithm to find global optima of (1). Later, Wozabal [25] introduced a DC Algorithm (DCA) to the DC formulation of the problem (1). To obtain this DC formulation, the author used a penalty technique that is not exact. In [25] it has been not proved that the DC formulation is equivalent to the original problem.

In this paper, we propose an algorithm based on the Cross-Entropy (CE) method to tackle Problem (1). The CE method was motivated by an adaptive algorithm for estimating probabilities of rare events in complex stochastic networks [19], which involves variance minimization. It was soon realized [20, 21] that a simple cross-entropy modification of [19] could be used not only for estimating probabilities of rare event but for solving difficult combinatorial optimization problems as well. This is done by translating the "deterministic" optimization problem into a related "stochastic" optimization problem and then using rare event simulation technique similar to [19]. Several recent applications demonstrate the power of the CE method as a generic and practical tool for solving NP-hard problems. The crucial point in applying CE method is how to find a family of pdfs (probability density functions) on the feasible set of the optimization problem such that updating the parameters could be done as easily as possible. Due to the special structure of the feasible set of problem (1), we will construct a family of pdfs based on a family of exponential pdfs for our CE algorithm. The numerical results demonstrate that our proposed algorithm finds a near-optimal solution. It is a fast and scalable algorithm.

The rest of paper is organized as follows. In Section 2, we give a short presentation of the CE method. In Section 3, we propose an application of CE for our problem based on a family of exponential pdfs. Results of numerical experiments are reported in Section 4 while some conclusions and perspectives are discussed in Section 5.

## 2 The Cross-Entropy method

The Cross - Entropy (CE) method [7, 17, 19–21] has been developed by Rubinstein initially for evaluating rare events probabilities, for which a direct computation by usual methods would be unreliable. The only way to evaluate them is then to resort to a simulation method based on importance sampling. The CE method allows to tilt proposed densities in order to favor sampling of rare events. It has been demonstrated that this method is particularly relevant for solving "hard" optimization problems like combinatorial optimization problems. Indeed, when deterministic methods failed to find the optimal solution within a reasonable amount of computation, in most cases the CE method allows to find a fairly good one

more quickly. To use the CE method for solving a deterministic optimization problem, this problem must be first translated into a stochastic one. The set of feasible solutions is then regarded as a set of events subjected to an importance density. Thus, we use rare event simulation technique.

Suppose that we wish to maximize a real-valued performance function $S$ over a set $\mathscr{X}$. Let us denote the maximum by $\gamma^*$

$$\gamma^* = \max_{x \in \mathscr{X}} S(x). \tag{2}$$

The starting point in CE method is to associate an estimation problem with the optimization problem (2). To this end we define a collection of indicator functions $I_{\{S(x) \leq \gamma\}}$ on $\mathscr{X}$ for family of (discrete) probability density functions (pdfs) on $\mathscr{X}$, parameterized by a real-valued (vector) $v$.

For a certain $u \in V$, we consider the *associated stochastic problem* (ASP):

$$\ell(\gamma) = \mathbf{P}_u(S(x) \geq \gamma) = \sum_{x \in \mathscr{X}} I_{\{S(x) \geq \gamma\}} f(x;u) = \mathbf{E}_u I_{\{S(x) \geq \gamma\}},$$

where $\mathbf{P}_u$ is the probability measure under which the random state $\mathscr{X}$ has the pdf $f(\cdot;u)$, and $\mathbf{E}_u$ denotes the corresponding expectation operator. The idea of CE method is to construct simultaneously a sequence of levels $\widehat{\gamma}_1, \widehat{\gamma}_2, ..., \widehat{\gamma}_T$ and parameters (vectors) $\widehat{v}_1, \widehat{v}_2, ..., \widehat{v}_T$ such that $\widehat{\gamma}_T$ is close to the optimal $\gamma^*$ and $\widehat{v}_T$ is such that the corresponding density assigns high probability mass to the collection of states that give a high value. More specifically, we initialize by setting $v_0 = u$, choosing a not very small quantity $\theta$, and then we proceed as follows:

1. **Adaptive updating of $\gamma_t$.** For a fixed $v_{t-1}$, let $\gamma_t$ be the $(1-\theta)$-quantile of $S(X)$ under $v_{t-1}$. That is, $\gamma_t$ satisfies

$$\mathbf{P}_{v_{t-1}}(S(X) \geq \gamma_t) \geq \theta, \tag{3}$$

$$\mathbf{P}_{v_{t-1}}(S(X) \leq \gamma_t) \geq 1-\theta, \tag{4}$$

   where $X \sim f(\cdot; v_{t-1})$.
   A simple estimator of $\gamma_t$, denote $\widehat{\gamma}_t$ can be obtained by drawing a random sample $X^1, X^2, ..., X^N$ from $f(\cdot; v_{t-1})$. Suppose that $S(X^{\sigma(1)}) \leq S(X^{\sigma(2)}) \leq ... \leq S(X^{\sigma(N)})$, where $\sigma$ is a permutation of the set $\{1, ..., N\}$. Evaluating the $(1-\theta)$-quantile of $S(X)$ as

$$\widehat{\gamma}_t = S_{\lfloor (1-\theta)N \rfloor}. \tag{5}$$

2. **Adaptive updating of $v_t$.** For a fixed $\gamma_t$ and $v_{t-1}$, derive $v_t$ by minimizing the Kullback-Leibler distance, or equivalent to solving the following program:

$$\max_v \mathbf{E}_{v_{t-1}} I_{\{S(X) \geq \gamma_t\}} W(X;u,v_{t-1}) \ln f(X;v), \tag{6}$$

   where

$$W(x;u,v_{t-1}) = \frac{f(x;u)}{f(x;v_{t-1})}.$$

   The stochastic counterpart of (6) is as follows: for fixed $\widehat{\gamma}_t$ and $\widehat{v}_{t-1}$ (the estimate of $v_{t-1}$), derive $\widehat{v}_t$ from the following program:

$$\max_v D(v) := \frac{1}{N} \sum_{i=1}^{N} I_{\{S(X^i) \geq \gamma_t\}} W(X^i;u,v_{t-1}) \ln f(X^i;v). \tag{7}$$

In typical applications, the function $D$ is concave and differentiable with respect to $v$, and thus the updating equation (7) is equivalent to solving the following system of equations:

$$\frac{1}{N}\sum_{i=1}^{N} I_{\{S(X^i)\geq \widehat{\gamma}_t\}} W(X^i;u,v_{t-1})\nabla \ln f(X^i;v) = 0, \tag{8}$$

where the gradient is with respect to $v$.

*Remark 1* (**Smoothing parameter**): instead of updating the parameter $v$ directly via the solution of (7) we use the following smoothed version

$$\widehat{v}_t = \beta \widetilde{v}_t + (1-\beta)\widehat{v}_{t-1}, t = 1,2,..., \tag{9}$$

where $\widetilde{v}_t$ is the parameter vector from the solution of (7), and $\beta$ is called the smoothing parameter, with $0.7 \leq \beta \leq 1$.

**Generic CE Algorithm for Optimization**

1. Choose $\widehat{v}_0$, and $0 < \theta < 1$. Set $t = 1$.
2. Generate $N$ samples $X^1, X^2, ..., X^N$ according to $f(\cdot;\widehat{v}_{t-1})$, and compute $(1-\theta)$-quantile $\widehat{\gamma}_t$ of $S$ according to (5).
3. Using the same samples $X^1, X^2, ..., X^N$ solve the stochastic programming problem (7). Denote the solution by $\widetilde{v}_t$.
4. Apply (9) to smooth out the vector $\widetilde{v}_t$.
5. Repeat step 2-4 until a pre-specified stopping criterion is met.

## 3 A CE algorithm for solving Problem (1)

Let $X$ be the set defined by

$$X = \{x = (x_1, x_2, ..., x_n) \in \mathbb{R}^n : \sum_{i=1}^{n} x_i = 1,\ x_i \geq 0, i = 1,2,...,n\}.$$

To apply CE, we will construct a family of pdfs $\{f(\cdot;v), v \in V\}$ on $X$ and then use the penalty technique to treat the Value-at-Risk constraint of Problem (1) as follows

$$\text{if } x \in X, VaR_\alpha(x^T\xi) < a \text{ then } S(x) = 0,$$

where $S(x) = \mathbb{E}(x^T\xi)$ is the objective function of (1). Here, we assume that the distribution of $\xi$ allows us to compute $S(x)$ and $VaR_\alpha(x^T\xi)$ uncomplicatedly, (for instance, in the case of finite scenario - a very common case in practice).

A direct construction of a "natural" family of pdfs $\{f(\cdot;v), v \in V\}$ on $X$ is in general difficult. Thus, instead of the set $X$, we consider the following set

$$\Omega = \{x = (x_1, x_2, ..., x_{n-1}) \in \mathbb{R}^{n-1} : \sum_{i=1}^{n-1} x_i \leq 1,\ x_i \geq 0, i = 1,2,...,n-1\}.$$

There exists a diffeomorphism $P$ from $X$ to $\Omega$ as follows

$$x = (x_1, x_2, ..., x_n) \in X \mapsto P(x) = (x_1, x_2, ..., x_{n-1}) \in \Omega. \tag{10}$$

Therefore, having a family of pdfs $\{f(\cdot;v), v \in V\}$ on $\Omega$ is equivalent to having a family of pdfs on $X$. The reason to choose $\Omega$ is that we have a rich choice of the "natural" family of pdfs on it. In this paper, we choose a family of pdfs as follows.

Firstly, we consider the exponential distribution on $\mathbb{R}^n_+$

$$f(x;v) = \exp(-\sum_{i=1}^{n} \frac{x_i}{v_i}) \prod_{i=1}^{n} \frac{1}{v_i}, \quad x \in \mathbb{R}^n_+,$$

where $v = (v_1, v_2, ..., v_n) \in \mathbb{R}^n_+$ is the parameter. Thus, we have

$$\int_{\mathbb{R}^n_+} f(x;v)dx = 1, \quad \forall v \in \mathbb{R}^n_+.$$

By considering the diffeomorphism $H$ from $\Omega \times (0,+\infty)$ to $\mathbb{R}^n_+$

$$\begin{aligned} H: \quad & \Omega \times (0,+\infty) \to \mathbb{R}^n_+ \\ & (y_1, ..., y_{n-1}, t) \mapsto (x_1, x_2, ..., x_n), \\ & \begin{cases} x_i = t y_i, \quad i = 1, 2, ..., n-1, \\ x_n = t(1 - y_1 - ... - y_{n-1}), \end{cases} \end{aligned}$$

we have a transformation of variables for the above integral.

The Jacobian matrix of the function $H$ is given by

$$J_H(y,t) = \begin{pmatrix} t & 0 & 0 & \dots & 0 & y_1 \\ 0 & t & 0 & \dots & 0 & y_2 \\ & & & \ddots & & \\ 0 & 0 & 0 & \dots & t & y_{n-1} \\ -t & -t & -t & \dots & -t & 1 - \sum_{i=1}^{n-1} y_i \end{pmatrix}.$$

It is clear to see that $\det(J_H(y,t)) = t^{n-1}$. Thus, we have

$$\begin{aligned} 1 &= \int_{\mathbb{R}^n_+} f(x;v)dx = \int_{\Omega} \left( \int_0^{+\infty} f((y,t);v) t^{n-1} dt \right) dy \\ &= \frac{1}{\prod_{i=1}^{n} v_i} \int_{\Omega} \left( \int_0^{+\infty} \exp\left( -t \sum_{i=1}^{n-1} \frac{y_i}{v_i} - \frac{t(1 - y_1 - ... - y_{n-1})}{v_n} \right) t^{n-1} dt \right) dy \\ &= \int_{\Omega} \frac{1}{\prod_{i=1}^{n} v_i} \cdot \frac{(n-1)!}{\left( \sum_{i=1}^{n-1} \frac{y_i}{v_i} - \frac{(1 - y_1 - ... - y_{n-1})}{v_n} \right)^n} dy. \end{aligned}$$

Now, we get a family of probability measures on $\Omega$ with the pdfs

$$g(x;v) = \frac{1}{\prod_{i=1}^{n} v_i} \cdot \frac{(n-1)!}{\left( \sum_{i=1}^{n-1} \frac{x_i}{v_i} - \frac{(1 - x_1 - ... - x_{n-1})}{v_n} \right)^n}, \quad x = (x_1, ..., x_{n-1}). \tag{11}$$

Therefore, by using the map $P$, we have a family of pdfs on $X$ as follows

$$g(x;v) = \frac{1}{\prod_{i=1}^{n} v_i} \cdot \frac{(n-1)!}{\left(\frac{x_1}{v_1} + \frac{x_2}{v_2} + \ldots + \frac{x_n}{v_n}\right)^n}, \quad x = (x_1, \ldots, x_{n-1}, x_n) \in X. \tag{12}$$

**Updating the parameter $v_t$**

We have

$$\ln g(x;v) = \sum_{i=1}^{n-1} \ln i - n \ln\left(\sum_{i=1}^{n} \frac{x_i}{v_i}\right) - \sum_{i=1}^{n} \ln v_i.$$

Thus

$$\frac{\partial}{\partial v_j} \ln g(x;v) = \frac{1}{v_j^2} \left( \frac{n x_j}{\sum_{i=1}^{n} \frac{x_i}{v_i}} - v_j \right) \quad j = 1, 2, \ldots, n.$$

In our case, we can solve the system of equation (8) to update the parameter $v$:

$$\sum_{i=1}^{N} I_{\{S(X_i) \geq \gamma\}} W(X^i; u, v_{t-1}) \left( \frac{n x_j}{\sum_{i=1}^{n} \frac{x_i}{v_i}} - v_j \right) = 0, \quad j = 1, 2, \ldots, n,$$

therefore

$$v_j = \frac{\sum_{i=1}^{N} I_{\{S(X_i) \geq \gamma\}} W(X^i; u, v_{t-1}).n.X_{ij}}{\sum_{i=1}^{N} I_{\{S(X_i) \geq \gamma\}} W(X^i; u, v_{t-1}). \sum_{i=1}^{n} \frac{X_{ij}}{v_j}}, \tag{13}$$

where

$$X^i = (X_{i1}, X_{i2}, \ldots, X_{in}) \in X, \quad i = 1, 2, \ldots, N.$$

Set

$$a_j = \sum_{i=1}^{N} I_{\{S(X_i) \geq \gamma\}} W(X^i; u, v_{t-1}) X_{ij}, \quad j = 1, 2, \ldots, n.$$

The system of equations (13) becomes

$$v_j = \frac{n a_j}{\sum_{j=1}^{n} \frac{a_j}{v_j}}, \quad j = 1, 2, \ldots, n,$$

therefore

$$\sum_{j=1}^{n} \frac{a_j}{v_j} = n \frac{a_j}{v_j}, \quad j = 1, 2, \ldots, n.$$

Thus

$$\frac{a_1}{v_1} = \frac{a_2}{v_2} = \ldots = \frac{a_n}{v_n} = c = const,$$

or

$$v_j = \frac{a_j}{c}, j = 1, 2, \ldots, n.$$

We only consider $v = (v_1, v_2, ..., v_n)$ satisfying

$$v_j > 0, j = 1,2,...,n \text{ and } \sum_{j=1}^{n} v_j = 1.$$

Thus

$$\begin{aligned} c &= \sum_{j=1}^{n} a_j = \sum_{j=1}^{n} \sum_{i=1}^{N} I_{\{S(X_i)\geq\gamma\}} W(X^i; u, v_{t-1}) X_{ij} \\ &= \sum_{i=1}^{N} I_{\{S(X_i)\geq\gamma\}} W(X^i; u, v_{t-1}) \sum_{j=1}^{n} X_{ij} \\ &= \sum_{i=1}^{N} I_{\{S(X_i)\geq\gamma\}} W(X^i; u, v_{t-1}). \end{aligned}$$

Hence, we get the formula to update $v$:

$$\widehat{v}_j^t = \frac{\sum_{i=1}^{N} I_{\{S(X_i)\geq\gamma\}} W(X^i; u, v_{t-1}) X_{ij}}{\sum_{i=1}^{N} I_{\{S(X_i)\geq\gamma\}} W(X^i; u, v_{t-1})}, j = 1,2,...,n. \tag{14}$$

In practice, we can generate the samples on $X$ by $g(\cdot; v)$ as follows:

1. Generate the sample $x = (x_1, x_2, ..., x_n) \in \mathbb{R}_+^n$ by $f(\cdot; v)$.
2. If $x_1 + x_2 + ... + x_n > 0$ then take $y = (y_1, y_2, ..., y_n) \in X$, where

$$y_i = \frac{x_i}{x_1 + x_2 + ... + x_n}, i = 1,2,...,n.$$

Finally our CE algorithm for solving Problem (1) can be described as follows.

**The CE algorithm for solving Problem (1):**

**Step 1.** Initialize $v_0 = u = (\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n})$, and $\theta \in (0,1)$, $VM = [\emptyset]$, $\varepsilon > 0$.
**Step 2.** Draw $N$ samples $X^1, X^2, ..., X^N$ according to $g(x; v_t)$. Compute $S(X^k), k = 1,2,...,N$. Sort the sequence $\{S(X^k)\}_{k=1}^N$ in the increasing order. Suppose that $S(X^{\sigma(1)}) \leq S(X^{\sigma(2)}) \leq ... \leq S(X^{\sigma(N)})$, where $\sigma$ is a permutation of the set $\{1,2,...,N\}$. Set $H = \lfloor (1-\theta)N \rfloor$, then choose $H$ best draws $X^{\sigma(H)}, X^{\sigma(H+1)}, ..., X^{\sigma(N)}$.
**Step 3.** Update $v_{t+1}$ by the formula (14) and the smoothed updating (9).
**Step 4.** Set $M = \frac{1}{N-H+1} \sum_{i=H}^{N} S(X^{\sigma(i)}), VM = [VM; M], \Delta_t = std(VM)$, where $std(V)$ computes the sample standard deviation of the data in $V$.
**Step 5.** Iterate step 2-5 until $\Delta_t < \varepsilon$.

Here, $M$ is the average of the best values $\{S(X^k)\}_{k=H}^N$, and $VM$ in iteration $k$ is a vector which contains $M$ at all iterations less or equal $k$. In practice, in each iteration we should store the "best" value of the sequence $\{S(X^k)\}_{k=1}^N$, i.e., $S(X^{\sigma(N)})$, to get the "best" value when the algorithm stops.

## 4 Numerical experiments

The algorithm described above is tested on real-world financial market data. The code is written in MATLAB 2007a, and is tested on a notebook with chipset Intel(R) Core(TM) Duo CPU 2.0 GHz, 3GB of RAM.

Our experiment is composed of two parts. In the first part we study the globality of the CE algorithm by comparing the solutions given by the CE with an optimal solution furnished by the software CPLEX 12.1 applied to the mixed-integer formulation of the problem (1) (see [4]). In the second experiment we consider large datasets (for which CPLEX can not find global solutions) and compare our method with a heuristic algorithm developed in [13].

### 4.1 The globality of the CE algorithm: comparison with global optima

We used the empirical distribution of two years of weekly data, i.e., $S = 104$ scenarios of the following 5 indices: CAC 40, Standard&Poors 100, Nasdaq 100, FTSE 100, and Hang Seng. We take $\alpha = 0.045$. Different values of $a$ are considered in the dataset ($a$ is incremented by 0.0005). It is easy to see that if $a$ is smaller than the $VaR$ of the portfolio that consists only of the asset with the highest return, then the problem is trivial. Hence, in our test problems, $a$ is greater than that $VaR$.

| Name | Average Return | Variance | $VaR_{0.045}$ | $AVaR_{0.045}$ |
|---|---|---|---|---|
| CAC 40 | 1.0027144 | 0.000217647 | 0.9742 | 0.9694 |
| Standard&Poors100 | 1.0004326 | 0.000179629 | 0.9755 | 0.9717 |
| Nasdaq 100 | 1.0013597 | 0.000471718 | 0.9653 | 0.9504 |
| FTSE 100 | 1.0021901 | 0.000151213 | 0.9824 | 0.9797 |
| Hang Seng | 1.0016546 | 0.000421260 | 0.9624 | 0.9561 |

**Table 1** Time frame: 2004-2005, weekly data

The parameters in the CE method are taken as follows: the number of samples $N = 2000$, $\theta = 0.01$, the parameter smooth $\beta = 0.8$, and the number of iterations is limited to 20.

The results are presented in Table (2) where we compare the objective value (say, the expected return) obtained by our CE algorithm and the optimal value given by CPLEX software.

The Figure 2 shows the efficient frontier traced by each algorithm that illustrates the globality of the CE algorithm in almost all cases. We observe from the numerical results that the CE algorithms furnished an $\varepsilon$-optimal solution with $\varepsilon$ between $7.52 * 10^{-08}$ and $6.26 * 10^{-05}$. Moreover the CE algorithm is fast: CPU time is less than 2.5 seconds.

### 4.2 The scability of the CE algorithm

To study the scability of the CE method we first test on the 1304 daily returns for 43 assets of the Eurostoxx50 index from January 1, 2003 to December 31, 2007, which gave positive returns. Then, we test on the empirical distribution of 11 years (from January 1, 2000 to

| a | CE method | | | | Optimal solution | | | Error |
|---|---|---|---|---|---|---|---|---|
| | Expected Return | $VaR_{0.045}$ | Time(s) | Delta | Expected Return | $VaR_{0.045}$ | Time (s) | |
| 0.9745 | 1.00270419 | 0.974502 | 2.26 | 7.13E-06 | 1.00270427 | 0.974500 | 0.49 | 7.52E-08 |
| 0.9750 | 1.00268620 | 0.975001 | 2.39 | 3.13E-07 | 1.00268702 | 0.975000 | 0.58 | 8.19E-07 |
| 0.9755 | 1.00266935 | 0.975500 | 2.26 | 6.04E-07 | 1.00266978 | 0.975500 | 0.41 | 4.27E-07 |
| 0.9760 | 1.00265231 | 0.976002 | 2.31 | 6.09E-07 | 1.00265254 | 0.976000 | 0.47 | 2.33E-07 |
| 0.9765 | 1.00263104 | 0.976528 | 2.37 | 1.51E-06 | 1.00263307 | 0.976500 | 0.39 | 2.03E-06 |
| 0.9770 | 1.00260890 | 0.977058 | 2.29 | 1.77E-06 | 1.00261218 | 0.977000 | 0.39 | 3.28E-06 |
| 0.9775 | 1.00258806 | 0.977524 | 2.36 | 1.56E-06 | 1.00259129 | 0.977500 | 0.42 | 3.23E-06 |
| 0.9780 | 1.00256581 | 0.978067 | 2.42 | 2.15E-06 | 1.00257040 | 0.978000 | 0.38 | 4.59E-06 |
| 0.9785 | 1.00254669 | 0.978545 | 2.39 | 2.01E-06 | 1.00254951 | 0.978500 | 0.34 | 2.82E-06 |
| 0.9790 | 1.00252463 | 0.979010 | 2.25 | 2.27E-06 | 1.00252862 | 0.979000 | 0.36 | 3.98E-06 |
| 0.9795 | 1.00250197 | 0.979522 | 2.23 | 1.28E-05 | 1.00250773 | 0.979500 | 0.36 | 5.76E-06 |
| 0.9800 | 1.00242424 | 0.980006 | 2.33 | 2.06E-05 | 1.00248684 | 0.980000 | 0.34 | 6.26E-05 |
| 0.9805 | 1.00230423 | 0.980501 | 2.26 | 9.05E-07 | 1.00230464 | 0.980500 | 0.38 | 4.15E-07 |
| 0.9810 | 1.00228115 | 0.981023 | 2.25 | 3.10E-06 | 1.00228322 | 0.981000 | 0.50 | 2.07E-06 |
| 0.9815 | 1.00226396 | 0.981501 | 2.31 | 1.82E-06 | 1.00226662 | 0.981500 | 0.53 | 2.66E-06 |
| 0.9820 | 1.00224391 | 0.982037 | 2.19 | 2.78E-06 | 1.00224655 | 0.982000 | 0.53 | 2.64E-06 |
| 0.9825 | 1.00222225 | 0.982528 | 2.34 | 4.53E-06 | 1.00222647 | 0.982500 | 0.59 | 4.21E-06 |
| 0.9830 | 1.00167853 | 0.983036 | 2.31 | 0.239363 | 1.00173056 | 0.983000 | 0.70 | 5.20E-05 |
| 0.9835 | 1.00155414 | 0.983539 | 2.22 | 0.015037 | 1.00160072 | 0.983500 | 0.89 | 4.66E-05 |

**Table 2** The performance of the CE algorithm compared with CPLEX solver

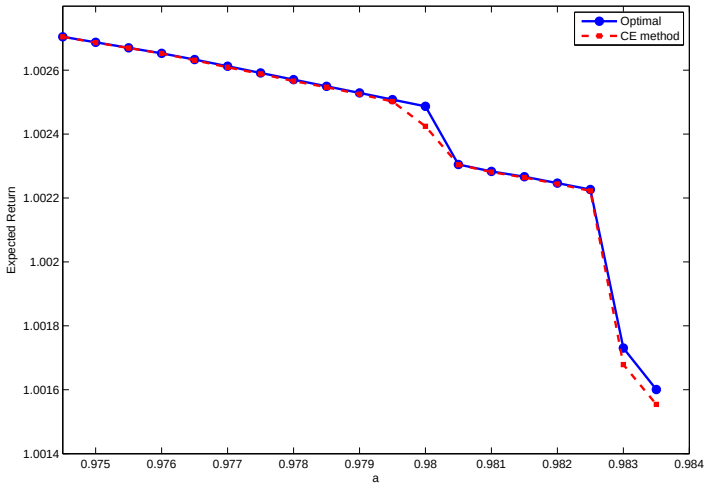


**Fig. 1** The efficient frontier: the CE method and Optima

December 31, 2010) of daily data, i.e., 2759 scenarios of the 87 assets comprising the NYSE US 100 index.

For these datasets, CPLEX furnished only a feasible solution which is not a global solution. We then compare our CE algorithm with an efficient existing method developed in [13], called Algorithm A1. This method is widely used in the industry for the Value-at-Risk portfolio selection. It solves the following problem:

$$\begin{cases} \min & VaR_{1-\alpha}(x^T\xi) \\ \text{s.t.} & x \in [0,1]^n, \sum_{i=1}^{n} x_i = 1 \\ & \mathbb{E}(x^T\xi) \geq C, \end{cases} \tag{15}$$

where $C$ is a given lower bound expectation.

Note that in [13] $X$ represents the random losses. Hence minimizing $VaR_{1-\alpha}$ amounts to maximizing $VaR_\alpha$. The main idea of Algorithm A1 is the approximation of the $VaR$ by $CVaR$ and iterative application of discarding scenarios with large losses. This algorithm performs well even on large datasets. Algorithm A1 from [13] adapted to our problem is described as follows ([25]):

1. Fix $C$ as a lower bound on expectation, a parameter for the Value-at-Risk $\alpha$ and a parameter for the heuristic $0 < \zeta < 1$.
2. Set $\alpha_0 = \alpha, i_0 = 0$, and $k = 0$.
3. Solve the problem

$$\begin{cases} \max\limits_{x} & AVaR_{\alpha_k}(\sum_{i=i_k}^{S} x^T\xi^i) \\ \text{s.t.} & \sum\limits_{i=1}^{n} x_i = 1, \\ & \mathbb{E}(x^T\xi) \geq C, \\ & x^T\xi^i \leq \gamma, \;\; i \leq i_k, \\ & x^T\xi^i \geq \gamma, \;\; i > i_k, \\ & x_i \geq 0, \;\; 1 \leq i \leq n. \end{cases} \tag{16}$$

4. Call the solution of the above problem $x_k$ and sort the scenarios $\xi^i$ according to their returns $r_i = x_k^T\xi^i$.
5. Set $k = k+1, b_k = \alpha + (1-\alpha)(1-\zeta)^k, i_k = \lfloor S(1-b_k) \rfloor$, and $\alpha_k = 1 - \dfrac{1-\alpha}{b_k}$.
6. If $i_k \leq \lfloor S/\alpha \rfloor$ goto step 3, otherwise exit.

The comparison between our CE algorithm and the A1 algorithm has proceeded as follows: for each value $a$, we solve the problem (1) by the CE algorithm and obtain an expected return and the corresponding Value-at-Risk. That expected return becomes lower bound expectation (say, the value of $C$) for the A1 algorithm. The A1 algorithm will produce a new Value-at-Risk, and then we compare the Value-at-Risk obtained by the two methods.

We run the CE method with the number of samples $N = 2000$, $\theta = 0.1$, the smoothing parameter $\beta = 0.8$ and the number of iteration is limited to 50. We choose the parameter $\zeta = 0.5$ for the A1 algorithm.

Table 3 represents the results corresponding to Eurostoxx50 data, when $a$ varies from 0.972 to 0.985. The lower bound for $a$ is the $VaR_{0.05}$ of the portfolio that consists only of the asset with the highest return (i.e., AXA with $VaR_{0.05}$ of 0.9713956 and expected daily return 1.002344), and the upper bound is 0 .985, because the CE algorithm does not work

| a | CE method | | | A1 algorithm | | Difference |
|---|---|---|---|---|---|---|
| | Expected Return | $VaR_{0.05}(CE)$ | Time(s) | $VaR_{0.05}(A1)$ | Time(s) | VaR |
| 0.972 | 1.00230523 | 0.972034 | 31.76 | 0.972112 | 57.89 | 0.000078 |
| 0.973 | 1.00224968 | 0.973001 | 30.81 | 0.973372 | 57.80 | 0.000372 |
| 0.974 | 1.00217375 | 0.974043 | 30.64 | 0.974555 | 57.97 | 0.000512 |
| 0.975 | 1.00205663 | 0.975048 | 30.43 | 0.976237 | 58.45 | 0.001189 |
| 0.976 | 1.00190228 | 0.976163 | 31.07 | 0.978605 | 62.20 | 0.002442 |
| 0.977 | 1.00178848 | 0.977255 | 30.79 | 0.979443 | 62.44 | 0.002188 |
| 0.978 | 1.00174446 | 0.978191 | 33.94 | 0.980305 | 68.31 | 0.002114 |
| 0.979 | 1.00161818 | 0.979144 | 34.94 | 0.981539 | 88.55 | 0.002395 |
| 0.980 | 1.00151780 | 0.980039 | 35.67 | 0.982554 | 69.74 | 0.002515 |
| 0.981 | 1.00144481 | 0.981269 | 38.65 | 0.983351 | 85.02 | 0.002082 |
| 0.982 | 1.00139708 | 0.982213 | 37.45 | 0.984068 | 62.45 | 0.001855 |
| 0.983 | 1.00137637 | 0.983122 | 42.65 | 0.983897 | 76.00 | 0.000775 |
| 0.984 | 1.00131175 | 0.984109 | 30.42 | 0.984358 | 67.00 | 0.000249 |
| 0.985 | 1.00091841 | 0.985104 | 44.35 | 0.988501 | 57.00 | 0.003397 |

**Table 3** The performance of CE algorithm compared with the A1 algorithm, Eurostoxx50 daily data

anymore for a greater value. In this table, for each value $a$, we solve the problem (1) by the CE to get an objective value that becomes lower bound expectation for the A1 algorithm.

The Figure 2 shows the efficient frontier traced by each algorithm while the Figure 3 shows their CPU times.

From the Table 3 and the Figure 2 we see that the difference of $VaR_{0.05}$ of solutions between two methods is small, it varies from $0.000078$ to $0.002515$. Additionally, the CE algorithm still runs very fast: CPU time varies from 30 to 44 seconds. It is much faster than the A1 algorithm. The ratio of CPU time between the two algorithms varies from $1.3$ to $2.24$

Tables 4 and 5 represent, respectively, the results (objective function and CPU time) of the CE algorithm and the A1 algorithm in case $\alpha = 0.05$ and $\alpha = 0.1$, for the data of NYSE US 100. Also, the curves in figures 4 and 5 (resp. figures 6 and 7) show the efficient frontier and CPU time of each algorithm in case $\alpha = 0.05$ (resp. $\alpha = 0.1$).

Here, $a$ varies from $0.953$ to $0.983$ when $\alpha = 0.05$, and from $0.971$ to $0.988$ when $\alpha = 0.1$. The lower bound for $a$ is the $VaR$ of the portfolio that consists only of the asset with the highest return (i.e., AIG with $VaR_{0.05}$ of $0.95263158$, $VaR_{0.1}$ of $0.97037850$, and expected daily return $1.00518861$).

We observe from the numerical results that the difference of $VaR_{0.05}$ (resp. $VaR_{0.1}$) given by the two methods varies from $0.000039$ to $0.004869$ (resp. from $0.000352$ to $0.003048$). On the other hand, the proposed CE is much faster than the A1 algorithm: the ratio of CPU time between the two algorithms varies from $2.75$ to $3.78$ in Table 4 and from $2.53$ to $3.29$ in Table 5. Moreover, the larger the sample size $N$ we use, the better the solution we obtain, and the more stable the CE algorithm works.

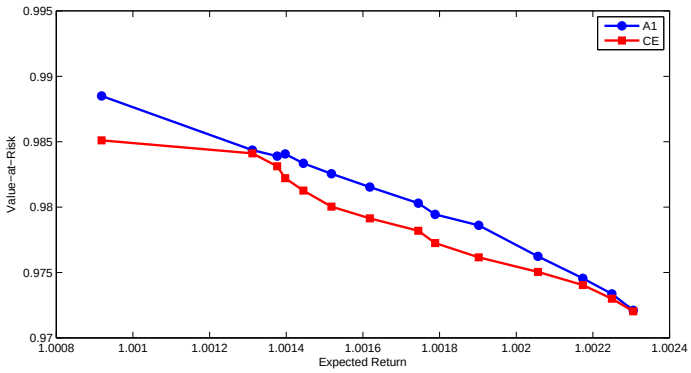


**Fig. 2** Efficient frontier for Eurostoxx50 data

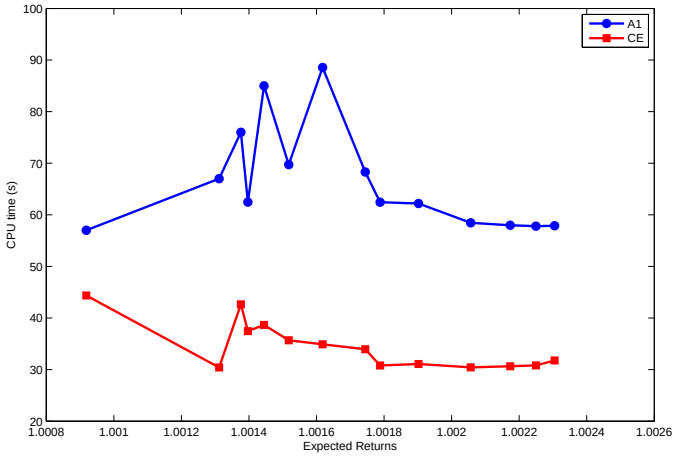


**Fig. 3** CPU time: Eurostoxx50 data

| a | CE method | | | A1 algorithm | | Difference |
|---|---|---|---|---|---|---|
| | Expected Return | $VaR_{0.05}(CE)$ | Time(s) | $VaR_{0.05}(A1)$ | Time(s) | |
| 0.953 | 1.00516360 | 0.953000 | 191.16 | 0.953039 | 646.13 | 0.000039 |
| 0.955 | 1.00494747 | 0.955093 | 229.84 | 0.955462 | 727.56 | 0.000368 |
| 0.957 | 1.00468419 | 0.957047 | 249.14 | 0.957465 | 862.25 | 0.000418 |
| 0.959 | 1.00430391 | 0.959184 | 242.67 | 0.960987 | 710.94 | 0.001802 |
| 0.961 | 1.00398844 | 0.961062 | 258.80 | 0.963480 | 736.92 | 0.002419 |
| 0.963 | 1.00377939 | 0.963250 | 234.31 | 0.965365 | 735.95 | 0.002115 |
| 0.965 | 1.00354935 | 0.965161 | 234.33 | 0.968078 | 862.31 | 0.002917 |
| 0.967 | 1.00327660 | 0.967272 | 233.41 | 0.969814 | 747.13 | 0.002542 |
| 0.969 | 1.00304977 | 0.969002 | 233.14 | 0.971765 | 745.59 | 0.002763 |
| 0.971 | 1.00264867 | 0.971137 | 231.77 | 0.975143 | 874.66 | 0.004006 |
| 0.973 | 1.00227647 | 0.973088 | 230.47 | 0.977629 | 755.27 | 0.004541 |
| 0.975 | 1.00192505 | 0.975047 | 229.78 | 0.979916 | 765.66 | 0.004869 |
| 0.977 | 1.00168743 | 0.977045 | 240.27 | 0.981061 | 670.20 | 0.004015 |
| 0.979 | 1.00149955 | 0.979033 | 230.66 | 0.982378 | 770.92 | 0.003346 |
| 0.981 | 1.00121198 | 0.981043 | 229.72 | 0.983926 | 773.48 | 0.002882 |
| 0.983 | 1.00088400 | 0.983163 | 226.50 | 0.985883 | 779.64 | 0.002720 |

**Table 4** NYSE US 100 daily data with $\alpha = 0.05$.

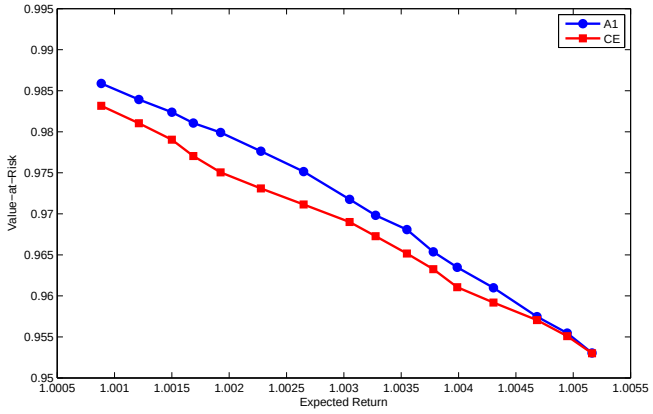


**Fig. 4** Efficient frontier: NYSE US 100 daily data with $\alpha = 0.05$.

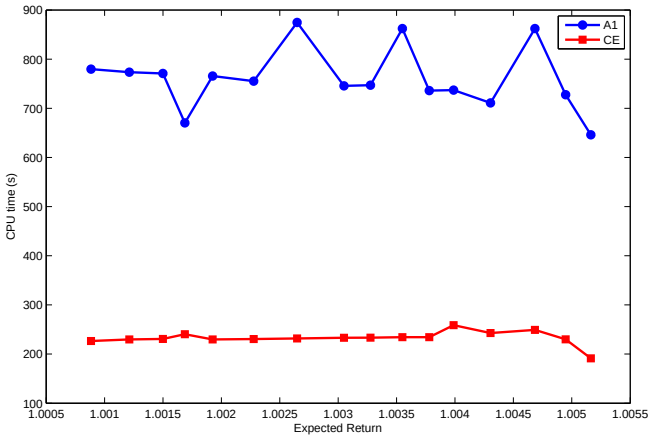


**Fig. 5** CPU time: NYSE US 100 daily data with $\alpha = 0.05$.

| a | CE method | | | A1 algorithm | | Difference |
|---|---|---|---|---|---|---|
| | Expected Return | $VaR_{0.1}(CE)$ | Time(s) | $VaR_{0.1}(A1)$ | Time (s) | |
| 0.971 | 1.00505703 | 0.971038 | 249.45 | 0.971390 | 819.36 | 0.000352 |
| 0.972 | 1.00489083 | 0.972007 | 258.28 | 0.972466 | 878.23 | 0.000459 |
| 0.973 | 1.00470412 | 0.973110 | 264.47 | 0.973609 | 769.77 | 0.000500 |
| 0.974 | 1.00447797 | 0.974020 | 274.28 | 0.975321 | 761.50 | 0.001301 |
| 0.975 | 1.00424399 | 0.975024 | 254.22 | 0.976104 | 996.36 | 0.001080 |
| 0.976 | 1.00401883 | 0.976018 | 267.45 | 0.977502 | 737.53 | 0.001484 |
| 0.977 | 1.00378923 | 0.977045 | 260.13 | 0.978636 | 853.59 | 0.001591 |
| 0.978 | 1.00352334 | 0.978022 | 268.81 | 0.979688 | 849.47 | 0.001665 |
| 0.979 | 1.00333735 | 0.979009 | 257.64 | 0.980898 | 698.95 | 0.001890 |
| 0.980 | 1.00307247 | 0.980060 | 272.19 | 0.982067 | 620.45 | 0.002007 |
| 0.981 | 1.00284515 | 0.981052 | 275.66 | 0.983053 | 701.24 | 0.002001 |
| 0.982 | 1.00251888 | 0.982145 | 273.27 | 0.984617 | 691.67 | 0.002471 |
| 0.983 | 1.00223072 | 0.983015 | 256.48 | 0.985813 | 711.09 | 0.002798 |
| 0.984 | 1.00186933 | 0.984177 | 270.23 | 0.987225 | 766.99 | 0.003048 |
| 0.985 | 1.00167539 | 0.985171 | 252.91 | 0.987890 | 772.53 | 0.002719 |
| 0.986 | 1.00156141 | 0.986435 | 267.56 | 0.988408 | 776.99 | 0.001973 |
| 0.987 | 1.00138666 | 0.987110 | 248.14 | 0.988752 | 703.80 | 0.001642 |
| 0.988 | 1.00118943 | 0.988237 | 259.52 | 0.989365 | 687.17 | 0.001128 |

**Table 5** NYSE US 100 daily data with $\alpha = 0.1$.

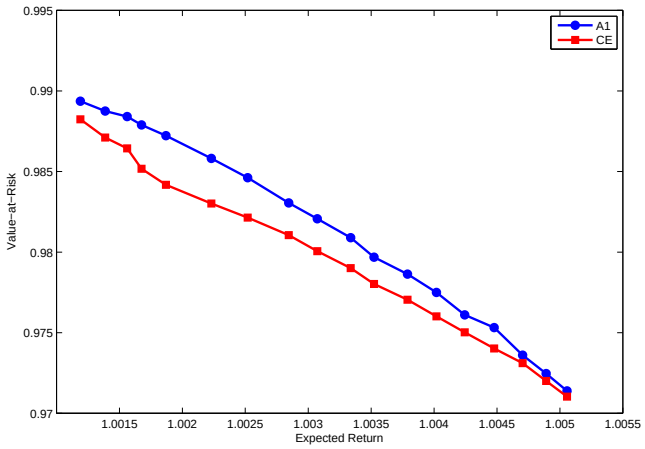


**Fig. 6** Efficient frontier: NYSE US 100 daily data with $\alpha = 0.1$.

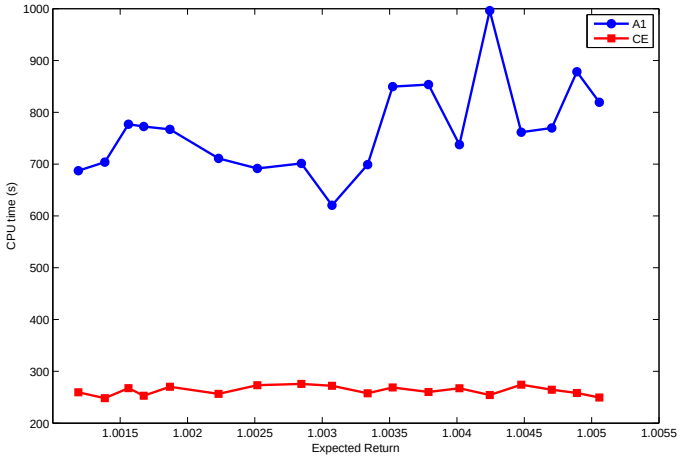


**Fig. 7** CPU time: NYSE US 100 daily data with $\alpha = 0.1$.

## 5 Conclusion

In this paper, we have proposed a new and efficient heuristic approach based on the Cross-Entropy method for solving a Value-at-Risk constrained optimization problem. Due to the special structure of the feasible set of problem (1), we have introduced an appropriate family of exponential pdfs, and developed a fast and scalable CE algorithm. Although the theoretical convergence properties of the CE method are not yet fully understood, the computational results in Table 2 and 3 show the efficiency of the proposed approach. It found a near-optimal solution within an acceptable time for large scale problems. We plan to combine our CE algorithm with other approaches for globally solving Problem (1), where CE can be used to find good feasible solutions.

## References


1. F. Andersson, H. Mausser, D. Rosen, and S. Uryasev, Credit risk optimization with conditional value-at-risk criterion, Mathematical Programming, 89(2001), pp. 273-291.
2. Ph. Artzner, F. Delbaen, J-M. Eber, and D. Heath, Thinking coherently, Risk, 10(11) (1997),pp.68-71.
3. Ph. Artzner, F. Delbaen, J-M. Eber, and D. Heath, Coherent measures of risk, Mathematical Finance,9(1999), pp.203-228.
4. S. Benati and R. Rizzi, A mixed integer linear programming formulation of the optimal mean/value-at-risk portfolio problem, European Journal of Operational Research, 176 (2007), pp. 423-434.
5. R. Campbell, R. Huisman, and K. Koedijk, Optimal portfolio selection in avalue-at-risk framework, Journal of Banking & Finance, 25(2001), pp. 1789-1804.
6. M. S. Cheon, S. Ahmed, and F. Al-Khayyal, A branch-reduced-cut algorithm for the global optimization of probabilistically constrained linear programs, Mathematical Programming, 108 (2006), pp. 617-634.
7. Costa A, Dafydd O, Kroese D, Convergence properties of the cross-entropy method for discrete optimization, Operations Research Letters 2007, 35(5); 57380.
8. Dambreville F, Cross-entropy method: convergence issues for extended implementation. In: Proceedings of the rare event simulation conference (RESIM 2006), Bamberg, Germany, 2006.
9. M. Gilli and E. Kellezi, A global optimization heuristic for portfolio choice with VaR and expected shortfall,in Computational methods in decision-making, economics and finance, vol. 74 of Applied Optimization, Kluwer, 2002, pp. 167-183.
10. M. Gilli, E.Kellezi,and H. Hysi, A data-driven optimization heuristic for downside risk minimization, The Journal of Risk, 8(2006), pp. 1-18.
11. R. Hochreiter, An evolutionary computation approach to scenario-based risk-return portfolio optimization for general risk measures, in EvoWorkshops 2007, vol. 4448 of Lecture Notes in Computer Science, Springer, 2007, pp. 199-207.
12. P. Jorion, Value at Risk: The New Benchmark for Controlling Market Risk, McGraw-Hill, 2000.
13. Larsen, N., Mausser, H., Uryasev, S.: Algorithms for optimization of value-at-risk. In: Pardalos, P., Tsitsiringos, V. (eds.) Financial Engineering, e-Commerce and Supply Chain, pp. 129-157. Kluwer Academic Publishers, Dordrecht, Netherlands(2002).
14. J. S. Pang and S. Leyffer, On the global minimization of the value-at-risk, Optimization Methods and Software, 19(2004), pp. 611-631.
15. G. Ch. Pflug and W. Romisch, Modeling, Measuring and Managing Risk, World Scientific, Singapore, 2007.
16. G. Ch. Pflug, Some remarks on the Value-at-Risk and the Conditional Value-at-Risk, in Probabilistic constrained optimization, vol. 49 of Nonconvex Optimization and its Applications, Kluwer, 2000, pp. 272-281.
17. Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor and Reuven Y. Rubinstein, A Tutorial on The Cross-Entropy Method, Annals of Operations Research 134, 19 – 67, 2005.
18. R. T. Rockafellar and S. Uryasev, Optimization of Conditional Value-at-Risk, The Journal of Risk, 2(2000), pp. 21-41.
19. Rubinstein RY, Optimization of cumputer simulation models with rare events, European Journal of Operation Reseach, 99, 89-112; 1997.
20. Rubinstein RY, The simulated entropy method for combinatorial and continuous optimization, Methodology and Computing in Applied Probability, 2, 127-190; 1999.

21. Rubinstein RY, Combinatorial optimization, cross-entropy, ants and rare events. In S. Uryasev and P. M. Pardalos, editors, Stochastic Optimization: Algorithms and Application, pages 304-358, Kluwer, 2001.
22. Rubinstein RY, Kroese D, The cross-entropy method: a unified approach to combinatorial optimization, Monté Carlo simulation, and machine learning, Berlin: Springer; 2004.
23. S. Uryasev, Conditional Value-at-Risk: Optimization algorithms and applications, Financial Engineering News, 14 (2000), pp.1-5.
24. Wozabal, D., Hochreiter, R., Pflug, G. : A d. c. formulation of value-at-risk constrained optimization. Tech. Rep. TR2008-01, Department of Statistics and Decision Support Systems,University of Vienna, Vienna(2008).
25. Wozabal, D., Value-at-Risk optimization using the difference of convex algorithm, OR Spectrum, Online First, 9 September 2010.

# A new method for Value-at-Risk constrained Optimization using the DC programming and DCA

LE THI Hoai An, PHAM DINH Tao, NGUYEN Duc Manh

**Abstract**

In this paper, we consider a portfolio optimization problem with a Value-at-Risk constraint. It is a nonconvex nonsmooth optimization problem, which is very hard to solve. We firstly reformulate this problem as a polyhedral DC program by using an exact penalty technique, and then propose DCA (DC algorithm) for its solution. The numerical results show the efficiency of our approach, especially in large scale setting.

**Keywords**: Portfolio Optimization, Risk Management, Value at Risk, DC programming and DCA.

## 1 Introduction

Value-at-Risk, denoted VaR, is an important and popular risk measure that has been used extensively in recent years in portfolio selection and in risk analysis (see e.g. [9]). Let $X$ be the anticipated random returns, then the Value-at-Risk of $X$ is defined as

$$VaR_\alpha(X) = \inf\{u : F_X(u) \geq \alpha\} = F_X^{-1}(\alpha), \quad 0 < \alpha < 1,$$

where $F_X$ is the distribution function of $X$. $VaR_\alpha$ is said to be an acceptability functional [15].

The VaR has the undesirable mathematical characteristics such as a lack of subadditivity and concavity [1, 2]. Due to the non-concavity of VaR, the problem of maximizing Value-at-Risk or maximizing the return under a Value-at-Risk constrain is nonconvex. That is why, in practice VaR is often replaced by the Average Value-at-Risk (AVaR, also called Conditional Value-at-Risk, CVaR), which is defined as

$$AVaR_\alpha(X) = \frac{1}{\alpha}\int_0^\alpha F_X^{-1}(t)dt = \frac{1}{\alpha}\int_0^\alpha VaR(t)dt, \quad 0 < \alpha \leq 1.$$

Some reasons justifying the use of CVaR instead of VaR can be found in [21]. However, due to regulatory frameworks such as Basel II and Solvency II, Value-at-Risk remains to be widely used in portfolio management in most notably banks.

In this paper, we consider the following non-convex portfolio selection under the Value-at-Risk constraint:

$$\begin{cases} \max\ \mathbb{E}(w^T\xi) \\ \text{s.t. } \sum_{i=1}^{m} w_i = 1, \\ \quad w_i \in [a_i, b_i], 1 \le i \le m, \\ \quad VaR_\alpha(w^T\xi) \ge a, \end{cases} \tag{1}$$

where $w_i$ denotes the relative weight of asset $i$ in the portfolio, $\xi_i$ the random return of asset $i$, $w^T\xi = \sum_{i=1}^{m} w_i\xi_i$, and $m$ is the number of assets. Because of the non-convexity of Value-at-Risk constraint, this problem is NP-hard (NP-complete in the strong sense) [3].

There are several approaches including deterministic and heuristic to solve the problem (1) in the literature (see e.g. [3, 4, 5, 6, 7, 8, 14] or [21] for an overview). Recently, Wozabal et al. [21] gave a representation of the $VaR$ as a DC function (Difference of Convex functions) in the case finite scenarios, and proposed a conical Branch-and-Bound algorithm to find global optima of (1). Later, Wozabal [22] introduced a DC Algorithm (DCA) [11, 17, 18] to the DC formulation of the problem (1). To obtain this DC formulation, the author used a penalty technique that is not exact. In [22] it has been not proved that the DC formulation is equivalent to the original problem.

In this work we propose a new approach based on DC programming and DCA for solving this problem. DC programming and DCA ([11, 17, 18] and references therein) aim to solve a general DC program that takes the form

$$\alpha = \inf\{f(x) := g(x) - h(x) : x \in \mathbb{R}^p\} \quad (P_{dc}) \tag{2}$$

where $g, h$ are lower semicontinuous proper convex functions on $\mathbb{R}^p$. Such a function $f$ is called DC function, and $g - h$, DC decomposition of $f$ while $g$ and $h$ are DC components of $f$. The construction of DCA involves DC components $g$ and $h$ but not the function $f$ itself: each iteration $k$ of DCA consists of computing

$$y^k \in \partial h(x^k), x^{k+1} \in \arg\min\{g(x) - h(x^k) - \langle x - x^k, y^k\rangle : x \in \mathbb{R}^p\} \quad (P_k).$$

Hence, for a DC program, each DC decomposition corresponds to a different version of DCA. Since a DC function $f$ has an infinite number of DC decompositions which have crucial impacts on the qualities (speed of convergence, robustness, efficiency, globality of computed solutions,...) of DCA, the search for a "good" DC decomposition is important from algorithmic point of views. Moreover, despite its local character, DCA with a good initial point could converge to global solutions. Finding a "good" initial point is then also an important stage of DCA. How to develop an efficient algorithm based on the generic DCA scheme for a practical problem is thus a judicious question to be studied, and the answer depends on the specific structure of the problem being considered. Due to the representation of the $VaR$ as a DC function [21], we reformulate the problem (1) as a polyhedral DC

program based on an exact penalty technique and apply DCA to find its solution.

This paper is organized as follows. In Section 2, we reformulate the problem (1) as a polyhedral DC program. In Sections 3, we propose an application of DCA to the problem. Numerical experiments are reported in Section 4 while some conclusions and perspectives are discussed in Section 5.

# 2 Reformulation of the problem (1)

## 2.1 Reformulation of $VaR_\alpha$ as a DC function.

In this part, we recall how to reformulate $VaR_\alpha$ as a DC function (see [21] for more details). Let $m$ be number of available budget. We need the two following assumptions.

**Assumption 1** *The distribution of the random asset returns $\xi = (\xi_1, ..., \xi_m)$ is discrete with finite atoms, i.e. there are $S \in N$ scenarios for the joint realizations of the random variables $\xi_i$. The $(m \times 1)$ vector of realizations of $\xi^s$ in scenario $s$ will be denoted by $\xi^s$ and the probability of the scenario will be denoted by $p_s$.*

**Assumption 2** *Assume that all the scenarios have equal probabilities, i.e.,*

$$p_s = \frac{1}{S}, \quad 1 \leq s \leq S.$$

Assumption 2 is made for the sake of simplicity of presentation. In fact, the $VaR$ functional can be represented as a DC function even in the case of unequal weights [21].

Given the above assumptions, the problem (1) becomes

$$\begin{cases} \max\limits_{w} \ \dfrac{1}{S} \sum\limits_{s=1}^{S} (w^T \xi^s) \\ \text{s.t.} \ \sum\limits_{i=1}^{m} w_i = 1, \\ \qquad w_i \in [a_i, b_i], 1 \leq i \leq m, \\ \qquad VaR_\alpha(w^T \xi) \geq a. \end{cases} \tag{3}$$

It is easy to see that if $X$ follows a discrete distribution taking the values $x_1, ..., x_S$ with equal probability (in our case $X$ represents the anticipated random returns of the portfolio, i.e., $X = w^T \xi$ ), then

$$AVaR_{\frac{k}{S}}(X) = \frac{S}{k} \sum_{i=1}^{k} x_{\sigma(i)} \frac{1}{S} = \frac{1}{k} \sum_{i=1}^{k} x_{\sigma(i)},$$

where $x_{\sigma(1)} \leq x_{\sigma(2)} \leq ... \leq x_{\sigma(S)}$ is the set of ordered values of $X$ ($\sigma$ is a permutation of the set {1, 2, ..., S}). Therefore

$$VaR_\alpha(X) = x_{\sigma(k)} = kAVaR_{\frac{k}{S}}(X) - (k-1)AVaR_{\frac{k-1}{S}}(X),$$

with $k = \lfloor \alpha S \rfloor$. Hence under the above assumptions, $VaR_\alpha(X)$ can be written as the difference of the two concave functions

$$kAVaR_{\frac{k}{S}}(X) \text{ and } (k-1)AVaR_{\frac{k-1}{S}}(X).$$

Therefore, the problem (3) becomes

$$\begin{cases} \max\limits_{w} \ \dfrac{1}{S}\sum\limits_{s=1}^{S}(w^T\xi^s) \\ \text{s.t. } \sum\limits_{i=1}^{m} w_i = 1, \\ \quad w_i \in [a_i, b_i], 1 \le i \le m, \\ \quad kAVaR_{\frac{k}{S}}(w^T\xi) - (k-1)AVaR_{\frac{k-1}{S}}(w^T\xi) \ge a, \end{cases} \tag{4}$$

or equivalently can be written as a minimization problem with a DC constraint:

$$\begin{cases} \max\limits_{w} \ -\dfrac{1}{S}\sum\limits_{s=1}^{S}(w^T\xi^s) \\ \text{s.t. } \sum\limits_{i=1}^{m} w_i = 1, \\ \quad w_i \in [a_i, b_i], 1 \le i \le m, \\ \quad (-kAVaR_{\frac{k}{S}}(w^T\xi)) - (-(k-1)AVaR_{\frac{k-1}{S}}(w^T\xi)) \le -a. \end{cases} \tag{5}$$

## 2.2 An exact penalty technique

We consider the following nonconvex optimization problems

$$\alpha = \min\{f(x) : x \in K, g(x) = 0\} \qquad (P)$$

$$\alpha(\tau) = \min\{f(x) + \tau g(x) : x \in K, g(x) \ge 0\} \qquad (P_\tau)$$

**Theorem 2.1** ***(H.A. Le Thi et al. [12])***
*Let $K$ be a non-empty bounded polyhedral convex set in $\mathbb{R}^n$ and let $f, g$ be finite concave function on $K$. Suppose that the feasible set of $(P)$ is not empty. Then there exists $\tau_0 \ge 0$ such that for all $\tau > \tau_0$, the problems $(P)$ and $(P_\tau)$ are identical. Furthermore, we can take $\tau_0 = \frac{f(x_0)-\alpha(0)}{m}$, with $m = \min\{g(x) : \ x \in V(K), g(x) > 0\}$ and any $x_0 \in K, g(x_0) = 0$. Here, the convention $\min\limits_{\emptyset} g(x) = +\infty$ is used.*

Due to this theorem, we could transform problem (5) to polyhedral DC program as follows.

Set

$$C = \{w \in \mathbb{R}^m : \sum_{i=1}^{m} w_i = 1, w_i \in [a_i, b_i], 1 \le i \le m\}.$$

It is clear to see that $(-kAVaR_{\frac{k}{S}}(w^T\xi))$ and $(-(k-1)AVaR_{\frac{k-1}{S}}(w^T\xi))$ are polyhedral convex functions in $w$ since

$$kAVaR_{\frac{k}{S}}(w^T\xi) = \min\left\{\sum_{s\in I} w^T\xi^s : I \in S_k\right\},$$

$$(k-1)AVaR_{\frac{k-1}{S}}(w^T\xi) = \min\left\{\sum_{s\in J} w^T\xi^s : J \in S_{k-1}\right\},$$

where

$$S_k = \{I \subset \{1,2,...,S\} : |I| = k\}, S_{k-1} = \{J \subset \{1,2,...,S\} : |J| = k-1\}.$$

Let

$$f(w) = -\frac{1}{S}\sum_{s=1}^{S}(w^T\xi^s),$$

$$g(w) = -kAVaR_{\frac{k}{S}}(w^T\xi) + a,$$

$$h(w) = -(k-1)AVaR_{\frac{k-1}{S}}(w^T\xi).$$

The problem (5) can be written in the form:

$$\begin{cases} \min & f(w) \\ \text{s.t.} & w \in C, \\ & g(w) - h(w) \le 0. \end{cases} \tag{6}$$

We have

$$g(w) - h(w) \le 0 \Leftrightarrow g(w) \le h(w) \Leftrightarrow g(w) \le t \le h(w)$$
$$\Leftrightarrow g(w) - t \le 0, t - h(w) \le 0.$$

Thus, the problem (6) becomes

$$\begin{cases} \min & f(w) \\ \text{s.t.} & (w,t) \in C \times [t_1, t_2], \\ & g(w) - t \le 0, \\ & t - h(w) \le 0, \end{cases} \tag{7}$$

where

$$t_1 \le \min_{w\in C} g(w), \quad t_2 \ge \max_{w\in C} h(w).$$

This problem is equivalent to

$$\begin{cases} \min & f(w) \\ \text{s.t.} & (w,t,s) \in C \times [t_1, t_2] \times [0, \beta], \\ & g(w) - t \le 0, \\ & t + s - h(w) = 0, \end{cases} \tag{8}$$

where

$$\begin{aligned}\beta &= \max\{h(w) - t :\ w \in C, t \in [t_1, t_2]\} \\ &= t_2 - t_1.\end{aligned}$$

Since $g$ is polyhedral convex in $w$, the set $\{(w,t) \in \mathbb{R}^{m+1} :\ w \in C,\ g(w) - t \leq 0\}$ is also polyhedral convex. Due to Theorem 2.1, the problem (8) is equivalent to

$$\begin{cases} \min & f(w) + \tau(t + s - h(w)) \\ \text{s.t.} & (w,t,s) \in C \times [t_1, t_2] \times [0, \beta], \\ & g(w) - t \leq 0, \\ & h(w) - t - s \leq 0. \end{cases} \tag{9}$$

Because $-[f(w)+\tau(t+s-h(w))]$ is polyhedral convex and the constraints display a bounded polyhedral convex set. The problem (9) is a polyhedral DC program.

# 3 Application of DCA to problem (9)

## 3.1 Outline of DC Programming and DCA

DC Programming and DCA [11, 17, 18] constitute the backbone of smooth/nonsmooth nonconvex programming and global optimization. They are introduced by Pham Dinh Tao in 1985 in their preliminary form and extensively developed by Le Thi Hoai An and Pham Dinh Tao since 1994 to become now classic and more and more popular.

We consider the DC (difference of convex functions) program:

$$\alpha = \inf\{f(x) := g(x) - h(x) : x \in X\} \quad (P_{dc})$$

where $X = \mathbb{R}^n$ is the usual Euclidean space and $g, h$ are lower semicontinuous proper convex functions on $X$.

We are interested in local and global optimality conditions, relationships between local and global solutions to primal DC programs and their dual

$$\alpha = \inf\{h^*(y) - g^*(y) : y \in Y\} \quad (D_{dc})$$

where $Y$ is the dual space of $X$, which can be identified with $X$ it self, and $g^*, h^*$ denote the conjugate functions of $g$ and $h$, respectively and solution algorithms.

The transportation of global solutions between $(P_{dc})$ and $(D_{dc})$ is expressed as:

- If $x^*$ is an optimal solution of $(P_{dc})$, then $y^* \in \partial h(x^*)$ is an optimal solution of $(D_{dc})$,
- If $y^*$ is an optimal solution of $(D_{dc})$, then $x^* \in \partial g^*(y^*)$ is an optimal solution of $(P_{dc})$.

Under technical conditions, this transportation holds also for local solutions of $(P_{dc})$ and $(D_{dc})$.

DC programming investigates the structure of the vector space $DC(X)$, DC duality and optimality conditions for DC programs. The complexity of DC programs resides, of course, in the lack of practical optimal global conditions. We developed instead the following necessary local optimality conditions for DC programs in their primal part, (by symmetry their dual part is trivial):

$$\partial g(x^*) \cap \partial h(y^*) \neq \emptyset \quad (i)$$

such a point $x$ is called critical point of $g - h$ or for $(P_{dc})$, and

$$\partial g(x^*) \subset \partial h(y^*) \quad (ii)$$

The condition $(ii)$ is also sufficient for many important classes of DC programs. In particular it is sufficient for the next cases quite often encountered in practice:

- In polyhedral DC programs with $h$ being a polyhedral convex function. In this case, if $h$ is differentiable at a critical point $x$, then $x$ is actually a local minimizer for $(P_{dc})$. Since a convex function is differentiable everywhere except for a set of measure zero, one can say that a critical point $x$ is almost always a local minimizer for $(P_{dc})$.
- In case the function $f$ is locally convex at $x$.

Based on local optimality conditions and duality in DC programming, the DCA consists in the construction of two sequences $\{x^k\}$ and $\{y^k\}$, candidates to be optimal solutions of primal and dual programs respectively, such that the sequences $\{g(x^k)-h(x^k)\}$ and $\{h^*(y^k)-g^*(y^k)\}$ are decreasing, and $\{x^k\}$ (resp. $\{y^k\}$) converges to a primal feasible solution $x^*$ (resp. a dual feasible solution $y^*$) verifying local optimality conditions and

$$x^* \in \partial g^*(y^*), \quad y^* \in \partial h(x^*).$$

Starting $x^0 \in dom(g)$, these two sequences $x^k$ and $y^k$ are determined in the way following

$$x^k \in \partial g^*(y^{k-1}) \to y^k \in \partial h(x^k) = \operatorname{argmin}\{h^*(y) - g^*(y^{k-1}) - \langle y - y^{k-1}, x^k\rangle : y \in R^n\} \ (D_k)$$

$$y^k \in \partial h(x^k) \to x^{k+1} \in \partial g^*(y^k) = \operatorname{argmin}\{g(x) - h(x^k) - \langle x - x^k, y^k\rangle : x \in R^n\} \ (P_k)$$

Problem $(P_k)$ is a convex program obtained from $(P)$ by replacing $h$ with its affine minorization defined by $y^k \in \partial h(x^k)$. Similarly, the convex problem $(D_k)$ is obtained from $(D)$ by using the affine minorization of $g^*$ defined by $x^k \in \partial(y^{k-1})$.

Implement the algorithm that consists of three steps:

1. Choose $x^0 \in \mathbb{R}^n$.
2. Set $y^k \in \partial h(x^k)$.
3. Set $x^{k+1} \in \partial g^*(y^k)$ that leads to solving the convex program

$$\inf\{g(x) - h(x^k) + \langle x - x^k, y^k\rangle : x \in \mathbb{R}^n\}.$$

Until the convergence.

## 3.2 Application DCA for solving Problem (9)

For applying DCA to Problem (9), we need to compute $\partial h(w)$ for each $w \in C$. Because

$$h(w) = -(k-1)AVaR_{\frac{k-1}{S}}(w^T\xi) = -\sum_{i=1}^{k-1} r_{i:S}(w),$$

where the $r_{i:S}(w)$ are the ordered returns depending on the portfolio $w$. Suppose that we have $w^p$, therefore $r_{i:S}(w^p)$ for $i = 1, 2, ..., m$, we will compute $y^p = \partial h(w^p)$ as follows:

- If $r_{1:S} \leq ... \leq r_{(k-1):S} < r_{k:S} \leq ... \leq r_{S:S}$ then

$$\partial h(w^p) = \nabla h(w^p) = -\sum_{i=1}^{k-1} \xi^{i:S}.$$

- If $r_{1:S} \leq ... \leq r_{(s-1):S} < r_{s:S} = ... = r_{(k-1):S} = r_{k:S} = ... = r_{t:S} < r_{(t+1):S} \leq ... \leq r_{S:S}$ then $\partial h(w^p)$ can be written as the convex hull of the following vectors

$$V = \left\{ \sum_{i=1}^{s-1} \nabla r_{i:S}(w^p) + \sum_{j\in J} \nabla r_{j:S}(w^p) : J \subset \{s, s+1, ..., t\}, |J| = k-s-2 \right\}$$
$$= \left\{ -\sum_{i=1}^{s-1} \xi^{i:S} - \sum_{j\in J} \xi^{j:S} : J \subset \{s, s+1, ..., t\}, |J| = k-s-2 \right\}.$$

**Algorithm 1: The DCA for solving the problem (9)**

**Step 1.** Initialization: Let $w^0 \in C$, $\epsilon_1 > 0, \epsilon_2 > 0$, and $p = 0$.

**Step 2.** Compute $y^p = \partial h(w^p)$ by the above formulas.

**Step 3.** Compute $w^{p+1}$ by solving the linear program

$$\begin{cases} \min\ (f(w) + \tau(t+s) - \tau\langle w, y^p\rangle) \\ \text{s.t.}\ \ (w,t,s) \in C \times [t_1, t_2] \times [0, \beta], \\ \qquad g(w) - t \leq 0, \\ \qquad h(w) - t - s \leq 0. \end{cases} \tag{10}$$

**Step 4.** Iterate Step 2 and 3 until

$$|f(w^{(p+1)}) - f(w^p)| \leq \epsilon_1(1 + |f(w^{(p+1)})|)$$
$$\text{or}\ \ ||w^{p+1} - w^p||_\infty \leq \epsilon_2(1 + ||w^{p+1}||_\infty).$$

In practice, to solve the problem (10), we use an useful linear reformulation for constraints, which is presented in [19, 20]. Because the problem (9) is polyhedral DC program, we have the following convergence theorem:

**Theorem 3.1** *The discrete sequence* $\{(w^p, t^p, s^p)\}$ *(i.e., it has only finitely many different elements) generated by Algorithm 1 verifies:*

1. *The sequence* $\{f(w^p) + \tau(t^p + s^p - h(w^p))\}$ *is decreasing.*
2. *The sequence* $\{(w^p, t^p, s^p)\}$ *is convergent.*

**Proof** see [11, 17].

# 4 Numerical Experiment

The algorithm described above is tested on real-world financial market data. The code is written in C++, and is tested on a notebook with chipset Intel(R) Core(TM) Duo CPU 2.0 GHz, 3GB of RAM. We use CPLEX 12.1 for solving linear programming.

Our experiment is composed of two parts. In the first part, we study the globality of the DCA by comparing the solutions given by the DCA with an optimal solution furnished by the software CPLEX 12.1 applied to the mixed-integer formulation of the problem (1) (see [3]). In the second experiment, we consider large datasets (for which CPLEX can not find global solutions) and compare our method with a heuristic algorithm developed in [10]. In the following tests, we choose $a_i = 0, b_i = 1, i = 1, ..., m$.

## 4.1 Comparison with global optima: DCA gives a good near-optimal solution

We used the empirical distribution of two years of weekly data, i.e., $S = 104$ scenarios of the following 5 indices: CAC 40, Standard&Poors 100, Nasdaq 100, FTSE 100, and Hang Seng. We take $\alpha = 0.045$. Different values of $a$ are considered in the dataset ($a$ is incremented by 0.0005). It is easy to see that if $a$ is smaller than the $VaR$ of the portfolio that consists only of the asset with the highest return, then the problem is trivial. Hence, in our test problems, $a$ is greater than that $VaR$.

The results are presented in Table 2, where we compare the objective value (say, the expected return) obtained by our DCA and the optimal value given by CPLEX software. The Figure 2 shows the efficient frontier traced by each algorithm that illustrates the globality of the DCA in almost all cases.

We observe from the numerical results that in almost cases the results given by DCA the same as the global solutions, except the cases $a = 0.9805$ and $a = 0.9825$. Moreover the DCA is fast: CPU time is approximately 0.5 seconds.

| Name | Average Return | Variance | $VaR_{0.045}$ | $AVaR_{0.045}$ |
|---|---|---|---|---|
| CAC 40 | 1.0027144 | 0.000217647 | 0.9742 | 0.9694 |
| Standard&Poors100 | 1.0004326 | 0.000179629 | 0.9755 | 0.9717 |
| Nasdaq 100 | 1.0013597 | 0.000471718 | 0.9653 | 0.9504 |
| FTSE 100 | 1.0021901 | 0.000151213 | 0.9824 | 0.9797 |
| Hang Seng | 1.0016546 | 0.000421260 | 0.9624 | 0.9561 |

Table 1: Time frame: 2004-2005, weekly data

| a | DCA | | | Optimal solution | | |
|---|---|---|---|---|---|---|
| | Expected Return | $VaR_{0.045}$ | Time(s) | Expected Return | $VaR_{0.045}$ | Time(s) |
| 0.9745 | 1.00270427 | 0.974500 | 0.47 | 1.00270427 | 0.974500 | 0.49 |
| 0.9750 | 1.00268702 | 0.975000 | 0.47 | 1.00268702 | 0.975000 | 0.58 |
| 0.9755 | 1.00266978 | 0.975500 | 0.47 | 1.00266978 | 0.975500 | 0.41 |
| 0.9760 | 1.00265254 | 0.976000 | 0.48 | 1.00265254 | 0.976000 | 0.47 |
| 0.9765 | 1.00263307 | 0.976500 | 0.45 | 1.00263307 | 0.976500 | 0.39 |
| 0.9770 | 1.00261218 | 0.977000 | 0.47 | 1.00261218 | 0.977000 | 0.39 |
| 0.9775 | 1.00259129 | 0.977500 | 0.50 | 1.00259129 | 0.977500 | 0.42 |
| 0.9780 | 1.00257040 | 0.978000 | 0.47 | 1.00257040 | 0.978000 | 0.38 |
| 0.9785 | 1.00254951 | 0.978500 | 0.52 | 1.00254951 | 0.978500 | 0.34 |
| 0.9790 | 1.00252862 | 0.979000 | 0.50 | 1.00252862 | 0.979000 | 0.36 |
| 0.9795 | 1.00250773 | 0.979500 | 0.47 | 1.00250773 | 0.979500 | 0.36 |
| 0.9800 | 1.00248684 | 0.980000 | 0.47 | 1.00248684 | 0.980000 | 0.34 |
| 0.9805 | 1.00229933 | 0.980500 | 0.47 | 1.00230464 | 0.980500 | 0.38 |
| 0.9810 | 1.00228322 | 0.981000 | 0.47 | 1.00228322 | 0.981000 | 0.50 |
| 0.9815 | 1.00226662 | 0.981500 | 0.47 | 1.00226662 | 0.981500 | 0.53 |
| 0.9820 | 1.00224655 | 0.982000 | 0.47 | 1.00224655 | 0.982000 | 0.53 |
| 0.9825 | 1.00186039 | 0.982500 | 0.45 | 1.00222647 | 0.982500 | 0.59 |
| 0.9830 | 1.00173056 | 0.983000 | 0.47 | 1.00173056 | 0.983000 | 0.70 |
| 0.9835 | 1.00160072 | 0.983500 | 0.47 | 1.00160072 | 0.983500 | 0.89 |
| 0.984 | 1.00146105 | 0.984000 | 0.46 | 1.00146105 | 0.984000 | 0.72 |

Table 2: DCA compares with global optima

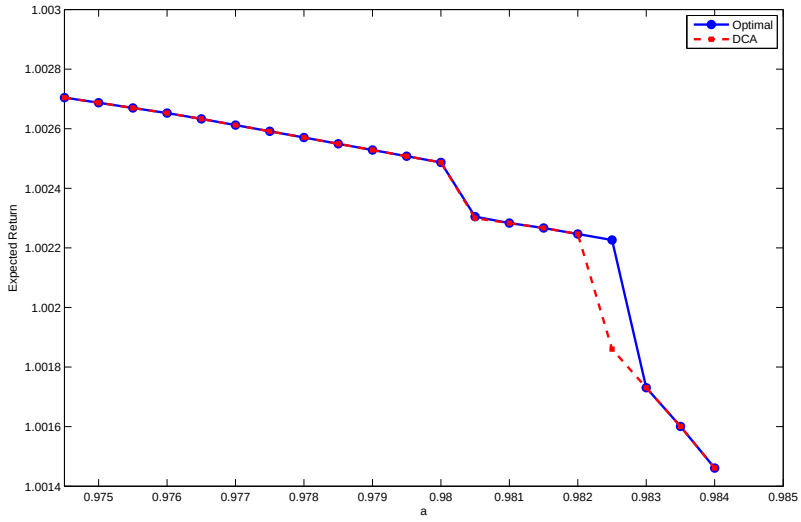


Figure 1: Efficient frontier: DCA and Optima

## 4.2 Large scale problem: DCA is a scalable approach

To study the scability of the DCA we first test on the 1304 daily returns for 43 assets of of the Eurostoxx50 index in year from January 1, 2003 to December 31, 2007, which give positive returns. Then, we test on the empirical distribution of 11 years (from January 1, 2000 to December 31, 2010) of daily data, i.e., 2759 scenarios of the 87 assets comprising the NYSE US 100 index.

For these datasets, CPLEX furnished only a feasible solution which is not a global solution. We then compare our DCA with an efficient existing method developed in [10], called Algorithm A1. This method is widely used in the industry for the Value-at-Risk portfolio selection. It solves the following problem:

$$\begin{cases} \min\limits_{w} \ VaR_{1-\alpha}(w^T\xi) \\ \text{s.t. } w \in [0,1], \sum\limits_{i=1}^{n} w_i = 1, \\ \quad \mathbb{E}(w^T\xi) \geq C, \end{cases} \tag{11}$$

where $C$ is a given lower bound expectation.
Note that in [10] $X$ represents the random losses. Hence minimizing $VaR_{1-\alpha}$ amounts to maximizing $VaR_{\alpha}$. The main idea of Algorithm A1 is the approximation of the $VaR$ by $CVaR$ and iterative application of discarding scenarios with large losses. This algorithm performs well even on large datasets. Algorithm A1 from [10] adapted to our problem is described as follows ([22]):

1. Fix $C$ as a lower bound on expectation, a parameter for the Value-at-Risk $\alpha$ and a

parameter for the heuristic $0 < \zeta < 1$.

2. Set $\alpha_0 = \alpha, i_0 = 0$, and $k = 0$.

3. Solve the problem

$$\begin{cases} \max\limits_{w} \ AVaR_{\alpha_k}(\sum_{i=i_k}^{S} w^T \xi^i) \\ \text{s.t.} \ \sum\limits_{i=1}^{n} w_i = 1, \\ \quad \mathbb{E}(w^T \xi) \geq C, \\ \quad w^T \xi^i \leq \gamma, \quad i \leq i_k, \\ \quad w^T \xi^i \geq \gamma, \quad i > i_k, \\ \quad w_i \geq 0, \quad 1 \leq i \leq n. \end{cases} \tag{12}$$

4. Call the solution of the above problem $w_k$ and sort the scenarios $\xi^i$ according to their returns $r_i = w_k^T \xi^i$.

5. Set $k = k + 1, b_k = \alpha + (1 - \alpha)(1 - \zeta)^k, i_k = \lfloor S(1 - b_k) \rfloor$, and $\alpha_k = 1 - \dfrac{1 - \alpha}{b_k}$.

6. If $i_k \leq \lfloor S/\alpha \rfloor$ goto step 3, otherwise exit.

The comparison between DCA and A1 algorithm has proceeded as follows: for each value $a$, we solve the problem (1) by DCA to get an expected return and the corresponding Value-at-Risk. That expected return becomes lower bound expectation (say, the value of $C$) for A1 algorithm. The A1 algorithm will produce a new Value-at-Risk, and then we compare the Value-at-Risk obtained by the two methods. In the two numerical comparisons below, we choose the parameter $\zeta = 0.5$ for A1 algorithm.

Table 3 represents the results corresponding to Eurostoxx50 data, while the Figure 2 shows the efficient frontier traced by each algorithm. The lower bound for $a$ is the $VaR$ of the portfolio that consists only of the asset with the highest return (i.e., AXA with $VaR_{0.05}$ of 0.9713956 and expected daily return 1.002344), and the upper bound is 0.983, because DCA does not work anymore for a greater value.

From Table 3 and 2 we can see that the difference of $VaR_{0.05}$ between two methods is small, it varies from 7.85E-04 to 3.95E-03. Additionally, the number of active assets, i.e., the number of assets whose portfolio contributions are different from 0, in solution obtained by DCA in most cases is greater. It means that the portfolio given by DCA is more diversified.

Table 4 presents the results corresponding to NYSE US 100 daily data, with $\alpha = 0.05$. Also, the curve in figure 3 shows the efficient frontier each algorithm. Here, $a$ varies from 0.953 to 0.983. The lower bound for $a$ is the $VaR$ of the portfolio that consists only of the asset with the highest return (i.e., AIG with $VaR_{0.05}$ of 0.95263158 and expected daily return 1.00518861).

| a | DCA | | | | A1 algorithm | | | Difference |
|---|---|---|---|---|---|---|---|---|
| | Expected Return | $VaR_{0.05}(CE)$ | Active | Time(s) | $VaR_{0.05}(A1)$ | Active | Time(s) | |
| 0.972 | 1.002214 | 0.973127 | 5 | 31.59 | 0.974010 | 6 | 138.28 | 0.000883 |
| 0.973 | 1.002172 | 0.973544 | 6 | 31.86 | 0.974329 | 9 | 156.09 | 0.000785 |
| 0.974 | 1.002038 | 0.974685 | 8 | 163.67 | 0.976597 | 8 | 143.40 | 0.001912 |
| 0.975 | 1.001877 | 0.975563 | 10 | 93.64 | 0.978440 | 11 | 163.28 | 0.002877 |
| 0.976 | 1.001839 | 0.976539 | 11 | 79.38 | 0.978823 | 11 | 169.12 | 0.002284 |
| 0.977 | 1.001702 | 0.977581 | 14 | 78.50 | 0.980512 | 11 | 145.23 | 0.002931 |
| 0.978 | 1.001535 | 0.978353 | 15 | 78.22 | 0.982240 | 10 | 168.01 | 0.003887 |
| 0.979 | 1.001535 | 0.980599 | 16 | 211.88 | 0.982240 | 10 | 166.75 | 0.001641 |
| 0.980 | 1.001369 | 0.980145 | 17 | 238.14 | 0.983807 | 15 | 165.43 | 0.003662 |
| 0.981 | 1.001338 | 0.982272 | 19 | 295.49 | 0.984424 | 11 | 167.48 | 0.002152 |
| 0.982 | 1.001338 | 0.982272 | 19 | 203.11 | 0.984424 | 11 | 166.37 | 0.002152 |
| 0.983 | 1.000976 | 0.983827 | 20 | 205.00 | 0.987778 | 14 | 166.57 | 0.003951 |

Table 3: The performance of CE algorithm compared with the A1 algorithm, Eurostoxx50 daily data

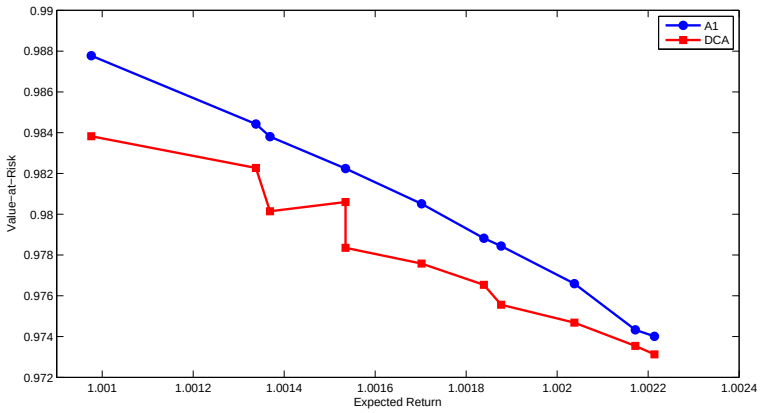


Figure 2: Efficient frontier: Eurostoxx50 data

| a | DCA | | | | A1 algorithm | | | Difference |
|---|---|---|---|---|---|---|---|---|
| | Expected Return | $VaR_{0.05}(CE)$ | Active | Time(s) | $VaR_{0.05}(A1)$ | Active | Time(s) | |
| 0.953 | 1.005061 | 0.953390 | 6 | 959.63 | 0.953819 | 5 | 546.218 | 0.000429 |
| 0.955 | 1.004888 | 0.955019 | 6 | 379.09 | 0.955824 | 7 | 686.235 | 0.000805 |
| 0.957 | 1.004705 | 0.957126 | 8 | 697.45 | 0.957619 | 10 | 658.171 | 0.000493 |
| 0.959 | 1.004429 | 0.959168 | 8 | 863.77 | 0.958908 | 11 | 666.422 | 0.000260 |
| 0.961 | 1.004007 | 0.961210 | 10 | 765.27 | 0.963695 | 10 | 658.188 | 0.002485 |
| 0.963 | 1.003834 | 0.963117 | 14 | 954.78 | 0.965239 | 14 | 657.718 | 0.002122 |
| 0.965 | 1.003576 | 0.965158 | 15 | 980.16 | 0.967907 | 13 | 579.469 | 0.002749 |
| 0.967 | 1.003248 | 0.967065 | 14 | 980.75 | 0.970413 | 15 | 758.078 | 0.003348 |
| 0.969 | 1.003061 | 0.969107 | 13 | 967.88 | 0.971647 | 17 | 666.156 | 0.002540 |
| 0.971 | 1.002784 | 0.971149 | 22 | 1520.00 | 0.973388 | 22 | 656.734 | 0.002239 |
| 0.973 | 1.002317 | 0.973059 | 36 | 164.03 | 0.976911 | 24 | 830.688 | 0.003852 |
| 0.975 | 1.002174 | 0.975101 | 37 | 328.45 | 0.977981 | 22 | 681.609 | 0.002880 |
| 0.977 | 1.002042 | 0.977007 | 36 | 347.08 | 0.979110 | 24 | 687.922 | 0.002103 |
| 0.979 | 1.001789 | 0.979047 | 32 | 521.23 | 0.980465 | 20 | 705.547 | 0.001418 |
| 0.981 | 1.001271 | 0.981192 | 45 | 288.92 | 0.983466 | 27 | 693.766 | 0.002274 |
| 0.983 | 1.001064 | 0.983097 | 44 | 354.47 | 0.984695 | 21 | 596.891 | 0.001598 |

Table 4: NYSE US 100 daily data with $\alpha = 0.05$.

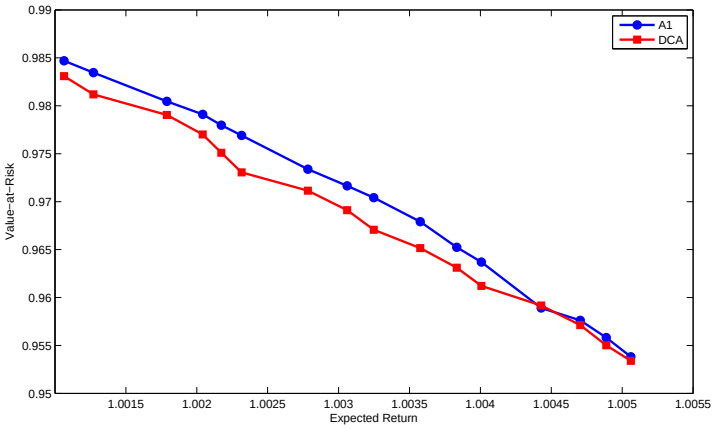


Figure 3: Efficient frontier: NYSE US 100 daily data

We observe from the numerical results that the difference of $VaR_{0.05}$ given by the two methods varies from 2.60E-04 to 3.85E-03. Moreover, for $a$ from 0.973 to 0.983, the number of active assets in solution obtained by DCA is greater.

# 5 Conclusion

In this paper, we have proposed a new reformulation of the problem (1) based on an exact penalty technique, and DCA for solving locally the polyhedral DC program (9). Some numerical experiments have shown the efficiency of our approach. In fact, the quality of the solution obtained by DCA, as well as the CPU time in running DCA depend on strongly the starting point. Whenever having a good starting point, we can get a better local solution or even a global solution within a shorter time. Therefore, how to find a good starting point for DCA is an interesting work in the future.

# References


[1] Ph. Artzner, F. Delbaen, J-M. Eber, and D. Heath, Thinking coherently, Risk, 10(11) (1997), pp. 68-71.

[2] Ph. Artzner, F. Delbaen, J-M. Eber, and D. Heath, Coherent measures of risk, Mathematical Finance, 9(1999), pp. 203-228.

[3] S. Benati and R. Rizzi, A mixed integer linear programming formulation of the optimal mean/value-at-risk portfolio problem, European Journal of Operational Research, 176 (2007), pp. 423-434.

[4] R. Campbell, R. Huisman, and K. Koedijk, Optimal portfolio selection in avalue-at-risk framework, Journal of Banking & Finance, 25(2001), pp. 1789-1804.

[5] M. S. Cheon, S. Ahmed, and F. Al-Khayyal, A branch-reduced-cut algorithm for the global optimization of probabilistically constrained linear programs, Mathematical Programming, 108 (2006), pp. 617-634.

[6] M. Gilli and E. Kellezi, A global optimization heuristic for portfolio choice with VaR and expected shortfall,in Computational methods in decision-making, economics and finance, vol. 74 of Applied Optimization, Kluwer, 2002, pp. 167-183.

[7] M. Gilli, E.Kellezi, and H. Hysi, A data-driven optimization heuristic for downside risk minimization, The Journal of Risk, 8(2006), pp. 1-18.

[8] R. Hochreiter, An evolutionary computation approach to scenario-based risk-return portfolio optimization for general risk measures, in EvoWorkshops 2007, vol. 4448 of Lecture Notes in Computer Science, Springer, 2007, pp. 199-207.

[9] P. Jorion, Value at Risk: The New Benchmark for Controlling Market Risk, McGraw-Hill, 2000.

[10] Larsen, N., Mausser, H., Uryasev, S.: Algorithms for optimization of value-at-risk. In: Pardalos, P., Tsitsiringos, V. (eds.) Financial Engineering, e-Commerce and Supply Chain, pp. 129-157. Kluwer Academic Publishers, Dordrecht, Netherlands(2002).

[11] H.A. Le Thi, and T. Pham Dinh, The DC (difference of convex functions) Programming and DCA revisited with DC models of real world nonconvex optimization problems, Annals of Operations Research, Vol 133, pp. 23-46, 2005.

[12] H.A. Le Thi, T. Pham Dinh, and N. Huynh Van, Exact Penalty and Error Bound in DC Programming, to appear in Journal of Global Optimization.

[13] Mausser, H. and D. Rosen (1998): Beyond VaR: From Measuring Risk to Managing Risk, Algo Research Quarterly, Vol.1, No.2, 5-20.

[14] J.S. Pang and S. Leyffer, On the global minimization of the value-at-risk, Optimization Methods and Software, 19(2004), pp. 611-631.

[15] G.Ch. Pflug and W. Romisch, Modeling, Measuring and Managing Risk, World Scientific, Singapore, 2007.

[16] G.Ch. Pflug, Some remarks on the Value-at-Risk and the Conditional Value-at-Risk, in Probabilistic constrained optimization, vol. 49 of Nonconvex Optimization and its Applications, Kluwer, 2000, pp. 272-281.

[17] T. Pham Dinh, and H.A. Le Thi, Convex analysis approach to DC programming: Theory, Algorithms and Applications (dedicated to Professor Hoang Tuy on the occasion of his 70th birthday), Acta Mathematica Vietnamica, 22, pp. 289-355, 1997.

[18] T. Pham Dinh, and H.A. Le Thi, DC optimization algorihms for solving the trust region subproblem, SIAM J.Optimization 8, 476-505, 1998.

[19] R. T. Rockafellar and S. Uryasev, Optimization of Conditional Value-at-Risk, The Journal of Risk, 2(2000), pp. 21-41.

[20] S. Uryasev, Conditional Value-at-Risk: Optimization algorithms and applications, Financial Engineering News, 14 (2000), pp. 1-5.

[21] Wozabal, D., Hochreiter, R., Pflug, G. : A d. c. formulation of value-at-risk constrained optimization. Tech. Rep. TR2008-01, Department of Statistics and Decision Support Systems, University of Vienna, Vienna(2008).

[22] Wozabal, D., Value-at-Risk optimization using the difference of convex algorithm, OR Spectrum: 1-23, online first (2010)

# Conclusion and Perspective

## Conclusion

In this thesis, we have studied some optimization problems in Search Information, Assignment and Portfolio Management. Our methodologies are based on DC programming& DCA and the Cross-Entropy method, which are known to be powerful tools in optimization. Due to the techniques of formulation/reformulation, we have given the DC formulation of considered problems such that we can use DCA to obtain their solutions. Also, depending on the structure of feasible sets of considered problems, we have designed appropriate families of distributions such that the Cross-Entropy method could be applied. The main contribution is to develop new approaches for efficiently solving these problems, especially in very large dimension. Specifically:

- We have presented a new approach for solving the problem of *planning a multisensor multizone search for a target*. This is the first time a deterministic optimization model is introduced in the literature. This constitues an interesting contribution of the paper. For solving the combinatorial optimization problem by DCA, an innovative continuous approach in nonconvex programming framework we first reformulated the original problem in the form of DC program by proving an exact penalty result in which the penalty parameter can be estimated. That is our second important contribution. The third contribution deals with the development of an efficient DCA scheme for solving the resulting problem.

- In the case of *moving (Markovian) target*, although the above problem becomes very complicated, we have successfully combined the forward-backward split technique and the DCA for its solution.

- We proposed two approaches to tackle the *Nonlinear UAV Task Assignment Problem*. We first developed an heuristic method based on the Cross-Entropy (CE) method. Here, we have chosen an appropriate family of discrete distributions characterized by probability matrices on the feasible sets of this problem. The second one is the Branch and Bound algorithm for measuring the efficiency of the CE method and globally solving the considered problem.

- A new heuristic approach based on Cross-Entropy method for solving the *Multidimensional Assignment Problem (MAP)* is introduced. Due to the special structure of the

MAP, we designed an appropriate family of discrete distributions on the feasible set of the MAP, which allows us to apply the Cross-Entropy method efficiently.

- We developed both deterministic and heuristic approaches for solving the *Value-at-Risk constrained optimization problem*. In the first approach, we introduced an appropriate family of continuous distributions based on the family of exponential distributions on the feasible sets of this problem to apply the Cross-Entropy method. In the second one, we reformulated the problem as a polyhedral DC program by using an exact penalty technique and proposed DCA for its solution.

Moreover, to demonstrate the efficiency of our approaches, we implemented optimization algorithms in MATLAB, C/C++:

- The DCA for the problem of planning a multisensor in multizone search for a target.
- The combination of the forward-backward split technique and DCA (FAB&DCA) for the case of moving target.
- BB&DCA for the Nonlinear UAV Task Assignment Problem.
- The CE algorithm for the Nonlinear UAV Task Assignment Problem.
- The CE algorithm for the Multidimensional Assignment Problem.
- The DCA for the Value-at-Risk constrained Optimization problem.
- The CE algorithm for the Value-at-Risk constrained Optimization problem.

# Perspective

Some questions are posed in this thesis:

- Although DCA showed good performances in applications, the quality of the solution obtained by DCA, as well as the CPU time in running DCA strongly depend on the starting point. Whenever having a good starting point, we can get a better local solution or even a global solution within a shorter time. Thus, how to find a good starting point for DCA in our considered problems is an interesting work in the future.
- The approach for solving the problem of *planning a multisensor multizone search for a target* is a local method. With the obtained DC formulation of problem, how can we develop global approaches for solving this problem?
- The *Nonlinear UAV Task Assignment Problem* is only a sub-problem of UAV coordination problem. In the future, we plan to deeply study UAV coordination problems.

- The Cross-Entropy algorithm provided good solutions for the *Value-at-Risk constrained optimization problem*. But if the constraints become more complex, how we can use the CE method? And can we apply this approach for the other optimization problems in portfolio management.

- In this thesis, we used the CE method and DCA separately. They both showed good performances. How we could combine them is also an attractive job?

- From our experiences in performing the CE algorithms, there is a positive relationship between the quality of solutions and the number of samples $N$. However, when $N$ increases, so does the CPU time. How to parallelize the CE algorithms to save time?

- In the future, we intend to apply our proposed approaches in some industrial applications. For instance, application of the MAP to data association, image recognition, multisensor multitarget tracking, etc. Also, we continuously apply the cross-entropy for the other problems.

# Bibliography


[1] Renata M. Aiex, Mauricio G. C. Resende, Panos M. Pardalos, and Gerardo Toraldo. 2005. GRASP with Path Relinking for Three-Index Assignment. INFORMS J. on Computing 17, 2 (April 2005), 224-247.

[2] F. Akoa, Combining DC Algorithms (DCAs) and Decomposition Techniques for the Training of Nonpositive-Semidefinite Kernels, IEEE Transactions on Neural Networks 2008 Vol. 19 (No.11).

[3] Aleksandrov A.D., On the surfaces represented as the difference of convex functions. Izvestiya Akad. Nauk Kazah. SSR. 60, Ser. Math. Meh. 3, 1949.

[4] M. Alighanbari, L. Bertuccelli and J. How, Filter-Embedded UAV Task Assignment Algorithms for Dynamic Environments, Proceedings of the AIAA Guidance, Navigation and Control Conference, Providence, Rhode Island, Aug. 2004, AIAA-2004-5251.

[5] M. Alighanbari and J. P. How, Robust Decentralized Task Assignment for Cooperative UAVs, Proceedings of the AIAA Guidance, Navigation and Control Conference, Keystone, Colorado, Aug. 2006, AIAA-2006-6454.

[6] F. Andersson, H. Mausser, D. Rosen, and S. Uryasev, Credit risk optimization with conditional value-at-risk criterion, Mathematical Programming, 89, pp. 273-291,2001.

[7] Andrijich, S.M. and Caccetta, L.(2001), Solving the multisensor data association problem, Nonlinear Analysis, 47, 5525-5536.

[8] Ph. Artzner, F. Delbaen, J-M. Eber, and D. Heath, Thinking coherently, Risk, 10(11), pp. 68-71, 1997.

[9] Ph. Artzner, F. Delbaen, J-M. Eber, and D. Heath, Coherent measures of risk, Mathematical Finance, 9, pp. 203-228, 1999.

[10] Asmussen, S., Kroese, D.P., and Rubinstein, R.Y., HEAVY TAILS, IMPORTANCE SAMPLING AND CROSS-ENTROPY, Stochastic Models, 21(1), 57-76, 2005.

[11] Atteia M., Elqortobi A., Quasi-convex duality, in A. Auslender et al. (eds.), Optimisation and Optimal Control, Proc. Conference Oberwolfach March 1980, Lecture notes in Control and Inform, Sci. 30, 3-8, Springer-Verlag, Berlin 1981.

[12] Balas, E. and M.J. Saltzman (1991). An Algorithm for the Three-Index Assignment Problem. OPERATIONS RESEARCH 39(1): 150-161.

[13] H.-J. Bandelt, A. Maas and F.C.R. Spieksma, Local Search Heuristics for Multi-Index Assignment Problems with Decomposable Costs, The Journal of the Operational Research Society Vol. 55, No. 7, Part Special Issue: Local Search (Jul., 2004), pp. 694-704

[14] Beard, R., McLain, T., and Goodrich, M. (2000), Coordinated target assignment and intercept for unmanned air vehicles. Proc. ICRA'2000, pages 2581-2586.

[15] Beard, R., McLain, T., Goodrich, M., and Anderson, E., Coordinated target assignment and intercept for unmanned air vehicles. IEEE Trans. On Robotics and Automation.

[16] R.W. Beard and V. Stepanyan, Synchronization of Information in Distributed Multiple Vehicle Coordinated Control, Proceedings of the IEEE Conference on Decision and Control, Maui, Hawaii, Dec. 2003, pp. 2029-2034.

[17] J. Bellingham, A. Richards, J. How, Receding Horizon Control of Autonomous Aerial Vehicles, to appear in the IEEE American Control Conference, May 2002.

[18] J. Bellingham, M. Tillerson, A. Richards, and J. P. How, Multi-Task Allocation and Path Planning for Cooperative UAVs, Cooperative Control: Models, Applications, and Algorithms, Editors: S. Butenko, R. Murphey, and P. M. Pardalos, Kluwer Academic Publishers, 2003, pp. 23-41.

[19] A. Ben-Tal and A. Nemirovski, Robust Solutions of Uncertain Linear Pro- grams, Operations Research Letters, 1999, vol. 25, pp. 1-13.

[20] S. Benati and R. Rizzi, A mixed integer linear programming formulation of the optimal mean/value-at-risk portfolio problem, European Journal of Operational Research, 176, pp. 423-434, 2007.

[21] S.J. Benkoski, M.G. Monticino, and J.R. Weisinger, A Survey of the Search Theory Literature, Naval Research Logistics, Vol. 38, pp. 469-494, 1991.

[22] D. Bertsimas and M. Sim, Robust Discrete Optimization and Network Flows, Mathematical Programming, Series B, 2003, vol. 98, pp. 49-71.

[23] D. Bertsimas, D. Pachamanova, and M. Sim, Robust linear optimization under general norms, Operations Research Letters, 2004, vol. 32, pp. 510-516.

[24] D. Bertsimas, K. Katarajan and C. Teo, Probabilistic Combinatorial Optimizations: Moments, Semidefinite Programming, and Asymptotic Bounds, SIAM Journal of Optimization, 2005, vol. 15, pp. 185-209.

[25] L. Bertuccelli, M. Alighabari and J. P. How, Robust Planning for Coupled Cooperative UAV Missions, Proceedings of the IEEE Conference on Decision and Control, Paradise Island, Bahamas, Dec. 2004, pp. 2917-2922.

[26] P.T. de Boer, D.P. Kroese and R.Y. Rubinstein. Estimating buffer overflows in three stages using cross-entropy. In Proceedings of the 2002 Winter Simulation Conference, San Diego, pages 301-309, 2002.

[27] P.T. de Boer, D.P. Kroese and R.Y. Rubinstein, A Fast Cross-Entropy Method for Estimating Buffer Overflows in Queueing Networks, Management Science, 50(7), pp. 883-895, 2004.

[28] P.-T. de Boer, D.P. Kroese, S. Mannor and R.Y. Rubinstein, A Tutorial on The Cross-Entropy Method, Annals of Operations Research, 134:19-67, 2005.

[29] S.S. Brown, Optimal search for a moving target in discrete time and space, Operations Research 32(5), pp. 1107-1115, 1979.

[30] R.E. Burkard and E. Çela. Quadratic and three-dimensional assignment problems. In M. Dell'Amico, F. Maffioli, and S. Martello, editors, Annotated Bibliographies in Combinatorial Optimization. 1997. Wiley, Chichester, pp. 373-392.

[31] R.E. Burkard and E. Çela, Linear Assignment Problems and extensions. In: Du. D., Pardalos, P.M. (eds.) Handbook of Combinatorial Optimization, pp. 75-149. Kluwer Academic, Dordrecht (1999). Chap. 21.

[32] Cadre, J.P., Soiris, G., Searching Tracks, IEEE Transactions on Aerospace and Electronic systems, Vol. 36, No.4, 1149-1166, 2000.

[33] Cambini, R. and Salvi, F., A branch and reduce approach for solving a class of low rank d.c. programs, J. Comput. Appl. Math. 233, 2 (Nov. 2009), 492-501.

[34] R. Campbell, R. Huisman, and K. Koedijk, Optimal portfolio selection in a value-at-risk framework, Journal of Banking & Finance, 25, pp. 1789-1804, 2001.

[35] Çela, E.: Assignment problems. In: Pardalos, P.M., Resende, M.G.C. (eds.) Handbook of Applied Optimization, pp. 661-678. Oxford University Press, New York (2002). Chap. 17.9

[36] Cerny, V., Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm, Journal of Optimization Theory and Applications 45: 41-51, 1985.

[37] M.S. Cheon, S. Ahmed, and F. Al-Khayyal, A branch-reduced-cut algorithm for the global optimization of probabilistically constrained linear programs, Mathematical Programming, 108, pp. 617-634, 2006.

[38] Champagne, L., Carl, E.G. and Hill, R., Search Theory, Agent-Based Simulation, and U-Boats in the Bay of Biscay. Proceedings of the 2003 Winter Simulation Conference, 991-998, 2003.

[39] Clemons, W., Grundel, D., Jeffcoat, Applying simulated annealing on the multidimensional assignment problem, In: Proceeding of the 2nd Cooperative Control and Optimization Conference. (2003)

[40] G. Consigli, Tail estimation and mean-VaR portfolio selection in market subject to financial instability, Journal of Banking and Finance 25 (2002) 1355-1382.

[41] Costa A., Dafydd O., Kroese D., Convergence properties of the cross-entropy method for discrete optimization, Operations Research Letters, 35(5): 573-580, 2007.

[42] J. Curtis, Cooperative Instability: The Churning Effect and its Causes, Recent Developments in Cooperative Control and Optimization, Editors: S. Butenko, R. Murphey, and P. M. Pardalos, Kluwer Academic Publishers, 2004, pp. 105-116.

[43] Chandler, P. and Pachter, M. (1998). Research issues in autonomous control of tactical uavs. Proc. ACC'1998, pages 394-398.

[44] Chandler, P., Rasmussen, S., and Pachter, M. (2000). UAV cooperative path planning. Proc. GNC'2000, pages 1255-1265.

[45] Chandler, P. and Pachter, M. (2001). Hierarchical control for autonomous teams. Proc. GNC'2001, pages 632-642.

[46] Chandler, P., Pachter, M., and Rasmussen, S. (2001). UAV cooperative control. Proc. ACC'2001.

[47] P.R. Chandler, M. Pachter, S.R. Rasmussen, and C. Schumacher: Multiple Task Assignment for a UAV Team. AIAA Guidance, Navigation, and Control Conference, Monterey, CA, Aug. (2002).

[48] Chandler, P. et al. (2002). Complexity in uav cooperative control. Proc ACC'2002.

[49] Dambreville F., Cross-entropy method: convergence issues for extended implementation. In: Proceedings of the rare event simulation conference (RESIM 2006), Bamberg, Germany, 2006.

[50] Dobbie, J.M., Search theory: A sequential approach, Naval Research Logistics Quarterly, 10(1): 323-334, March 1963.

[51] Dobbie, J.M., Transfer of detection contacts to tracking contacts in surveillance. Operations Research, 14 (5): 791-800, September-October 1966.

[52] Dobbie, J.M., Some Search Problems with False Contacts, Operations Research, 21: 907-925, July-August 1973.

[53] Dobbie, J.M., A Two-Cell Model of Search for a Moving Target, Operations Research, 22: 79-92, January-February 1974.

[54] Dobbie, J.M., Search for an Avoiding Target, SIAM Journal on Applied Mathematics Vol. 28, No. 1, pp. 72-86, Jan., 1975.

[55] U. Dubin, The cross-entropy method for combinatorial optimization with applications, Master's thesis, The Technion, Israel Institute of Technology, Haifa, June 2002.

[56] U. Dubin, Application of the cross-entropy method for image segmentation. Annals of Operations Research, 2004. Submitted.

[57] Y. Ephraim and N. Merhav, Hidden Markov processes, IEEE Transactionson Information Theory 48, pp. 1518-1569, 2002.

[58] P. Embrechts, C. Klüppelberg, T. Mikosch, Modelling Extremal Events, Springer, Berlin, 1997.

[59] Frost, J.R.(1999c), Principles of search theory, part III: Probability density distributions, Response, 17(3), pp. 1-10, 1999.

[60] Gal, S., Search Games. New York: Academic Press, 1980.

[61] Gilbert, K. C. and R. B. Hofstra (1988), Multidimensional assignment problems, Decision Sciences 19(2): 306-321.

[62] M. Gilli and E. Kellezi, A global optimization heuristic for portfolio choice with VaR and expected shortfall, in Computational methods in decision-making, economics and finance, vol. 74 of Applied Optimization, Kluwer, pp. 167-183, 2002.

[63] M. Gilli, E.Kellezi,and H. Hysi, A data-driven optimization heuristic for downside risk minimization, The Journal of Risk, 8, pp. 1-18, 2006.

[64] J.C. Gittins, An application of control theory to a game of hide and seek, International Journal of Control, 30(6): 981-987, 1979.

[65] J.C. Gittins and D. M. Roberts, Search for an intelligent evader concealed in one of an arbitrary number of regions, Naval Research Logistics Quarterly, Vol. 26, No. 4, pp. 657-666, 1979.

[66] David Glade, Unmanned Aerial Vehicles: Implications for Military Operations, Occasional Paper No. 16, Center for Strategy and Technology, Air War College, 325 Chennault Circle, Maxwell AFB, Montgomery, Alabama 36112, July 2000.

[67] F. Glover and M. Laguna (1997), Tabu Search. 408 pages. ISBN 0-7923-9965-X, ISBN 0-7923-8187-4, 1997.

[68] Don Grundel, Carlos A. Oliveira, Panos M. Pardalos, and Eduardo Pasiliao. 2005. Asymptotic Results for Random Multidimensional Assignment Problems. Comput. Optim. Appl. 31, 3 (July 2005), 275-293.

[69] Don A. Grundel and Panos M. Pardalos. 2005. Test Problem Generator for the Multidimensional Assignment Problem. Comput. Optim. Appl. 30, 2 (February 2005), 133-146.

[70] J. de Guenin, Optimum distribution of effort: an extension of the Koopman theory, Operations Research 9 (1) (1961), pp. 1-7.

[71] G. Gutin and D. Karapetyan (2009) Local Search Heuristics for the Multidimensional Assignment Problem. In Graph Theory, Computational Intelligence and Thought, Marina Lipshteyn, Vadim E. Levit, and Ross M. Mcconnell (Eds.). Lecture Notes In Computer Science, Vol. 5420. Springer-Verlag, Berlin, Heidelberg 100-115.

[72] Haley, K.B., and Stone, L. D. (Eds.). (1980). Search Theory and Applications. New York: Plenum Press.

[73] Harman P., On functions representable as a difference of convex functions. Pacific J. Math, 9, 707-713 (1959).

[74] Hiriart-Urruty J.B., Lemarechal C.: Convex Analysis and Minimization Algorihtms, Springer, Berlin (1993).

[75] Hoang Tuy, Concave programming under linear constraints, Translated Soviet Mathematics, 5 (1964), pp. 1437-1440.

[76] Hoang Tuy, Global Optimization : Deterministic Approaches, 2nd revised edition, Springer-Verlag, Berlin, 1993.

[77] Hoang Tuy, DC Optimisation : Theory, Methods and Algorithms, Handbook of Global Optimisation, Horst and Pardalos eds, Klaver Academic Publishers, pp. 149–216, 1995.

[78] Hoang Tuy, Convex Analysis and Global Optimization, Kluwer Academic Publishers, 1998.

[79] R. Hochreiter, An evolutionary computation approach to scenario-based risk-return portfolio optimization for general risk measures, in EvoWorkshops 2007, vol. 4448 of Lecture Notes in Computer Science, Springer, 2007, pp. 199-207.

[80] R. Hohzaki and K. Iida, A concave minimization problem with double layers of constraints on the total amount of resources, Journal of the Operations Research Society of Japan 43(1):109-127 (2000).

[81] Holland, J.H. (1975), Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor.

[82] Horst R., A general class of branch-and-bound methods in global optimization with some new approaches for concave minimization, J. Optim. Theory Appl. 51, no. 2, 271-291 (1986).

[83] Horst R., Deterministic global optimization with partition sets whose feasibility is not known: application to concave minimization, reverse convex constraints, DC programming, and Lipschitzian optimization, J. Optim. Theory Appl. 58, no. 1, 11-37 (1988).

[84] Horst R., Hoang T., Global Optimization: Deterministic Approaches, Third edition, Springer, Berlin 1996.

[85] Horst R., Pardalos P.M., Thoai N.V., Introduction to Global Optimization - Second Edition, Kluwer Academic Publishers, Netherlands. (2000)

[86] Iida, K., Studies on the optimal search plan. In Lecture Notes in Statistics, vol. 70. New York: Springer-Verlag, 1992.

[87] Jacques, D. (1998). Search, classification and attack decisions for coop- erative wide area search munitions. Proc. Cooperative Optimization and Control Workshop.

[88] P. Jorion, Value at Risk: The New Benchmark for Controlling Market Risk, McGraw-Hill, 2000.

[89] M. Jun and D. Jeffcoat, Convergence Properties of Continuous-Time Markov Chains with Application to Target Search, Proceedings of the IEEE American Control Conference, Portland, Oregon, June 2005, pp. 662-667.

[90] Yoshihiro Kanno, Makoto Ohsaki, Optimization-based stability analysis of structures under unilateral constraints, International Journal for Numerical Methods in Engineering 2009, Volume 77, Issue 1 (p 90-125).

[91] R. Kastner, C. Hsieh, M. Potkonjak, and M. Sarrafzadeh, On the Sensitivity of Incremental Algorithms for Combinatorial Auctions, Proceedings of the 4th IEEE International Workshop on Advanced Issues of E-Commerce and Webbased Information Systems (WECWIS), Newport Beach, California, June 2002, pp. 81-88.

[92] E. King, M. Alighanbari, Y. Kuwata, and J. How, Coordination and Control Experiments on a Multi-Vehicle Testbed, Proceedings of the IEEE American Control Conference, Boston, Massachusetts, June 2004, pp. 5315-5320.

[93] Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P. (1983). Optimization by Simulated Annealing. Science 220 (4598): 671-680.

[94] Klerk, E. de (2002). Aspects of semidefinite programming: Interior point algorithms and selected applications. Dordrecht: Kluwer Academic Publishers. (Applied optimization, ISSN 1384-6485, 65).

[95] Konno, H., Phan Thien Thach and Hoang Tuy. (1999). Optimization on Low Rank Nonconvex Structures. Kluwer Academic

[96] B.O. Koopman, The Theory of Search. I. Kinematic Bases, Operational Research, 4 (3): 324-346, Jun., 1956.

[97] B.O. Koopman, The Theory of Search. II. Target Detection, Operational Research, 4 (5): 503-531, October 1956.

[98] B.O. Koopman, The Theory of Search. III. The Optimum Distribution of Searching Effort, Operations Research, 5(5): 613-626, (Oct., 1957).

[99] B.O. Koopman, Search and its optimization, Mathematical Monthly 7 (1979), pp. 527-540.

[100] Koopman, B.O., Search and Screening: General Principles with Historical Applications, Pergamon Press, New York, 1980.

[101] Koopman, B.O., Search and Screening: General Principle with Historical Applications. Alexandria, VA: MORS Heritage Series, 1999.

[102] Kroese, D.P., and Rubinstein, R.Y. (2004). The Transform Likelihood Ratio Method for Rare Event Simulation with Heavy Tails. Queueing Systems, 46, pp 317–351.

[103] Kroese, D.P., Porotsky, S. and Rubinstein, R.Y.: The cross-entropy method for continuous multi-extremal optimization. Methodology and Computing In Applied Probability 8(3) (2006) 383–407.

[104] P. Krokhmal, J. Palmquist, and S. Uryasev, Portfolio Optimization with Conditional Value-At-Risk Objective and Constraints, The Journal of Risk, 2002, vol. 4, no. 2.

[105] P. Krokhmal, R. Murphey, P. Pardalos, S. Uryasev, and G. Zrazhevsky, Robust Decision Making: Addressing Uncertainties in Distributions, In Cooperative Control: Models, Applications and Algorithms, Kluwer Academic Publishers, 2003, pp. 165-185.

[106] R. Kumar, D. Hyland, Control Law Design Using Repeated Trials, IEEE American Control Conference, Arlington, VA, June 25-27, 2001, pp. 837-842.

[107] Kuroki, Y. and T. Matsui (2009). An approximation algorithm for multidimensional assignment problems minimizing the sum of squared errors. Discrete Applied Mathematics 157(9): 2124-2135.

[108] Y. Kuwata, A. Richards, T. Schouwenaars, and J. P. How, Decentralized Robust Receding Horizon Control for Multi-Vehicle Guidance, Proceedings of the IEEE American Control Conference, Minneapolis, Minnesota, June 2006, pp. 2047-2052.

[109] Larsen, N., Mausser, H., Uryasev, S., Algorithms for optimization of value-at-risk. In: Pardalos, P., Tsitsiringos, V. (eds.) Financial Engineering, e-Commerce and Supply Chain, pp. 129–157. Kluwer Academic Publishers, Dordrecht, Netherlands (2002).

[110] H.A. Le Thi. Analyse numérique des algorithmes de l'optimisation DC. Approches locale et globale. Codes et simulations numériques en grande dimension. Applications. Thèse de Doctorat de l'Université de Rouen, 1994.

[111] H.A. Le Thi, Contribution à l'optimisation non convexe et l'optimisation globale: Théorie, Algorithmes et Applications Habilitation à Diriger des Recherches, Université de Rouen, 1997.

[112] H.A. Le Thi, An efficient algorithm for globally minimizing a quadratic function under convex quadratic constraints, Mathematical Programming, Ser. A, Vol. 87(3), pp. 401–426, 2000.

[113] H.A. Le Thi, Solving large scale Molecular distance geometry problem by a smoothing technique via the Gaussian transform an DC programming, Journal of Global Optimization, Vol. 27, pp. 375–397, 2003.

[114] http://lita.sciences.univ-metz.fr/ lethi/english/index.html

[115] Le Thi H.A., Pham Dinh T., and Le Dung M., Numerical solution for optimization over the efficient set by d.c. optimization algorithms, Operations Research Letters 19(3): 117-128.

[116] Le Thi H.A., Pham Dinh T., and Le Dung M. (1998), A Combined DC Optimization - Ellipsoidal Branch-and-Bound Algorithm for Solving Nonconvex Quadratic Programming Problems. J. Combin. Optim. 2(1), 9-28.

[117] Le Thi H.A., Pham Dinh T., Le Dung M., Exact penalty in d.c. programming. Vietnam Journal of Mathematics, Vol. 27 (2), 169-178 (1999).

[118] Le Thi H.A. and Pham Dinh T., Large Scale Global Molecular Optimization from exact distance matrices by a DC optimization approach, SIAM Journal on Optimization, Volume 14, Number 1 (2003), pp. 77-114.

[119] Le Thi H.A., Pham Dinh T., and Le Dung M., Simplicially constrained D.C. Optimization formulation for optimizing over the efficient and weakly efficient sets, Journal of Optimization Theory and Applications, Vol 117, No. 3, pp. 503-531(2003).

[120] Le Thi H.A. and Pham Dinh T., The DC (difference of convex functions) Programming and DCA revisited with DC models of real world nonconvex optimization problems, Annals of Operations Research (2005), Vol 133, pp. 23-46.

[121] Le Thi H.A., Pham Dinh T., and Huynh Van N., Exact penalty techniques in DC programming. Technical report, National Institute for Applied Sciences - Rouen, France (2007).

[122] Le Thi H.A., Pham Dinh T., and Huynh Van N., Exact Penalty and Error Bound in DC Programming, to appear in Journal of Global Optimization.

[123] Li, S.-M., Boskovic, J., Seereeeram, S., Prasanth, R., Amin, R., Mehra, R., and Beard, R. a. M. T. (2002). Autonomous hierarchical control of multiple unmanned combat air vehicles (ucavs). Proc. ACC'2002, pages 274-279.

[124] P. Mahey, T.Q. Phong and H.P.L Luna, Separable convexification and DCA techniques for capacity and flow assignement problems, RAIRO-Recherche Opérationnelle 35 (2001), pp.269-281.

[125] H.M. Markowitz, Portfolio selection, Journal of Finance 25 (1) (1952) 71-79.

[126] Mausser, H. and D. Rosen (1998), Beyond VaR: From Measuring Risk to Managing Risk, Algo Research Quarterly, Vol.1, No.2, 5-20.

[127] McLain, T. and Beard, R. (2000), Trajectory planning for coordinated rendezvous of unmanned air vehicles. Proc. GNC'2000, pages 1247-1254.

[128] T. McLain, P. Chandler, S. Rasmussen, M. Pachter, Cooperative Control of UAV Rendezvous, IEEE American Control Conference, Arlington, VA, June 25-27, 2001, pp. 2309-2314.

[129] McLain, T., Beard, R., and Kelsey, J. (2002), Experimental demonstration of multiple robot cooperative target intercept. Proc GNC'2002.

[130] Mladenovic, N., and P. Hansen (1997), Variable neighborhood search, Comput. Oper. Res. 24(11): 1097-1100.

[131] Moitra, A., Szczerba, R., Didomizio, V., Hoebel, L., Mattheyses, R., and Yamrom, B. (2001), A novel approach for the coordination of multivehicle teams. Proc. GNC'2001, pages 608-618.

[132] Murphey, R., Pardalos,P., and Pitsoulis, L.(1998), A greedy randomized adaptive search procedure for the multitarget multisensor tracking problem, in: DIMACSSeries, volume 40, 277-302, American Mathematical Society.

[133] Nguyen Van Thoai and Hoang Tuy, Convergent Algorithms for Minimizing a Concave Function, Mathematics of Operations Research, Vol. 5, No. 4 (Nov., 1980), pp. 556-566.

[134] Nguyen Van Thoai (1998), Global Optimization Techniques for Solving the General Quadratic Integer Programming Problem, Computational Optimization and Applications 10(2): 149-163.

[135] M. Le Dung, P. Thai Quynh, T. Pham Dinh, Decoposition methods for solving a class of nonconvex programming problems dealing with bilinear and quadratic function. Computational Optimization and Applications 4, 203-216 (1995).

[136] J. Mulvey, R. Vanderbei, and S. Zenios, Robust optimization of large-scale systems, Operations Research, 1995, vol. 43, pp. 264-281.

[137] Niu Y.S., Programmation DC&DCA en Optimisation Combinatoire et Optimization Polynomiale via les Techniques de SDP. Codes et Simulations Numériques, Thèse de Doctorat de l'INSA de Rouen, May 2010.

[138] Oliveira, C.A.S. and P.M. Pardalos (2004). Randomized parallel algorithms for the multidimensional assignment problem. Appl. Numer. Math. 49(1): 117-133.

[139] D. Pachamanova, A Robust Optimization Approach to Finance, Ph.D. Thesis, MIT Operations Research Center, 2002.

[140] J.S. Pang and S. Leyffer, On the global minimization of the value-at-risk, Optimization Methods and Software 19 (2004) 611–631.

[141] C.H. Papadimitriou, and K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity. Prentice-Hall, Englewood Cliffs, NJ (1982).

[142] P.M. Pardalos and H. Wolkowicz (Eds.), Topics in Semidefinite and Interior Point Methods, Fields Institute Communications 18, AMS, Providence, Rhode Island, 1998.

[143] Pardalos, P.M., Pitsoulit, L.S. (eds.), Nonlinear Assignement Problems: Algorithms and Applications. Combinatorial Optimization, vol. 7, Kluwer Academic, Dordrecht (2000).

[144] Pasiliao, E.L. (2003), Algorithms for Multidimensional Assignment Problems, Ph.D. thesis,. Department of Industrial and Systems Engineering, University of Florida.

[145] Pasiliao, E.L. (2010), Local Neighborhoods for the Multidimensional Assignment Problem. Dynamics of Information Systems. M. J. Hirsch, P. M. Pardalos and R. Murphey, Springer New York. 40: 353-371.

[146] Passino, K. (2002), An introduction to research challenges in cooperative control for uninhabited autonomous vehicles.

[147] Pierskalla, W.(1968), The multidimensional assignment problem, Operations Research, 16, pp. 422-431.

[148] Penot J.P., Duality for anticonvex programs, Journal of Global Optimization archive. Volume 19, Issue 2, 163-182 (2001).

[149] G.Ch. Pflug, Some remarks on the Value-at-Risk and the Conditional Value-at-Risk, in Probabilistic constrained optimization, Nonconvex Optimization and its Applications, Kluwer, 49 (2000) 272–281.

[150] G.Ch. Pflug and W. Romisch: Modeling, Measuring and Managing Risk. World Scientific, Singapore (2007).

[151] Pham Dinh T., Algorithmes de calcul du maximum des formes quadratiques sur la boule unité de la norme du maximum, Séminaire d'analyse numérique, Grenoble, No. 247 (1976).

[152] T. Pham Dinh, Duality in DC (difference of convex functions) optimization. Subgradient methods Trends in Mathematical Optimization, International Series of Numer Math., Vol. 84, pp. 277–293, 1988.

[153] T. Pham Dinh and H.A. Le Thi, Stabilité de la dualité lagrangienne en optimisation DC (différence de deux fonctions convexes), C.R. Acad. Paris, P.318, Série I, pp. 379–384, 1994.

[154] T. Pham Dinh and H.A. Le Thi, Lagrangian stability and global optimality in nonconvex quadratic minimization over Euclidean balls and spheres, Journal of Convex Analysis, Vol. 2, pp. 263–276, 1995.

[155] T. Pham Dinh and H.A. Le Thi, DC optimization algorithms for globally minimizing nonconvex quadratic forms on Euclideans balls and spheres, Operations Research Letters, Vol. 19, pp. 207–216, 1996.

[156] T. Pham Dinh and H.A. Le Thi, Convex analysis approach to d.c. programming: Theory, Algorithms and Applications Acta Mathematica Vietnamica, dedicated to Professor Hoang Tuy on the occasion of his 70th birthday, Vol. 22 (1), pp. 289–355, 1997.

[157] T. Pham Dinh and H.A. Le Thi, DC optimization algorithms for solving the trust region subproblem, SIAM Journal of Optimization, 8(2), pp. 476–505, 1998.

[158] Pham Dinh T., Le Thi H.A., DC Programming. Theory, Algorithms, Applications: The State of the Art. First International Workshop on Global Constrained Optimization and Constraint Satisfaction, Nice, October 2-4, 2002.

[159] Pham Dinh T., Nguyen Canh N., and Le Thi H.A., An efficient combined DCA and B&B using DC/SDP relaxation for globally solving binary quadratic programs, J. of Global Optimization 48(4): 595-632 (2010).

[160] Phan Thien Thach, DC Sets, D.C. Functions and Nonlinear Equations, Math, Programming 58, 415-428, (1993a).

[161] Phan Thien Thach, Global Optimality Criteria and Duality with Zero Gap in Nonconvex Optimization Problems, SIAM J. Math. Anal. 24, 1537-1556, (1993b).

[162] Phan Thien Thach, A Nonconvex Duality with Zero Gap and Applications, SIAM J. Optim. 4, 44-64, 1994.

[163] Gianni Di Pillo, Almerico Murli, On Efficient Large Margin Semisupervised Learning: Method and Theory, Journal of Machine Learning Research 10 (2009) 719-742.

[164] M. Polycarpou, Y. Yang, and K. Passino, A Cooperative Search Framework for Distributed Agents, Proceedings of the IEEE International Symposium on Intelligent Control, Mexico City, Mexico, Sept. 2001, pp. 1-6.

[165] Polycarpou, M., Yang, Y., and Passino, K. (2002). Cooperative control of distributed multi-agent systems. IEEE Control Systems Magazine.

[166] Poore AB, Rijavec N (1993) A Lagrangian Relaxation Algorithm for Multidimensional Assignment Problems Arising from Multitarget Tracking. SIAM Journal of Optimization 3(3):544-563.

[167] Poore AB (1994) Multidimensional assignment formulation of data association problems arising from multitarget and multisensor tracking. Computation Optimization and Applications 3:27-54.

[168] Poore AB, Robertson III AJ (1997) A New Lagrangian Relaxation Based Algorithm for a Class of Multidimensional Assignment Problems. Computational Optimization and Applications 8(2):129-150.

[169] Pusztaszeri, J.F., Rensing, P.E., and Liebling, T.M. (1996), Tracking elementary particles near their primary vertex: acombinatorial approach, Journal of Global Optimization, 9 (1), 41-64.

[170] J.H. Reif, Complexity of the Mover's Problem and Generalizations, in the proceedings of the 20th IEEE Symposium on the Foundations of Computer Science, IEEE, Washington DC, pp. 421-427, 1979.

[171] A. Richards, J. Bellingham, M. Tillerson, and J. How, Coordination and Control of Multiple UAVs, submitted for publication at the 2002 AIAA Guidance, Navigation, and Control Conference.

[172] Richardson, H.R., Search theory. Encyclopedia of Statistical Science, 8 (1988), 314-321, Kotz, Johnson and Read eds.

[173] Rockafellar R.T., Convex analysis, Princeton University Press, N.J. (1970)

[174] R.T. Rockafellar and S. Uryasev, Optimization of Conditional Value-at-Risk, The Journal of Risk, 2(2000), pp. 21-41.

[175] R.Y. Rubinstein and A. Shapiro, Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization via the Score Funtion Method (Wiley, 1992)

[176] Rubinstein R.Y., Optimization of cumputer simulation models with rare events, European Journal of Operation Reseach, 99, 89-112, 1997.

[177] Rubinstein R.Y., The simulated entropy method for combinatorial and continuous optimization, Methodology and Computing in Applied Probability, 2, 127-190, 1999.

[178] Rubinstein R.Y., Combinatorial optimization, cross-entropy, ants and rare events. In S. Uryasev and P. M. Pardalos, editors, Stochastic Optimization: Algorithms and Application, pages 304-358, Kluwer, 2001.

[179] Rubinstein R.Y., The cross-entropy method and rare-events for maximal cut and bipartition problems. ACM Transactions on Modelling and Computer Simulation, 12(1):27-53, 2002.

[180] Rubinstein R.Y., Kroese D, The cross-entropy method: a unified approach to combinatorial optimization, Monté Carlo simulation, and machine learning, Berlin: Springer; 2004.

[181] Rubinstein, R.Y., Kroese, D.P. (2007), Simulation and the Mont Carlo Method (2nd ed.), New York: John Wiley & Sons. ISBN 9780470177938.

[182] Schouwenaars, T., De Moor, B., Feron, E., and How, J. (2001). Mixed integer programming for multi-vehicle path planning. Proc. ACC'2001.

[183] T. Schouwenaars, E. Feron, and J. P. How, Safe Receding Horizon Path Planning for Autonomous Vehicles, Proceedings of the 40th Allerton Conference on Communication, Control, and Computation, Monticello, Illinois, Oct. 2002.

[184] Schumacher, C., Chandler, P., and Rasmussen, S. (2001). Task allocation for wide area search munitions via network flow optimization. Proc. GNC'2001, pages 619-626.

[185] T. Schüle, C. Schnörr, S. Weber, and J. Hornegger . Discrete tomography by convex-concave regularization and d.c. programming. Discr. Appl. Math., 151:229-243, Oct 2005.

[186] C. Simonin, J.P. Le Cadre, and F. Dambreville, A hierarchical approach for planning a multisensor multizone search for a moving target, Computers and Operations Research 36(7): 2179-2192.

[187] L. Singh and J. Fuller, Trajectory Generation for a UAV in Urban Terrain, Using Nonlinear MPC, IEEE American Control Conference, Arlington, VA, June 25-27, 2001, pp. 2301-2308.

[188] F.C.R. Spieksma: Multi Index Assignment Problems: Complexity, Approximation, Applications. In: Pardalos, P.M., Pitsoulit, L.S. (eds.) Nonlinear Assignement Problems: Algorithms and Applications. Combinatorial Optimization, vol. 7, pp. 1-12. Kluwer Academic, Dordrecht (2000). Chap. 1

[189] L.D. Stone, Necessary and suffcient conditions for optimal search plans for moving targets, Mathematics of Operations Research4 (4)(1979), pp. 431-440. MathSciNet.

[190] Stone, L.D., What's happened in search theory since the 1975 Lanchester prize? Operations Research, 37, 3 (May-June 1989), 501-506.

[191] Stone, L.D., Theory of Optimal Search (2nd ed.). Arlington, VA: Operations Research Society of America, ORSA Books, 1989.

[192] Storms, P.P.A. and F.C.R. Spieksma (2003). An LP-based algorithm for the data association problem in multitarget tracking. Computers & Operations Research 30(7): 1067-1085.

[193] W.R. Stromquist and L.D. Stone, Constrained optimization of functionals with search theory applications, Mathematics of Operations Research 6(4)(1981), pp. 518-527.

[194] Kirk Sturtz, Gregory Arnold and Matthew Ferrara, DC optimization modeling for shape-based recognition, Proc. SPIE 7337, 73370O (2009); doi:10.1117/12.820293

[195] Tan, K., Lee, L., Zhu, Q., and Ou, K. (2002). Heuristic methods for vehicle routing problem with time windows. Intelligent in Engineering, pages 281-295.

[196] Thai Quynh P., Le Thi H.A., and Pham Dinh T., On the global solution of linearly constrained indefinite quadratic minimization problems by decomposition branch and bound method. RAIRO, Recherche Opérationnelle 30(1), 31-49.

[197] J. Tierno and A. Khalak, Frequency Domain Control Synthesis for Time-Critical Planning. Proceedings of the IEE European Control Conference, Cambridge, UK, Sept. 2003

[198] Toland J.K., Duality in Nonconvex Optimization, J. Mathematical Analysis and Applications, 58, 415-428 (1978).

[199] S. Uryasev, Conditional Value-at-Risk: Optimization algorithms and applications, Financial Engineering News, 14 (2000), pp. 1-5.

[200] Van Keuk, G. (1998) Sequential track extraction. IEEE Transactions on Aerospace and Electronic Systems, 34, 4 (Oct.1998), 1135-1148.

[201] L. Vandenberghe and S. Boyd, Semidefinite Programming, SIAM Review 38, March 1996, pp. 49-95.

[202] Veenman, C.J., Hendriks, E.A., and Reinders, M.J.T. (1998), A fast and robust point tracking algorithm, in: Proceedings of the Fifth IEEE International Conferenceon Image Processing, 653-657, Chicago.

[203] D.H. Wagner and L.D. Stone, Necessity and Existence Results on Constrained Optimization of Separable Functionals by a Multiplier Rule, SIAM J. Control Volume 12, Issue 3, pp. 356-372 (August 1974).

[204] Wagner, D.H., Mylander, W. C., and Sanders, T. J. (Eds.), Naval Operations Analysis (3rd ed.). Annapolis, MD: Naval Institute Press, 1999.

[205] Washburn, A.R.(1983) Search for a moving target, The FAB algorithm, Operations Research, 31, 4(July-Aug. 1983), 739-751.

[206] Washburn, A.R., Search and detection (2nd ed.). Arlington, VA: Operations Research Society of America, ORSA Books, 1989.

[207] Washburn, A.R., Two Persons Zero-Sum Games (2nd ed.). Topics in Operations Research, INFORMS, 1994.

[208] Wozabal, D., Hochreiter, R., Pflug, G. : A d. c. formulation of value-at-risk constrained optimization. Tech. Rep. TR2008-01, Department of Statistics and Decision Support Systems, University of Vienna, Vienna (2008).

[209] Wozabal, D., Value-at-Risk optimization using the difference of convex algorithm, OR Spectrum: 1-23, online first (2010).

[210] Y. Yang, A. Minai, and M. Polycarpou, Evidential Map-Building Approaches for Multi-UAV Cooperative Search, Proceedings of the IEEE American Control Conference, Portland, Oregon, June 2005, pp. 116-121.

[211] Kai Zhang, Ivor W. Tsang, James T. Kwok, Maximum margin clustering made practical, IEEE Transactions on Neural Networks, v.20 n.4, p.583-596, April 2009.

**Abstract:** In this thesis we focus on developing deterministic and heuristic approaches for solving some classes of optimization problems in Finance, Assignment and Search Theory. They are large-scale nonconvex optimization problems. Our approaches are based on DC programming & DCA and the Cross-Entropy method. Due to the techniques of formulation/reformulation, we have given the DC formulation of considered problems such that we can use DCA to obtain their solutions. Also, depending on the structure of feasible sets of considered problems, we have designed appropriate families of distributions such that the Cross-Entropy method could be applied efficiently. This thesis is divided into two parts:

In the part 1, besides the introduction to some classes of nonconvex programs in Finance, Assignment and Search Information, we present the methodologies of DC programming&DCA and the CE method. They are two main tools, which we use to investigate solution methods for our considered problems.

Part 2 is devoted to the resolutions of our considered problems. It contains three sections:

In the first section, we propose a deterministic continuous optimization approach based on DC programming and DCA for solving the problem of planning a multisensor multizone search for a target. Then, we extend the obtained result to the case of moving (Markovian) target by combining the forward-backward split technique and DCA.

In the second one, we apply the CE method for solving the UAV (Unmanned Aerial Vehicles) Task Assignment Problem and the Multidimensional Assignment Problem. We construct appropriate families of discrete distributions on the feasible sets of the problems such that the CE method could be applied efficiently. Particularly, for the UAV Task Assignment Problem, we also give a global approach based on Branch and Bound algorithm for measuring the quality of our CE algorithm.

In the last one, we study both deterministic and heuristic approaches for solving the Value-at-Risk constrained optimization problem. On the one hand, we introduce a new and efficient heuristic approach based on the Cross-Entropy method for its solution, where an appropriate family of continuous distributions on the feasible set of this problem is designed. On the other hand, we reformulate this problem as a polyhedral DC program based on an exact penalty technique, and then apply the DCA for solving this problem.

All these proposed approaches have been implemented by using MATLAB, C/C++. They show very good performance even in very large dimension, for instance, up to 500,000 binary variables and 1000 constraints in the UAV (Unmanned Aerial Vehicles) Task Assignment Problem, up to 3.2 millions binary variables and 100 constraints in the Multidimensional Assignment Problem, or the testing on the daily data of 11 years (from January 1, 2000 to December 31, 2010), i.e., 2759 scenarios of the 87 assets comprising the NYSE US 100 index in the Value-at-Risk constrained optimization problem, etc.

Key words: Search Theory, Hierarchical Optimization, Combinatorial Optimization, DC programming and DCA, Nonlinear Mixed 0-1 Programming, Exact Penalty, UAV, Assignment Problem, Stochastic Programming, Binary Nonlinear Programming, Cross-Entropy (CE) method, Brand and Bound, Portfolio Management, Risk Management, Value-at-Risk, Multidimensional Assignment Problem.

**Résumé:** La présente thèse a pour objectif principal de développer des approches déterministes et heuristiques pour résoudre certaines classes de problèmes d'optimisation en Finance, Affectation et Recherche d'Informations. Il s'agit des problèmes d'optimisation non convexe de grande dimension. Nos approches sont basées sur la programmation DC & DCA et la méthode Cross-Entropy (CE). Grâce aux techniques de formulation/reformulation, nous avons donné la formulation DC des problèmes considérés afin d'obtenir leurs solutions en utilisant DCA. En outre, selon la structure des ensembles réalisables de problèmes considérés, nous avons conçu des familles appropriées de distributions pour que la méthode Cross-Entropy puisse être appliquées efficacement. Cette thèse est divisée en deux parties:

Dans la partie 1, en plus de certaines classes de problèmes d'optimisation en Finance, Affectation et Recherche d'informations, nous présentons la méthodologie de la programmation DC&DCA et la méthode Cross-Entropy. Ce sont les deux outils principaux que nous utilisons pour résoudre des problèmes considérés.

La partie 2 est consacrée à la résolution des problèmes considérés. Elle comprend trois sections:

Dans la première section, nous proposons une approche d'optimisation déterministe continue basée sur la programmation DC et DCA pour résoudre le problème de planification multi-capteurs multizones pour la recherche d'une cible. Puis, nous étendons notre résultat obtenu au cas de la cible mobile (markovienne) en combinant la technique FAB et l'algorithme DCA.

Dans la seconde section, nous appliquons la méthode Cross-Entropy pour résoudre le problème d'affectation des tâches d'UAVs, et le problème d'affectation multi-dimensionnelle. Nous construisons les familles appropriées de distributions discrètes sur des ensembles réalisables des problèmes afin que la méthode CE puisse être appliquée efficacement. Particulièrement, pour le problème d'affectation des tâches d'UAVs, nous présentons aussi une approche globale basée sur l'algorithme de séparation et d'évaluation pour mesurer la qualité de notre algorithme CE.

Dans la dernière section, nous étudions des approches déterministes et heuristiques pour résoudre le problème d'optimisation de portefeuille sous la contrainte de Value-at-Risk. Pour résoudre ce problème, d'une part nous introduisons une approche heuristique nouvelle et efficace basée sur la méthode Cross-Entropy pour, où nous construisons une famille appropriée de distributions continues sur l'ensemble réalisable de ce problème. D'autre part, nous reformulons ce problème comme une programmation DC polyédrale basée sur une

technique de pénalité exacte, et puis nous utilisons DCA.

Les simulations numériques de nos différentes approches ont été réalisées en utilisant Matlab, C/C++. Elles montrent de très bonnes performances même en très grande dimension, par exemple, jusqu'à 500.000 variables binaires et 1000 contraintes dans le problème d'affectation des tâches d'UAVs, jusqu'à 3,2 millions variables binaires et 100 contraintes dans le problème d'affectation multi-dimensionnelle, ou les tests sur les données quotidiennes de 11 ans (du 1 janvier 2000 au 31 décembre 2010), c'est à dire 2759 scénarios des 87 actifs composant l'indice NYSE US 100 dans le problème d'optimisation de portefeuille sous la contrainte de Value-at-Risk, etc.

Mots clés: Recherche d'Informations, Optimisation Hiérarchique, Optimisation Combinatoire, Programmation DC et DCA, Programmation non linéaire Mixte 0-1, Pénalité Exacte, Problème d'affectation, Programmation Stochastique, Programmation non linéaire binaire, Méthode Cross-Entropy, Algorithme de séparation et évaluation, Gestion de portefeuille, Gestion des risques, Value-at-Risk, Problème d'affectation multidimensionnelle.