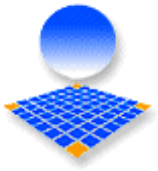


UNIVERSITE MONTPELLIER II
SCIENCES ET TECHNIQUES DU LANGUEDOC

T H È S E

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITE MONTPELLIER II

Discipline	**:**	Mathématiques Appliquées
Formation Doctorale	**:**	Mathématiques
Ecole Doctorale	**:**	Information, Structures et Systèmes

présentée et soutenue publiquement le Vendredi 09 Juin 2006 par

Ivorra Benjamin

OPTIMISATION GLOBALE SEMI-DÉTERMINISTE ET APPLICATIONS INDUSTRIELLES

JURY

M. Bijan Mohammadi	, Univ. Montpellier II	, Directeur
M. Yves Moreau	, Univ. Montpellier II	, Président
M. Olivier Pironneau	, Univ. Paris VI	, Examinateur
M. Angel Manuel Ramos Del Olmo	, Univ. Complutense de Madrid	, Examinateur
M. Laurent Dumas	, Univ. Paris VI	, Examinateur
M. Hedy Attouch	, Univ. Montpellier II	, Examinateur

RAPPORTEURS

M. Olivier Pironneau	, Univ. Paris VI
M. Angel Manuel Ramos Del Olmo	, Univ. Complutense de Madrid

Á ma future épouse, ma famille et mes amis...

Remerciements

Cette thèse représente trois années importantes de ma vie. Ce fut une période de rencontres, d'échanges, de doutes et d'avancées. Le bon déroulement de ce travail, je le dois à beaucoup de personnes et à travers cette page j'espère leur signifier toute ma gratitude.

Je tiens à remercier en premier lieu mon directeur de thèse, Bijan Mohammadi, pour tous les efforts qu'il a fourni. Il a su m'orienter vers des thèmes d'étude diversifiés et passionnants. Il m'a aussi permis de travailler avec de nombreux chercheurs nationaux et internationaux motivés et intéressants. Pour tout ceci je lui exprime toute ma reconnaissance.

Je voudrai aussi remercier Olivier Pironneau et Angel Manuel Ramos Del Olmo d'avoir accepté d'être rapporteurs surtout dans des délais aussi brefs! L'intérêt qu'ils ont montré envers mon travail m'a grandement touché. Je remercie aussi Hedy Attouch, Laurent Dumas et Yves Moreau de m'avoir fait l'honneur de participer à mon jury.

Cette partie ne serait pas complète si je ne remerciais pas tous mes collègues avec lesquels j'ai travaillé durant ces trois années: Jean-Paul Dufour, Patrick Redont, Michel Cuer, Carole Duffet, Damien Isèbe (I3M), Laurent Dumas (Paris VI), Olivier Durand (Alcatel), Yves Moreau (CEM2), Juan Santiago, David Hertzog (Stanford), Larvi Debiane (INRIA), Angel Manuel Ramos-Del Olmo (Madrid), Guillaume Quibel, Rim Terhaoui, Sebastien Delcourt (BNP-Paribas). Ils m'ont apporté un savoir et une aide précieuse et m'ont permis d'aller de l'avant.

Merci aussi à tous les membres administratifs du département, en particulier Bernadette Lacan, Nicole Grachet et Pierrette Arnaud, pour l'aide matérielle et administrative qu'ils m'ont fourni.

Bien sûr tout ce travail n'aurait pu être effectué sans l'appui moral de tous mes amis et les nombreux bon moments passés ensemble, je pense à Nath, Kuuu, Sylvie, Miguel, Ely, Hassène, Khalid, Ricardo ainsi que tous les autres thésards de France ou d'ailleurs (de Thaïlande...). Cette ambiance bonne enfant se retrouvait naturellement dans le célèbre bureau 319 avec mes deux colocataires préférés Julien et Damien. Un grand merci à tous mes amis de Montpellier (Lionnel, Magalie, Sylvain, Mireille, John), Nîmes (Fabrice, Véro, Marie-Agnes), Sahorre (Titi, Laure et Nabil) et Paris (l'équipe de BFI). Je n'oublie pas bien sûr la petite famille de Cagne (Seb, Sego et Anaïs) ainsi que le pingouin finlandais (François) et les grands moments 'sportifs' passés ensemble le samedi soir.

Durant ces trois années la famille a été aussi importante et malgré tous mes déplacements elle a toujours été présente avec moi. Merci papa, maman, mon frère, Marie-Hélène, Romain, Sylvane, Mario, Justine et ma-belle famille pour toute l'affection et l'aide que vous m'avez donné.

Enfin, je voudrai remercier et dédicacer ces années de ma vie à ma future épouse, Tatiana, sans laquelle je n'en serai certainement pas là. Merci pour tout ce que tu as su m'apporter...

Contents

Introduction

Les méthodes d'optimisation ont trouvé, depuis de nombreuses années, une place prépondérante dans de nombreux secteurs industriels. La nature et la diversité des problèmes pouvant être traités en font des outils toujours plus utilisés et demandés. L'enjeu pour un laboratoire comme le notre, étant alors de développer et de proposer des méthodes à la fois rapides et efficaces.

Après avoir rappelé dans le premier chapitre diverses notions mathématiques dont nous nous servons par la suite, nous introduisons dans le second chapitre, une méthodologie originale qui va nous permettre d'améliorer un large choix d'algorithmes d'optimisation classiques, aussi bien au niveau de la rapidité de calcul que de la précision du résultat.

Nous partons du constat suivant: résoudre un problème d'optimisation avec une méthode découlant de la discrétisation d'un système dynamique est équivalent à résoudre un problème à valeurs aux limites. Il est alors naturel d'essayer d'appliquer la théorie sur la résolution des problèmes à valeurs aux limites afin de nous aider à effectuer une optimisation de notre problème initial. Ce qui nous permet d'introduire une nouvelle classe de méthodes d'optimisation globale de type semi-déterministe. Nous pouvons alors étudier plus en détail l'implémentation algorithmique qui en découle, pour des familles d'algorithmes provenant de la discrétisation de systèmes dynamiques d'ordre 1 ou 2.

Une autre idée novatrice introduite dans cette partie est de donner une formulation uniforme pour les méthodes déterministes et stochastiques. En effet en utilisant, une description matricielle, les algorithmes génétiques peuvent être vus comme provenant de la discrétisation de systèmes dynamiques stochastiques et ils peuvent donc entrer dans le cadre d'application de notre méthodologie. L'objectif va être de proposer une méthode hybride tirant profit de chacune des méthodes: la robustesse des algorithmes génétiques dans les cas de problèmes d'optimisation non-convexes et la rapidité de notre méthode semi-déterministe.

Tous les algorithmes introduits dans ce chapitre sont validés et comparés entre eux, et avec des algorithmes génétiques classiques, sur des fonctions tests.

Les résultats présentés dans cette section ont donné lieu à la rédaction des papiers suivants:

- Ivorra, B. and Mohammadi, B. and Santiago, J. and Hertzog, D. "*Semi-Deterministic Global Optimization Algorithms*", Proceeding of 7ème Colloque national en calcul des structures. Lavoisier. Vol.: 2 p.p.: 253-259 (2005) [1].

- Ivorra, B. and Mohammadi, B. and Redont, P. "*Low-Complexity Global Optimization by Solution of BVP*". Journal of Global Optimization. Springer-Science. In

revision process, 2005[2].

Ensuite nous appliquons certaines méthodes décrites précédemment à trois problèmes d'optimisation concrets:

• **Synthèse de filtres optiques à base de fibres à réseaux de Bragg:** Cette application industrielle fut la première étudiée durant cette thèse. Elle a commencé par une collaboration avec le professeur Laurent Dumas du laboratoire Jacques-Louis Lions de l'Université Paris VI et Olivier Durand ingénieur aux bureaux Recherche et Développement (R&D) du groupe Alcatel.

Par la suite ces travaux ont donné lieu à l'acceptation par le Centre National de la Recherche Scientifique (CNRS) d'un projet en co-tutelle avec le professeur Yves Moreau du Centre d'Électronique et de Micro-optoélectronique de Montpellier (CEM2) de l'Université Montpellier 2. Une budget de 10.000 euros nous a été attribué pour le développement d'un nouveau type de dispositif de guide optique.

Nous considérons dans ce chapitre un problème inverse: Étant donné le spectre d'une fibre optique, dite à réseaux de Bragg, nous voulons reconstruire sa variation d'indice de réfraction interne. Ce type de fibre est utilisée depuis de nombreuses années en télécommunication pour le transport de données et possède de nombreux avantages: un faible coût de fabrication, une simplicité de réalisation et un vaste domaine d'application.

Dans ce chapitre après avoir présenté une modélisation des fibres optiques considérées, nous introduisons puis résolvons trois problèmes d'optimisa-tion liés à la conception de filtres optiques à l'aide des algorithmes introduits dans le chapitre 2. Nous comparons les résultats, lorsque cela est possible, avec ceux obtenus par une méthode industrielle classique.

Les travaux présentés dans cette partie ont été publiés dans:

- Ivorra, B. and Mohammadi, B. and Dumas, L. and Durand, O. and Redont, P. *"Semi-Deterministic vs. Genetic Algorithms for Global Optimization of Multichannel Optical Filters"*. International Journal of Computational Science for Engineering, Accepted. To be published, 2006. [3]

- Ivorra, B. and Mohammadi, B. and Dumas, L. and Durand, O. and Moreau, Y. *"Semi-Deterministic Recursive Optimization Methods for Multichannel Optical Filters"*. Numerical Mathematics and Advanced applications, Accepted, 2006. [4]

• **Optimisation de forme d'un mélangeur microfluidique utilisé pour le repliement de protéines:** Ce problème nous a été proposé par Juan G. Santiago et David E. Hertzog du département d'ingénierie mécanique de l'Université de Stanford aux États-Unis et a donné lieu à une collaboration entre nos deux laboratoires.

Nous nous intéressons ici à l'optimisation de forme d'un mélangeur microfluidique. Ce dispositif est utilisé, par exemple, pour l'étude et l'identification de séquences d'ADN ou bien encore pour le repliement de protéines. Nous nous concentrons dans cette partie sur cette dernière application.

Le repliement des protéines est un des principaux axes de recherche actuels de la biologie moléculaire. Les protéines sont des polymères de petite taille qui sont capables

d'exister sous deux formes, l'une dépliée et biologiquement inactive, l'autre repliée et biologiquement active. Elles possèdent la propriété de se replier en une forme géométrique unique, soit dans les conditions physiologiques, soit artificiellement, lors d'un processus faisant intervenir certains solvants.

L'objectif de ce travail est de déterminer la forme d'un mélangeur microfluidique, dit de Knight [5], pour laquelle un mélange, composé d'un dénaturant et d'une protéine, se dilue le plus rapidement possible dans un solvant, ceci afin de faciliter le repliement de la protéine. Dans un premier temps nous présentons l'état de l'art et les travaux déjà réalisés sur la conception de ce type de mélangeur. Ensuite, après avoir introduit une modélisation de notre problème, nous analysons les résultats numériques obtenus à l'aide des trois algorithmes sélectionnés à la fin du le chapitre 2. Enfin nous comparons et validons les résultats trouvés numériquement avec ceux obtenus de manière expérimentale.

Ces travaux ont été publiés dans les deux papiers suivants:

- Ivorra, B. and Mohammadi, B. and Santiago, J.G.and Hertzog, D.E. "*Semi-deterministic and genetic algorithms for global optimization of microfluidic protein folding devices*". International Journal of Numerical Method in Engineering. Vol.: 66(2) p.p.: 319-333 (2006) [3].

- Hertzog, D.E. and Ivorra, B. and Mohammadi, B. and Bakajin, O. and Santiago, J.G. "*Optimization of a Fast Microfluidic Mixer for Studying Protein Folding Kinetics*". Analytical chemistry. Accepted, published in (2006). [6]

• **Optimisation sous contraintes d'un portefeuille de crédits:** Le secteur financier européen est l'un de ceux qui connaît actuellement une des plus forte croissance et qui reste l'un des plus compétitif. La conception d'outils permettant de gérer des investissements toujours plus gros et toujours plus diversifiés devient une nécessité afin de soutenir ce développement.

Dans ce chapitre, nous nous intéressons en particulier à l'amélioration d'un portefeuille de crédits à l'aide d'un algorithme d'optimisation numérique, du point de vue mesure de risque et rendement. Plus précisément, après avoir défini le modèle, les contraintes et les mesures de performances associées à notre portefeuille de départ, nous formulons divers problèmes d'optimisation que nous résolvons à l'aide d'une méthode d'optimisation hybride présentée dans le chapitre 2. Ensuite, nous analysons et comparons l'impact de notre optimisation sur la structure du portefeuille par rapport aux intuitions financières des '*portfolio managers*'.

Afin de nous mettre dans un cadre réaliste et de mieux cerner les problématiques intéressantes, ces travaux ont été réalisés lors d'une collaboration avec l'équipe Asset-Management de la Banque de Financement et d'Investissement du groupe BNP-Paribas, sous la forme d'un stage de valorisation de thèse de 5 mois. Ils ont donné lieu à la rédaction du papier suivant:

- Ivorra, B. and Mohammadi, B. and Guillaume, Q. and Rim, T. and Delcourt, S. "*An Hybrid Optimization Method For Risk Measure Reduction Of A Credit Portfolio Under Constraints*". Applied Mathematical Finance. submitted, 2006. [7].

Dans le chapitre 6 nous fournissons une description détaillée du logiciel d'optimisation que nous avons développé au cours de cette thèse. Il est basé sur les méthodes introduites dans le chapitre 2.

Historiquement, la première version du logiciel, contenant un unique algorithme d'optimisation, a été écrite en Fortran. Très vite, lors de la réalisation de cette thèse nous nous sommes rendus compte de certaines limitations de ce logiciel autant au niveau technique qu'au niveau portabilité. Nous voulions développer notre optimiseur afin qu'il soit le plus facilement applicable au monde industriel. Le choix de Matlab et de son langage M-Script s'est donc imposé de lui-même de part sa facilité de programmation, de couplage (FEMLAB, FLUENT, Codes C++...) et sa place prépondérante dans les entreprises. Nous nous attachons donc ici à ne présenter que la dernière version Matlab du logiciel.

Enfin, après avoir tiré les conclusions de ces travaux, la partie annexe présente d'autres résultats obtenus en parallèle durant ces trois années de recherche:

• Un premier résultat porte sur l'application d'une des méthodes semi-déterministes présentée dans le chapitre 2 et sa comparaison avec un algorithme génétique sur un problème de contrôle par point de l'équation visqueuse de Burgers. La discussion tourne aussi autour de la nécessité d'utiliser ou non une méthode globale en fonction du coefficient de pondération posé devant le terme de contrôle de la fonction à minimiser. Ce problème a été soulevé par le professeur Angel Manuel Ramos del Olmo à l'occasion du stage de Diplôme dÉtudes Approfondies (DEA) effectué à l'Université Complutense de Madrid. Il a donné lieu récemment à la rédaction de l'article suivant:

- Ivorra, B. and Ramos del Olmo, A.M. and Mohammadi B. "*A Semi-Deterministic Global Optimization Mehtod. Application to a Control Problem of the Burger Equation, comparing with Other Methods.*". In preparation.

• Un ensemble de lemmes et de propriétés sur les systèmes dynamiques d'ordre deux a été trouvé lors de la recherche des preuves des théorèmes du chapitre 2.

Chapter 1

Rappels

Ce premier chapitre a pour but de rappeler quelques définitions et notions de base dont nous nous servons dans les chapitres suivants.

La première partie fera un bref rappel sur les équations différentielles. les résultats énoncés ici sont utilisés lors de la démonstration du premier théorème du chapitre 2 et des propriétés présentées en annexe B.

La deuxième partie portera sur les espaces probabilisés dont nous nous servons dans le chapitre 5.

Enfin la troisième partie présente de manière classique la structure des algorithmes génétiques, dont une formulation originale est donnée dans le chapitre 2.

1.1 Equations différentielles

1.1.1 Equations autonomes

Definition 1 *On considère les équations différentielles dans lesquelles la variable de temps, t, n'apparaît pas explicitement:*

$$\begin{cases} \dot{x} = f(x) \\ x \in D \end{cases} \tag{1.1}$$

avec $D \subset \mathbb{R}^n$ et $f : D \to \mathbb{R}^n$.

Un telle équation est appelée **équation autonome**.

Definition 2 *Dans l'équation (1.1), D est appelé* **plan de phase**.

Theorem 1 (Propriété de translation) *On suppose que $\phi(t)$ est solution de l'équation (1.1) dans le domaine $D \in \mathbb{R}^n$, alors $\phi(t + t_0)$, avec $t_0 \in \mathbb{R}$, est aussi une solution de (1.1).*

1.1.2 Points critiques

Definition 3 *On considère l'équation (1.1). Le point $x = a$ avec $f(a) = 0$ est appelé* **point critique** *(ou* **point singulier** *) de (1.1).*

Une partie de l'étude des équations différentielles consiste à observer le comportement des solutions en fonction des points critiques: leurs propriétés d'attraction et de stabilité.

Attraction des points critiques

Definition 4 *Soit $x(t)$ solution de l'équation (1.1).*

Un point critique $x = a$ de l'équation (1.1) est appelé:

• **attracteur positif** *ou* **attractif** *s'il existe $t_0 \in \mathbb{R}$ et un voisinage $\Omega_a \subset \mathbb{R}^n$ de $x = a$ tels que:*

$$x(t_0) \in \Omega_a \ \Rightarrow \lim_{t \to +\infty} x(t) = a$$

• **attracteur négatif** *ou* **répulsif** *s'il existe $t_0 \in \mathbb{R}$ et un voisinage $\Omega_a \subset \mathbb{R}^n$ de $x = a$ tels que:*

$$x(t_0) \in \Omega_a \ \Rightarrow \lim_{t \to -\infty} x(t) = a$$

Stabilité des points critiques

Dans l'étude de la stabilité d'un point critique, la notion de fonction de Lyapounov joue un rôle important.

Definition 5 *Une* **fonction de Lyapounov** *pour l'équation (1.1) est une fonction: $L : D \to \mathbb{R}^n$ telle que pour toute solution $\gamma(t)$ de (1.1), on ait:*

$$\frac{d}{dt}(L(\gamma(t))) \leq 0$$

Definition 6 *On considère l'équation:*

$$\dot{x}(t) = f(x,t) \qquad t \in \mathbb{R}, x \in \mathbb{R}^n \tag{1.2}$$

avec $f(t,x)$ continue par rapport à la variable t, et uniformément-continue par rapport à la variable x. Les solutions de (1.2) partant à $t = t_0$ de $x = x_0 \in D$ sont notées $x(t; t_0, x_0)$.

Soit $x = a$ un point critique de (1.1).

· $x = a$ est dit ***stable au sens de Lyapounov*** *si:*

$\forall \epsilon \in \mathbb{R}$, $\exists \delta(\epsilon, t_0)$ tel que:

$$|x_0 - a| \leq \delta(\epsilon, t_0) \Rightarrow |x(t; t_0, x_0)| \leq \epsilon \quad \forall t \geq t_0$$

· $x = a$ est dit ***asymptotiquement stable au sens de Lyapounov*** *si:*

$\exists \delta(t_0)$ tel que:

$$|x_0 - a| \leq \delta(t_0) \Rightarrow \lim_{t \to +\infty} |x(t; t_0, x_0)| = 0$$

Theorem 2 *On considère l'équation:*

$$\dot{x}(t) = f(x,t) \qquad t \geq t_0, x \in D \subset R^n \tag{2.4}$$

avec: $f(t,a) = 0 \ \forall t \geq t_0$

S'il existe une fonction de Lyapounov, L, définie au voisinage de $x = a$ et définie positive pour $t \geq t_0$ et $L(a) = 0$, alors la solution $x = 0$ est stable au sens de Lyapounov.

1.1.3 Ensembles $\omega - limite$ et $\alpha - limite$

Un autre étude classique consiste à chercher les points d'accumulation d'une trajectoire.

Definition 7 *1- Un point $p \in \mathbb{R}^n$ est appelé* **point limite positif** *de la trajectoire γ, correspondant à la solution x(t) de (1.1),si:*

$\exists (t_n)_n \to +\infty$ *telle que* $(\gamma(t_n))_n$ *ait pour limite p.*

2- Un point $p \in \mathbb{R}^n$ est appelé **point limite négatif** *de la trajectoire γ, correspondant à la solution x(t) de (1.1),si:*

$\exists (t_n)_n \to -\infty$ *telle que* $(\gamma(t_n))_n$ *ait pour limite p.*

Definition 8 *L'ensemble des points limites positifs d'une trajectoire γ est appelé $\omega - limite$* de γ.
L'ensemble des points limites négatifs d'une trajectoire γ est appelé $\alpha - limite$ de γ.

1.1.4 Lemme de Gronwall

Nous nous servons aussi par la suite du Lemme de Gronwall et de son corollaire:

Lemma 9 (Gronwall) *On suppose que pour: $t_0 \leq t \leq t_0 + a$, où $a \in R^+$, on ait:*

$$\phi(t) \leq \delta_1 \int_{t_0}^{t} \psi(s)\phi(s)ds + \delta_2$$

Avec:
· $\forall t \in [t_0, t_0 + a]$, $\phi(t)$ *et* $\psi(t)$ *sont des fonctions continues et positives.*
· δ_1 *et* δ_2 *constantes positives.*
Alors $\forall t \in [t_0, t_0 + a]$*:*

$$\phi(t) \leq \delta_3 \exp^{\delta_1 \int_{t_0}^{t} \psi(s)ds}$$

Corollary 10 (Gronwall) *On suppose que pour $t_0 \leq t \leq t_0 + a$, avec a constante positive, on ait l'inégalité suivante:*

$$\phi(t) \leq \delta_3(t - t_0) + \delta_1 \int_{t_0}^{t} \phi(s)ds + \delta_2$$

avec, $\forall t \in [t_0, t_0 + a]$*:*
· $\phi(t)$ *est une fonction continue et positive.*
· $\delta_1 > 0$ *et* δ_2, δ_3 *sont des constantes positives.*
Alors $\forall t \in [t_0, t_0 + a]$*:*

$$\phi(t) \leq (\frac{\delta_2}{\delta_1} + \delta_3) \exp^{\delta_1(t-t_0)} - \frac{\delta_1}{\delta_2}$$

1.2 Probabilité

1.2.1 Espaces probabilisés

Definition 11 *Un ensemble de parties Υ d'un ensemble Ω qui est stable par réunion, intersection et complémentarité s'appelle une* ***tribu*** *sur Ω.*

Definition 12 *Si Ω peut être muni d'une topologie, alors la tribu engendrée par la classe des ouverts de Ω est appellée* ***tribu borélienne****.*

Definition 13 *Soit Υ une tribu sur Ω, le couple (Ω, Υ), où Ω est un ensemble, est appelé* ***espace probabilisable****. $X \in \Upsilon$ est appelé événement.*

Definition 14 *Soit (Ω, Υ) un espace probabilisable. $\mathbb{P} : \Omega \to \mathbb{R}^+, A \to \mathbb{P}(A)$ est une* ***mesure de probabilité*** *si et seulement si:*

- $\mathbb{P}(\Omega) = 1$
- $(A_n)_{n\in\mathbb{N}}$ *est une famille d'événements 2 à 2 disjoints, alors:*

$$\mathbb{P}(\bigcup_{n\in\mathbb{N}} A_n) = \sum_{n\in\mathbb{N}} \mathbb{P}(A_n)$$

Definition 15 *Le triplet $(\Omega, \Upsilon, \mathbb{P})$ est appelé* ***espace probabilisé****.*

Definition 16 *Soit (Ω, Υ) et (Θ, Ψ) deux espaces probabilisés. On dit que:*

$$\begin{array}{rccc} X: & \Omega & \mapsto & \Theta \\ & \omega & \to & X(\omega) \end{array} \tag{1.3}$$

est une ***variable aléatoire*** *si et seulement si pour tout $\theta \subset \Theta$, $X^{-1}(\theta) \subset \Omega$ (X est dite mesurable).*

Definition 17 *Soit $X : (\Omega, \Upsilon, \mathbb{P}) \to (\mathbb{R}^n, B(\mathbb{R}^n))$ une variable aléatoire.*

- *On appelle* ***loi*** *de X la mesure, notée $\mathbb{P}_X$, définie pour tout borélien B de $\mathbb{R}^n$ par:*

$$\mathbb{P}_X(B) = \mathbb{P}(X \in B) = \int_B f(x)dx$$

- *f est appelée* ***fonction densité*** *de la loi $\mathbb{P}_X$*

Definition 18 *On dit que X est $\mathbb{P}$-**intégrable** si:*
$\int_\Omega X d\mathbb{P} \in \mathbb{R}^+$ si X est a valeur positive ou
$\int_\Omega X^+ d\mathbb{P} < +\infty$ et $\int_\Omega X^- d\mathbb{P} < +\infty$ si X est quelconque.

Definition 19 *On considère la tribu borélienne $(\mathbb{R}^n, B(\mathbb{R}^n))$. Soit $X : (\Omega, \Upsilon, \mathbb{P}) \to (\mathbb{R}^n, B(\mathbb{R}^n))$ une variable aléatoire. On note $L^p(\Omega, \Upsilon, \mathbb{P})$ l'ensemble des variables $\mathbb{P}$-intégrables telles que $\int_\Omega \|X\|^p d\mathbb{P} < +\infty$. On dit que X est* ***puissance*** *p $\mathbb{P}$-**intégrable**.*

Definition 20 *Soit* $X : (\Omega, \Upsilon, \mathbb{P}) \to (\mathbb{R}^n, B(\mathbb{R}^n))$ *une variable aléatoire 1* $\mathbb{P}$*-intégrable.*

- *On appelle* ***espérance mathématique*** *de* X *la moyenne de* $X : \mathrm{E}(X) = \int_\Omega x f(x) dx$.
- *On appelle* ***variance*** *de X:* $Var(X) = \mathrm{E}(X^2) - \mathrm{E}(X)^2$.
- *On appelle* ***écart type*** *de* X*:* $\sigma(X) = \sqrt{Var(X)}$.

Definition 21 *Soit deux variables aléatoires* $X : (\Omega, \Upsilon, \mathbb{P}) \to (\mathbb{R}^n, B(\mathbb{R}^n))$ *et* $Y : (\Omega, \Upsilon, \mathbb{P}) \to (\mathbb{R}^n, B(\mathbb{R}^n))$.

- *On appelle* ***covariance*** *de* X *et de* Y*, et on note* $cov(X, Y)$*, la quantité:*

$$cov(X, Y) = \int_\Omega \int_\Omega (x - E(X))(x - E(Y)) f(x, y) dx dy$$

  où $f(x, y)$ *est la fonction de densité jointe de* X *et de* Y.

- *On appelle* ***coefficient de correlation*** *entre* X *et* Y*, et on note* $cor(X, Y)$*, la quantité:*

$$cor(X, Y) = \frac{cov(X, Y)}{\sigma(X)\sigma(Y)}$$

Definition 22 *Une variable aléatoire* $X : (\Omega, \Upsilon, \mathbb{P}) \to (\mathbb{R}^n, B(\mathbb{R}^n))$ *suit une loi normale (ou gaussienne) généralisée, d'espérance* $\mu \in \mathbb{R}$ *et d'écart type* $\sigma \in \mathbb{R}^+$*, si la fonction de densité de la loi* $\mathbb{P}_X$ *est définie par* $\aleph(\mu, \sigma)(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

1.3 Algorithmes génétiques

1.3.1 Définitions

On considère un problème d'optimisation (P) qui s'écrit sous la forme:

$$\min_{x \in \Omega} J(x) \tag{1.4}$$

et U un ensemble de solutions potentielles de (P):

Definition 23

- *le problème* (P) *est appelé* ***environnement***.
- *U est appelé* ***population***.
- *Un élément* $u \in U$ *est appelé* ***individu***.

Definition 24 *Un* ***chromosome*** *est un tableau de valeur dont chaque case s'appelle* ***loci*** *et les valeurs prises s'appellent* ***allèles***. *Si ces allèles sont inclus dans une partie de* $\mathbb{R}$ *on parle de* ***codage réel***, *si les allèles sont choisis dans l'ensemble {{0},{1}} on parle de* ***codage binaire***.

Remarque: Le codage binaire est bien approprié au calcul informatique mais l'un des principal inconvénient est la taille des chromosomes importante entraînant des temps de calcul importants. C'est pour cela que l'on préfère souvent le codage réel [8].

Definition 25 *Chaque individu de la population est représenté par son* ***génotype****, c'est à dire l'ensemble des chromosomes qui le caractérise. Lorsque le génotype est représenté par un seul chromosome, on parlera de structure* ***Haploïde****.*

On se place ici uniquement dans le cas d'une structure Haploïde.

1.3.2 Structure d'un algorithme génétique

Une fois que l'on a choisi un codage des allèles réel ou binaire, le schéma que suit l'algorithme génétique est le suivant:

1- Dans le cas d'un codage binaire: Dans un premier temps on calcule la longueur minimale n du chromosome. Pour cela on considère:
. d: la précision (nombre de décimale souhaitée) de l'allèle.
. $[x^i_{\min}, x^i_{\max}]$ l'intervalle de variation pour le i^{eme} loci du chromosome caractérisant u.
Alors n est le plus petit entier tel que: $\sum_i [x^i_{\min}, x^i_{\max}] .10^d \leq 2^n$.

exemple: Si x$\in [-2.58, 2.58]$, d=2 alors n=8 et Le chromosome 00000000 représente la valeur -2.58 et 11111111 la valeur 2,58.

2- On génère aléatoirement une population de solutions potentielles U^1. Celle-ci correspond à la première **génération**.
Il est à noter que la taille de la population, N_{pop}, est un paramètre important:

Proposition 26 *Lorsque la taille de la population est supérieure à une valeur critique, l'interaction assure une convergence en temps (fini ou non).*

3- A partir de U^1 on va créer récursivement de nouvelles populations $U^1, ..., U^i, ..., U^{N_{gen}}$, où i correspond au numéro de la génération et N_{gen} correspond au nombre maximal de générations que l'on se fixe (voir plus loin les conditions d'arrêt de l'algorithme).

Pour i de 1 à N_{Gen}

- **Sélection:**

  On va évaluer les différents individus de la population U^i grâce à une fonction appelée **fitness function** ou **fonction de mérite**. La fonction de mérite est en fait la fonction coût de notre problème d'optimisation (P), c'est à dire J.

  On notera par la suite J_i la valeur de la fonction mérite de l'individu i.

  On procède alors à une sélection de N individus, pour cela on dispose de plusieurs méthodes:

- Roulette: La probabilité P_i d'un individu d'être sélectionné est proportionnelle à sa fonction mérite: $\frac{J_i}{\sum_i J_i}$.
- Tournois T: On sélectionne une sous-population (i.e. un sous ensemble de U) de taille T au hasard et on prend le meilleur représentant. On réitère le processus N fois.
- Élitisme: Cette méthode consiste à sélectionner à chaque fois le meilleur individu de la génération précédente puis à compléter par une autre méthode de sélection les N-1 individus restants.

- **Croisement:**

  Une fois nos N individus sélectionnés on passe à l'étape de croisement. Cette technique permet l'échange de données au sein d'une population.

  On détermine aléatoirement (avec une probabilité $p_c \in [0,1]$) s'il doit participer au croisement ou non. À chaque individu j on associe un nombre $\mathbf{R}_j \in [0,1]$:

  - Si $\mathrm{R}_j \leq p_c$ on lui applique l'opérateur de croisement.
  - Si non, on parle de **reproduction**, on lui affecte directement l'opérateur mutation.

  Pour effectuer le croisement on choisit deux individus (appelés **parents**) parmi ceux susceptibles de se reproduire et on applique une des méthodes suivantes:

  Dans le cas d'un codage binaire:

  - Croisement simple à un point: On coupe le chromosome des parents en un point, choisi aléatoirement, et on échange les parties obtenues.

    exemple:

$$\begin{array}{llcll} A_1 & 001|01000 & \rightarrow & \mathbf{010}|01000 & B_1 \\ A_2 & \mathbf{010}|\mathbf{10011} & & 001|\mathbf{10011} & B_2 \end{array} \tag{1.5}$$

    où A_1 et A_2 sont les parents et B_1 et B_2 les **enfants**.
  - Croisement à p-points: On prend p coupures, choisies de manière aléatoire, dans le chromosomes et on intervertit les différentes parties.

    exemple: p=3

$$\begin{array}{llcll} A_1 & 001|010|00 & \rightarrow & \mathbf{010}|010|\mathbf{11} & B_1 \\ A_2 & \mathbf{010}|\mathbf{100}|\mathbf{11} & & 001|\mathbf{100}|00 & B_2 \end{array} \tag{1.6}$$

    où A_1 et A_2 sont les parents et B_1 et B_2 les enfants.

– Croisement uniforme: On détermine aléatoirement avec une probabilité q si le loci doit être échangé. À chaque loci k on associe aléatoirement la valeur q_k.
-Si $q_k > q$: $B_{1k} = A_{2k}$ et $B_{2k} = A_{1k}$
-Si $q_k \leq q$: $B_{1k} = A_{1k}$ et $B_{2k} = A_{2k}$
exemple: q=$\frac{1}{2}$

$$\begin{array}{llcll} A_1 & 00101000 & \rightarrow & \mathbf{0}01\mathbf{10}00\mathbf{1} & B_1 \\ A_2 & \mathbf{01010011} & & 0\mathbf{10}10\mathbf{01}0 & B_2 \end{array} \tag{1.7}$$

où A_1 et A_2 sont les parents et B_1 et B_2 les enfants.

Dans le cas d'un codage réel:

Les mêmes techniques de croisement utilisées dans le cas binaire peuvent être adaptées au cas réel.

exemple: Croisement à deux points de coupure:

$$\begin{array}{llcll} A_1 & x1|x2|x3 & \rightarrow & \mathbf{y1}|x2|\mathbf{y3} & B_1 \\ A_2 & \mathbf{y1}|\mathbf{y2}|\mathbf{y3} & & x1|\mathbf{y2}|x3 & B_2 \end{array} \tag{1.8}$$

où A_1 et A_2 sont les parents et B_1 et B_2 les enfants.

Il est à noter qu'ici on ne coupe pas les variables comme ce serait le cas en binaire. On n'a donc pas l'apparition de nouvelles valeurs. L'espace de recherche est donc plus restreint. L'étape de mutation décrite par la suite joue alors un rôle beaucoup plus important que dans le cas binaire.

Croisement barycentrique: Pour palier à ce problème, dans le cas où l'espace d'étude est convexe, on peut considérer une méthode de croisement barycentrique:

$$B_1 = \lambda_1 A_1 + (1 - \lambda_1) A_2$$
$$B_2 = \lambda_2 A_1 + (1 - \lambda_2) A_2$$

Où λ_1 et λ_2 sont aléatoirement choisis entre $[0, 1]$.

- **Mutation:**

Nous avons obtenu une nouvelle population intermédiaire composée de N individus. On va appliquer à chacun un opérateur de mutation.

On se fixe une probabilité de mutation $p_m \in [0, 1]$.

Dans le cas d'un codage binaire:

A chaque allèle k on associe aléatoirement une valeur $r_k \in [0, 1]$. Si $r_k < p_m$ on mute l'allèle en son opposé (0 deviens 1 et 1 deviens 0), sinon on ne le touche pas.

Dans le cas d'un codage réel:

Si l'allèle y_i doit subir une mutation, alors y_i^*, la nouvelle valeur, sera choisie aléatoirement entre $[y^i_{\min}, y^i_{\max}]$, un intervalle borné contenant y_i. De plus on crée un opérateur mutation de telle sorte que y_i^* tende vers y_i lorsque le temps $t \to +\infty$. Ceci afin d'affiner les résultats. Quand la mutation est grande on parle de **phase d'exploration**, à l'inverse lorsqu'elle est petite on parle de **phase d'exploitation**. Il y a pour cela un algorithme bien adapté à ces exigences:

.Mutation non uniforme:

$$\mathrm{y}_i^* = \begin{cases} y_i + \triangle(t, y^i_{\max} - y_i) si \Gamma = 0 \\ y_i + \triangle(t, y_i - y^i_{\min}) si \Gamma = 1 \end{cases}$$

où t correspond au nombre de génération écoulées, Γ est un booléen aléatoire, $\triangle(t, y) = y.r.(1 - \frac{t}{T})^b$ avec r un nombre aléatoire entre [0,1], T le nombre maximal de génération autorisées, et b un paramètre de **raffinement**.

- **Tests d'arrêt:**

  Nous venons donc de générer à l'aide des trois processus (**Sélection**, **Croisement**, **Mutation**) et de la population U^i une nouvelle population U^{i+1}. À cette étape nous devons décider si continuer ou arrêter l'algorithme.

  La principale difficulté lors de la résolution d'un problème par algorithme génétique est le choix du test d'arrêt. Nous pouvons nous trouver dans deux cas:

  – Nous connaissons le minimum de la fonctionnelle, dans ce cas on peut alors implémenter le test de la forme:

  $$\textbf{Si } \frac{J(x^n) - J_{sol}}{J(x^0)} < \varepsilon \textbf{ alors fin du programme}$$

  où x^n est le meilleur élément de la génération n et J_{sol} le minimum de la fonction J. Le rôle de $J(x^0)$ est de relativiser la condition d'arrêt en fonction de la valeur initiale de la fonction.

  – Nous ne connaissons pas le minimum de la fonctionnelle, c'est le cas le plus fréquent, alors:

  On fixe un nombre de génération maximum, le choix de celui-ci est assez subjectif. Il dépend avant tout de la complexité du problème.

  Si le meilleur élément d'une génération reste inchangé ou évolue de manière négligeable ($\leq \varepsilon$ fixé) depuis η générations alors on arrête l'algorithme. Là aussi η doit être choisis en fonction du problème. Le risque principal étant que le meilleur élément stagne d'une génération à une autre et puis d'un coups change, dans ce cas si ε est trop petit l'algorithme s'arrête trop tôt sans détecter le nouveau meilleur élément, on parle alors de **convergence prématurée**.

  En fait c'est surtout l'expérience sur le problème d'optimisation étudié qui va nous permettre de choisir de manière intuitive le nombre de génération maximum et η.

Une fois les tests d'arrêt choisi, si l'algorithme en vérifie un on **Sort du Pour**

Fin du Pour

4- L'algorithme retourne le meilleur élément de la dernière génération. C'est à dire celui avec la plus faible valeur de fonction mérite.

Les principaux défauts des algorithmes génétiques sont leur relative lenteur (le nombre d'évaluation de la fonction de mérite qu'ils requièrent est relativement important), leur manque de précision et dans certains cas une convergence prématurée. Pour remédier à ces problèmes, nous présentons ici différentes techniques [9].

1.3.3 Améliorations de l'algorithme

Convergence prématurée: Ce problème intervient lorsque nous avons apparition d'un **super-individu** (c'est à dire un individu dont la fonction mérite est nettement supérieure à la moyenne). Dans ce cas il se reproduira en de nombreux exemplaires faisant ainsi converger l'algorithme. Il y a deux causes à ce problème:

- On a trouvé un individu proche de la solution exacte.
- On est tombé sur un minimum local, et dans ce cas, du fait que l'individu se soit reproduit un trop grand nombre de fois on n'arrive pas à en sortir.

La principale difficulté est que, ne connaissant pas la solution exacte, on ne sait jamais dans quel cas on se trouve. Pour remédier à ce genre de problème on peut utiliser des méthodes dites de **sharing** qui ont pour but de maintenir une certaine diversité parmi les individus. Elles peuvent intervenir à différents endroits:

- Sélection: Prévenir un **inceste**: Deux individus trop proches (au sens de la distance voir ci-dessous) ne peuvent se croiser.
- Création d'enfant: On augmente le taux de croisement p_c.
- Mutation: On peut créer un opérateur **Mutation dépendant de la distance (MDD)**. Le taux de mutation va dépendre d'une distance qui sépare les deux parents lors du croisement. Cette stratégie peut être couplée à une sélection élitiste pour éviter de perdre un individu intéressant.

Grâce à ces méthodes dans les cas d'apparition de super individus notre algorithme va rentrer dans une phase exploratoire (le taux de mutation va être élevé). Il ne nous reste plus qu'à définir une distance adéquate. Pour cela on introduit:

Definition 27 ***Distance de Hamming:*** *On considère deux individus i, j alors la distance $d(i,j)$ qui les sépare est donnée par:*

- *Cas Binaire:*

$$d(i,j) = \frac{1}{n}\sum_{k=1}^{n} |m_i^k - m_j^k|$$

où n est la taille du chromosome, m_i^k la valeur du k^e allèle de l'individu i.

- *Cas Réel:*

$$d(i,j) = \frac{1}{n}\sum_{k=1}^{n}\left|\frac{m_i^k - m_j^k}{m_{\max}^i - m_{\max}^j}\right|^{\frac{1}{q}}$$

où $m_{\max}^i$ correspond à la borne sup du k^e allèle de l'individu i.

Le paramètre q permet de régler plus finement la nature de la distance. Pour une valeur de q élevée nous obtenons une distance qui favorise des taux de mutation faibles dès que des individus commencent à être différents. Plus les individus sont proches plus leur distance est proche de 0. À l'inverse plus, ils sont proches, plus leur distance est proche de 1. On peut alors définir le taux de mutation d'un enfant issue des parents i et j par:

$$T_m(i,j) = (1 - d(i,j)).T_{\max}$$

où T_{max} a pour but de limiter la mutation.

Amélioration de la précision: Nous allons ici utiliser une technique d'**hybridation**. Le but va être de diviser notre algorithme en deux parties:

- Dans un premier temps on utilise un algorithme génétique pour trouver une approximation grossière de la solution. En fait le but va être de trouver le bassin d'attraction de la solution globale.
- Ensuite on applique une méthode 'classique': Descente, Armijo, L-BFSG,... pour raffiner le résultat.

En effet les algorithmes utilisant les méthodes de gradient sont beaucoup plus précis, mais ont comme principal défaut de converger lors de la rencontre avec le bassin d'attraction d'un minimum local. L'algorithme génétique a ici pour fonction de contourner ce problème et de trouver un point dans le bassin d'attraction du minimum global.

Accélération de convergence: Pour remédier à la lenteur des algorithmes génétiques, il existe de nombreuses méthodes permettant de gagner du temps de calcul.

Une technique consiste à construire un algorithme génétique **parallèle**:

Le but va être de diviser la population en sous-groupes afin de faire travailler plusieurs machines sur chacun de ceux-ci. Il y a deux principaux modèles:

- **Voisinage** : Une seule population, on définit un environnement spatial, chaque individu ne peut se croiser qu'avec un voisin. Le calcul se fait alors de manière asynchrone (Il n' y a plus de contrôle global au niveau des générations). On affecte différents individus à chaque machine.
- **Insulaire** : On divise la population en sous-populations (isolation), elles cherchent toutes à optimiser la même fonction. De temps en temps il y a échange de certains individus (migration). On affecte à chaque machine une sous-population.

  On peut ensuite améliorer le fonctionnement de notre algorithme parallèle en développent une topologie dite **Hiérarchique**.

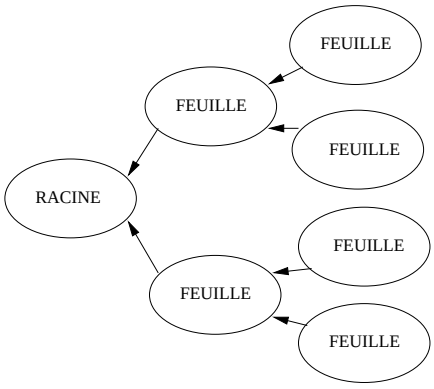


Figure 1.1: Topologie hiérarchique

Dans ce type de modèle des sous-populations sont là pour en aider d'autres (voir figure 1.1):

Il y a échange: Le **noeud père** reçoit les meilleurs éléments de ses **noeuds fils**.

Á chaque sous-population on peut affecter une stratégie différente (exploratoire tout en conservant un aspect exploitation).

-La sous population **racine** est dédiée à l'exploitation.

-Les noeuds **feuilles** sont dédiés à l'exploration.

-les sous-populations intermédiaires jouent les deux rôles.

1.3.4 Plus loin avec les méthodes stochastiques

Nous avons seulement mentionné ici les méthodes les plus fréquemment utilisées pour coder un algorithme génétique. Il existe d'autres moyens de sélection, croisement, mutation,... qui n'ont pas été décrites. Nous renvoyons le lecteur intéressé à la bibliographie suivante pour trouver de plus amples informations [9, 8, 10].

Part I

Global optimization method

Chapter 2

Global Optimization by the Solution of BVPs

2.1 Introduction

The global solution of minimization problems is of great practical importance and this is one of the reason why evolutionary algorithms received a tremendous interest in recent years [9, 11]. However, the main difficulties with these algorithms remain their computational time, their lack of precision and their slow convergence.

Many optimization algorithms can be viewed as discrete forms of Cauchy problems for a system of ordinary differential equations in the space of control parameters [12, 13]. We will see that if one introduces an extra information on the infimum, solving global optimization problems using these algorithms is equivalent to solving Boundary Value Problems (**BVP**) for the same equations. A motivating idea is therefore to apply algorithms solving BVPs to perform this global optimization. Furthermore, one notices that minimization algorithms, including non-deterministic ones, such as genetic algorithms, are discrete forms of first or second order ODEs (or systems of ODEs).

In this chapter we present a reformulation of global minimization problems in term of over-determined BVPs, discuss the existence of their solutions and present some algorithms solving those problems. Then we validate various algorithms included in former class on several benchmark minimization problems.

Section 2 presents our global optimization methods and mathematical background. In Section 3,the previous approaches are applied to some academic test cases.

2.2 Optimization method

We consider the following minimization problem:

$$\min_{x \in \Omega_{ad}} J(x) \tag{2.1}$$

where $J : \Omega_{ad} \to \mathbb{R}$ is called cost function, x is the optimization parameter and belongs to an admissible space $\Omega_{ad} \subset \mathbb{R}^N$, with $N \in \mathbb{N}$. We make the following assumptions [13]: $J \in C^2(\Omega_{ad}, \mathbb{R})$ and coercive. The infimum of J is denoted by J_m.

We propose to give an unified dynamical system formulation for stochastic (e.g. Genetic Algorithms (GA) [8, 9]) and deterministic algorithms. In fact, although GAs come from evolutionary considerations, it is possible to derive a set of stochastic coupled ODEs to represent the method. The aim is to overcome the main GA difficulty: the computational complexity.

To do so, we introduce a new class of global minimization methods based on the solution of BVP where GAs can be incorporated.

2.2.1 BVP formulation of optimization problems

Many minimization algorithms which perform the minimization of J can be seen as discretizations of continuous first or second order dynamical systems with associated initial conditions [12, 13].

A numerical global optimization of J with one of those algorithms, called here *core optimization method*, is possible if the following BVP has a solution:

$$\begin{cases} \text{First or second order initial value problem} \\ \min_{t\in[0,Z]}(|J(x(t)) - J_m|) < \epsilon \end{cases} \tag{2.2}$$

where $x(t)$ is the solution of the considered dynamical system found at time $t \in \mathbb{R}$, Z is a given finite time $Z \in \mathbb{R}$ and ϵ is the approximation precision. In practice, when J_m is unknown, we set J_m to a lower value (for example $J_m = 0$ for a non-negative function J) and look for the best solution for a given complexity and computational effort.

This BVP is over-determined as it includes more conditions than derivatives. This over-determination can be removed for instance by considering one of the initial conditions in the considered dynamical system as a new variable denoted by v. Then we could use what is known on BVP theory, for example a shooting method [12], in order to determine a suitable v solving (2.2).

2.2.2 General method for the resolution of BVP (2.2)

In order to solve previous BVP (2.2), we consider the following general method:

We introduce a function $h : \Omega_{ad} \to \mathbb{R}$ given by:

$$h(v) = \min_{t\in[0,Z]}(J(x(t,v)) - J_m) \tag{2.3}$$

where $x(t,v)$ is the solution of the dynamical system (2.2) starting from the initial condition v, defined previously, at time t. $Z \in \mathbb{R}$.

Solving BVP (2.2) can be performed by minimizing in Ω_{ad} function (2.3).

Depending on the selected optimization method, h is usually a discontinuous plateau function:

For example, if a Steepest Descent method is used as core optimization method, the associated dynamical system reaches, in theory, the same local minimum when it starts

from any points included in a same attraction basin. In other words, if Z is large enough, $h(v)$ is piecewise constant with values corresponding to the local minima of $J(x(Z,v))$. Furthermore, $h(v)$ is discontinuous where the functional reaches a local maximum, or has a plateau (see Figure (2.3)).

Thus, one way to minimize such a kind of function is for instance to use a GA. But this method is numerically expensive. We propose a cheaper multi-layers algorithm based on line search methods [12]:

We first consider the following algorithm $A_1(v_1, v_2)$:

- $(v_1, v_2) \in \Omega_{ad} \times \Omega_{ad}$ *given*

- *Find* $v \in argmin_{w \in \mathcal{O}(v_2)} h(w)$ *where* $\mathcal{O}(v_2) = \{v_1 + t(v_2 - v_1), t \in \mathbb{R}\} \cap \Omega_{ad}$ *using a line search method*

- *return* v

The line search minimization in A_1 is defined by the user. It might fails. For instance, a secant method [12] degenerates on plateaus and critical points. In this case, in order to have a multidimensional search, we add an external layer to the algorithm A_1 by minimizing h' defined by:

$$h'(v') = h(A_1(v', w')) \tag{2.4}$$

with w' chosen randomly in Ω_{ad}.

This leads to the following two-layers algorithm $A_2(v_1', v_2')$:

- $(v_1', v_2') \in \Omega_{ad} \times \Omega_{ad}$ *given*

- *Find* $v' \in argmin_{w \in \mathcal{O}(v_2')} h'(w)$ *where* $\mathcal{O}(v_2') = \{v_1' + t(v_2' - v_1'), t \in \mathbb{R}\} \cap \Omega_{ad}$ *using a line search method*

- *return* v'

The line search minimization in A_2 is defined by the user.

N.B Here we have only described a two-layers structure. But this construction can be pursued by building recursively $h^i(v_1^i) = h^{i-1}(A_{i-1}(v_1^i, v_2^i))$, *with* $h^1(v) = h(v)$ *and* $h^2(v) = h'(v)$ *where* $i = 1, 2, 3, ...$ *denotes the external layer.*

During this work, we call this general recursive algorithm: Semi-Deterministic Algorithm (**SDA**). For each class of method used as core optimization method, we will describe more precisely the SDA implementation.

2.2.3 1st order dynamical system based methods

We consider optimization methods that come from the discretization of the following dynamical system [13, 14, 12]:

$$\begin{cases} M(\zeta, x(\zeta))x_\zeta(\zeta) = -d(x(\zeta)) \\ x(\zeta = 0) = x_0 \end{cases} \tag{2.5}$$

where ζ is a fictitious time. $x_\zeta = \frac{dx}{d\zeta}$. M is an operator, $d : \Omega_{ad} \to \mathbb{R}^N$ is a function giving a suitable direction.

For example:

- If $d = \nabla J$, the gradient of J, and $M(\zeta, x(\zeta)) = Id$, the identity operator, we recover the classical steepest descent method.
- If $d = \nabla J$ and $M(\zeta, x(\zeta)) = \nabla^2 J(x(\zeta))$ the Hessian of J, we recover the Newton method.

In this case, BVP (2.2) can be rewritten as:

$$\begin{cases} M(\zeta, x(\zeta))x_\zeta = -d(x(\zeta)) \\ x(0) = x_0 \\ \min_{t\in[0,Z]}(|J(x(t)) - J_m|) < \epsilon \end{cases} \tag{2.6}$$

This BVP is over-determined by x_0. i.e, the choice of x_0 determines if BVP (2.6) admits or not a solution. For instance, in the case of a steepest descent method, BVP (2.6) generally has a solution if x_0 is in the attraction basin of the global minimum. More usually, it exits a x_0 solving BVP (2.6): $x_0 \in argmin_{x\in\Omega_{ad}} J(x)$.

In order to determine a such x_0, we consider the implementation of algorithms A_i with $i = 1, 2, 3, ...$ (here we limit the presentation to $i = 2$).

The first layer A_1 is applied with a secant method in order to perform line search. The output is denoted by$A_1(v_1, J, I, \epsilon)$, and the algorithm reads:

Input: v_1, J, I, ϵ
v_2 chosen randomly
For l
from 1 to J
$o_l = D(v_l, I, \epsilon)$
$o_{l+1} = D(v_{l+1}, I, \epsilon)$
If $J(o_l) = J(o_{l+1})$ **EndFor**
If $min\{J(o_m), m = 1, ..., l\} < \epsilon$ **EndFor**
$v_{l+2} = v_{l+1} - J(o_{l+1})\frac{v_{l+1}-v_l}{J(o_{l+1})-J(o_l)}$
EndFor
Output: $A_1(v_1, J, I, \epsilon)$:
$argmin\{J(o_m), p)m = 1, ..., l\}$

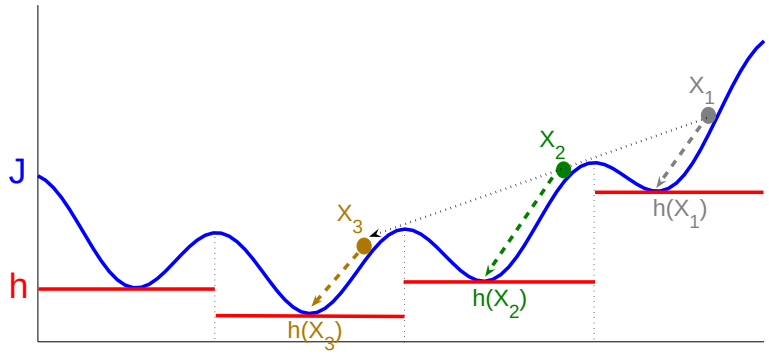


Figure 2.1: Graphical representation of one iteration of the algorithm A_1 considering a steepest descent method as core optimization method. $h(X_2)$ is lower than $h(X_1)$, thus X_3 is build starting from X_2 and considering the direction $\vec{X_1X_2}$.

where $v_1 \in \Omega$, $\epsilon \in \mathbb{R}^+$ and $(J, I) \in \mathbb{N}^2$ are respectively the initial condition, the stopping criterion and the iteration numbers. $D(v, I, \epsilon)$ is the solution returned by the core optimization algorithm starting from the initial point v after I iterations with a stopping criterion ϵ. A graphical representation of one iteration of A_1 is given by Figure 2.1.

The second layer A_2 is applied with a secant method in order to perform line search. The output is denoted by $A_2(w_1, K, J, I, \epsilon)$, and the algorithm reads:

Input: w_1, K, J, I, ϵ

w_2 chosen randomly

For l going from 1 to K

$p_l = A_1(w_l, J, I, \epsilon)$

$p_{l+1} = A_1(w_{l+1}, J, I, \epsilon)$

If $J(p_l) = J(p_{l+1})$ **EndFor**

If $min\{J(p_m), m = 1, ..., l\} < \epsilon$ **EndFor**

$w_{l+2} = w_{l+1} - J(p_{l+1})\frac{w_{l+1} - w_l}{J(p_{l+1}) - J(p_l)}$

EndFor

Output: $A_2(w_1, K, J, I, \epsilon)$:

$argmin\{J(p_m), m = 1, ..., l\}$

where $w_1 \in \Omega$, $\epsilon \in \mathbb{R}^+$ and $(K, J, I) \in \mathbb{N}^3$ are respectively the initial condition, the stopping criterion and the iteration numbers. A graphical representation of one iteration of A_2 is given by Figure 2.2.

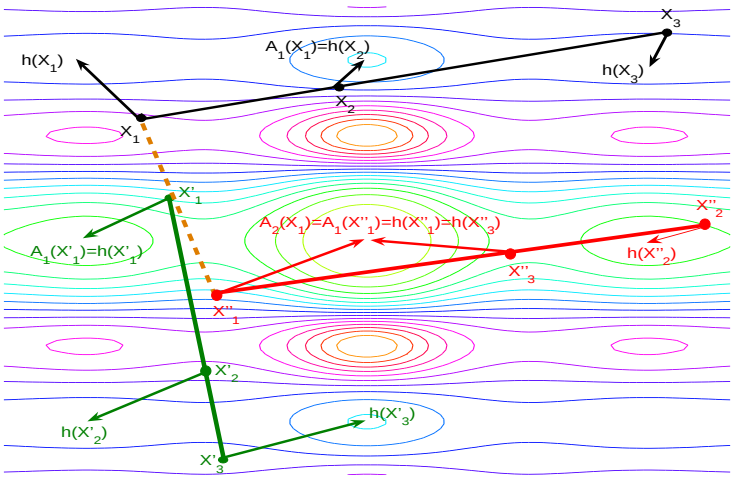


Figure 2.2: Graphical representation of one iteration of the algorithm A_2 considering a steepest descent method as core optimization method. Cost function $f : \mathbb{R}^2 \to \mathbb{R}$ is represented by its iso-contours.$A_1(X_1')$ is lower than $A_1(X_1)$, thus X_1'' is build starting from X_1' and considering the direction $\vec{X_1'X_1}$.

One dimensional geometrical interpretation

We give a simple geometrical interpretation of the approach above when the secant method is used to solve the external minimization problems (i.e. minimization of $h^i, i = 1, 2, ...$) and when a Steepest Descent method is used as the core minimization method (i.e. in the construction of h).

A one dimensional geometrical construction of the different functionals (J, h, h^2, h^3) is shown in Figure 2.3 with J non-convex, v_1 fixed and v_2^i for $i = 1, 2, 3$ take successively all the values of the discretized search space. h^2 and h^3 show growing attraction basins around the infimum. In that case, the attraction basin for h^3 is the full search space.

2.2.4 2nd order dynamical system based methods

In order to keep an exploratory character during the optimization process, allowing us to escape from attraction basins, we could use variants of previous methods after adding second order derivatives.

For instance we could consider methods coming from the discretization of the following '*heavy ball*' dynamical system [15]:

$$\begin{cases} \eta x_{\zeta\zeta}(\zeta) + M(\zeta, x(\zeta))x_\zeta(\zeta) = -d(x(\zeta)), \\ x(0) = x_0, \quad x_\zeta(0) = x_{\zeta,0} \end{cases} \tag{2.7}$$

with $\eta \in \mathbb{R}$.

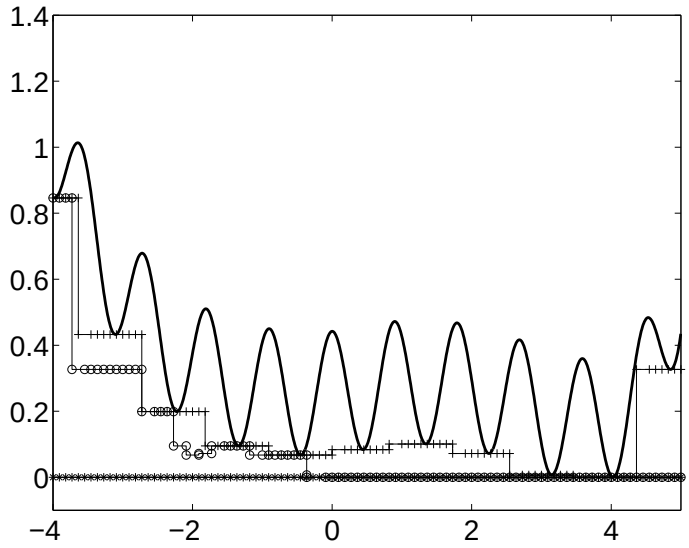


Figure 2.3: Geometrical interpretation of h (+), h^2 (o) and h^3 (*) considering a function $J(x)$ (continuous line) non-convex with several minima close to each other. h, h^2 and h^3 are built on a uniform sampling of the control space $x \in [-4, 5]$ at each level of the algorithm.

In this case, the associated BVP (2.2) is of the form:

$$\begin{cases} \eta x_{\zeta\zeta}(\zeta) + M(\zeta, x(\zeta))x_\zeta(\zeta) = -d(x(\zeta)), \\ x(0) = x_0, \quad x_\zeta(0) = x_{\zeta,0} \\ \min_{t\in[0,Z]}(|J(x(t)) - J_m|) < \epsilon \end{cases} \tag{2.8}$$

System (2.8) can be solved by considering x_0 (as previously) or $x_{\zeta,0}$ as a new variable. In the first case the existence of solution for BVP (2.8) is trivial. In the second case, considering particular hypothesis interesting in numerical analysis, when x_0 is fixed it can be proved that it exists a $x_{\zeta,0}$ such that BVP (2.8) admits numerical solutions.

Existence of solution for BVP (2.8)

During this work a first result was found in the one dimensional case (i.e. $N = 1$). This result was demonstrated in collaboration with Professor Jean-Paul Dufour (University of Montpellier II). It shows, in a particular context, the existence of initial conditions $x_{\zeta,0}$ solving BVP (2.8):

Theorem 3 *Let $J : \mathbb{R} \to \mathbb{R}$ such that:*
$\exists M \in \mathbb{R}$ such that $\forall \|x\| \geq M$, $J(x) = \frac{1}{2}\|x\|^2$.
Then $\forall (x_1, x_2) \in \mathbb{R}^2$, $\exists (x_0, \dot{x_0}) \in \mathbb{R}^2$ such that the solution of system:

$$\begin{cases} \ddot{x}(t) + \dot{x}(t) = -\nabla J(x(t)) \\ \dot{x}(0) = \dot{x_0} \\ x(0) = x_0 \end{cases} \tag{2.9}$$

pass through the points x_1 and x_2.

Proof:

• First, we study the following differential equation:

$$\ddot{x}(t) + \dot{x}(t) = x(t) \tag{2.10}$$

Let $y(t) = \dot{x}(t)$. Thus differential equation (2.10) becomes:

$$\begin{cases} \dot{x}(t) = y(t) \\ \dot{y}(t) = -y(t) - x(t) \end{cases} \tag{2.11}$$

We can reformulate system (2.11) as:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = A \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.12}$$

With A$= \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix}$

Eigenvalue of A are $\lambda_1 = \frac{-1+i\sqrt{3}}{2}$ and $\lambda_2 = \frac{-1-i\sqrt{3}}{2}$. Thus the eigen vectors of A are $(1, -\frac{1}{2} + i\frac{\sqrt{3}}{2})$ and $(1, -\frac{1}{2} - i\frac{\sqrt{3}}{2})$.

Denoting:

$$P = \begin{bmatrix} 1 & 1 \\ -\frac{1}{2} + i\frac{\sqrt{3}}{2} & -\frac{1}{2} - i\frac{\sqrt{3}}{2} \end{bmatrix}$$

We have:

$$P^{-1}AP = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

System (2.12) can be rewritten as:

$$P^{-1} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = P^{-1}APP^{-1} \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.13}$$

We denote:

$$\begin{bmatrix} X \\ Y \end{bmatrix} = P^{-1} \begin{bmatrix} x \\ y \end{bmatrix}$$

System (2.13) becomes:

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} \tag{2.14}$$

System (2.14) admits a complex solution :

$$X(t) = \exp^{\lambda_1 t} X_0$$

$$Y(t) = \exp^{\lambda_2 t} Y_0$$

Where X_0 and Y_0 depend of initial conditions.

Those solutions can be rewritten as:

$$X(t) = \exp^{-\frac{t}{2}}(A_0 \cos(\frac{\sqrt{3}}{2}t) + B_0 \sin(\frac{\sqrt{3}}{2}t))$$

$$Y(t) = \exp^{-\frac{t}{2}}(C_0 \cos(-\frac{\sqrt{3}}{2}t) + D_0 \sin(-\frac{\sqrt{3}}{2}t))$$

where A_0, B_0, C_0 and D_0 depend of the initial conditions.

Thus in order to find solution of system (2.10) we apply matrix P to $\begin{bmatrix} X \\ Y \end{bmatrix}$.

Solutions of (2.10) are of the form:

$$x(t) = \exp^{-\frac{t}{2}}(\delta_0 \sin(\frac{\sqrt{3}}{2}t + \varphi))$$

$$y(t) = \exp^{-\frac{t}{2}}(\rho_0 \sin(\frac{\sqrt{3}}{2}t + \psi))$$

These solutions describe spirals in the phase space(See Figure 2.4).

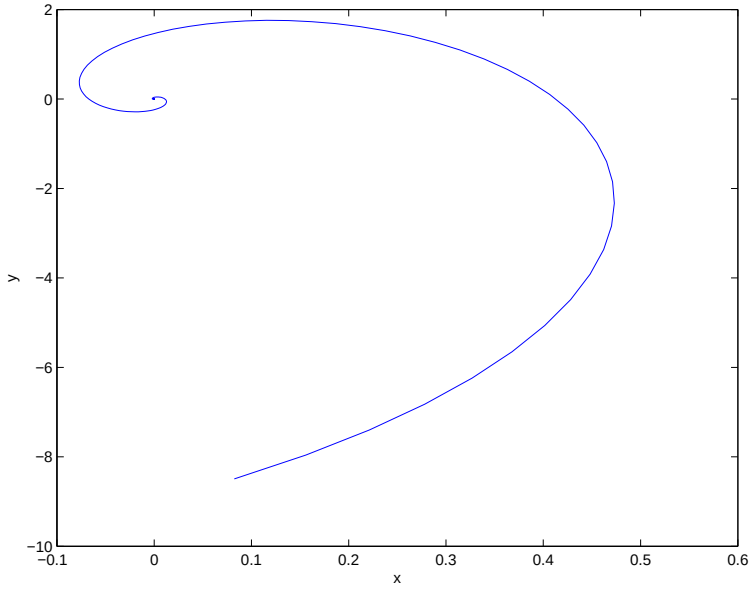


Figure 2.4: Solutions of (2.10) with: $\delta_0 = 1$, $\varphi = 0$, $\rho_0 = 9$ and $\psi = 99$ for $t \in [0.1; 1000]$

- Now we will study system (2.9) in dimension $n = 1$: We denote:

$$y(t) = \dot{x}(t)$$

System (2.9) becomes:

$$\begin{cases} \dot{x} = y \\ \dot{y} = -y - \frac{\partial J(x)}{\partial x} \\ \dot{x}(0) = \dot{x_0} \\ x(0) = x_0 \end{cases} \tag{2.15}$$

1-When $\|x\| \geq M$, (2.15) becomes:

$$\begin{cases} \dot{x} = y \\ \dot{y} = -y - x \\ \dot{x}(0) = \dot{x_0} \\ x(0) = x_0 \end{cases} \tag{2.16}$$

Solutions of (2.16) are the same than solutions of (2.10): $x(t) = \exp^{-\frac{t}{2}}(A_0 \sin(\frac{\sqrt{3}}{2}t + \varphi))$ and $y(t) = \exp^{-\frac{t}{2}}(B_0 \sin(\frac{\sqrt{3}}{2}t + \psi))$ with A_0 and B_0 depending of x_0 and $\dot{x}_0$.

2-When $\|x\| < M$:

We assume, at time t_0, that the trajectory x enters into the compact $\tau = \{\|x\| < M\}$. J is continuous, thus on τ: $\frac{\partial J(x)}{\partial x}$ is bounded, i.e.:

$$\exists c \in R^+ \; tel \; que \; \forall x \in \tau \; : \; |\frac{\partial J(x)}{\partial x}| \leq c$$

Thus, due to (2.16), We have:

$$-y - c \leq \dot{y} \leq -y + c \tag{2.17}$$

Then:

$$\int_{t_0}^{t} \dot{y} \leq c(t - t_0) - \int_{t_0}^{t} y(t)dt \quad \forall t \geq t_0$$

Thus:

$$|y(t)| \leq c(t - t_0) + |y(t_0)| + \int_{t_0}^{t} |y(t)|dt \quad \forall t \geq t_0$$

We apply the corollary of the Gronwall Lemma [16, 17] (See chapter 1):

$$|y(t)| \leq (|y(t_0)| + c) \exp^{t-t_0} - c \quad \forall t \geq t$$

Due to (2.17):

$$|\dot{y}(t)| \leq |y(t)| + c$$

Thus:

$$|\dot{y}(t)| \leq (|y(t_0)| + c) \exp^{t-t_0}$$

Denoting:

$$\mu(t) = (|y(t_0)| + c) \exp^{t-t_0}$$

In phase shift:

$$\begin{cases} \dot{x}(t_0) - \mu(t) \leq \dot{x}(t) \leq \dot{x}(t_0) + \mu(t) \\ x(t_0) + \dot{x}(t_0) - \mu(t) \leq x(t) \leq x(t_0) + \dot{x}(t_0) + \mu(t) \end{cases}$$

Trajectory $\gamma(t) = (x(t), y(t))$ is bounded by:

$.\gamma_{min} = (x(t_0) + \dot{x}(t_0) - \mu(t), \dot{x}(t_0) - \mu(t))$

$.\gamma_{max} = (x(t_0) + \dot{x}(t_0) + \mu(t), \dot{x}(t_0) + \mu(t))$

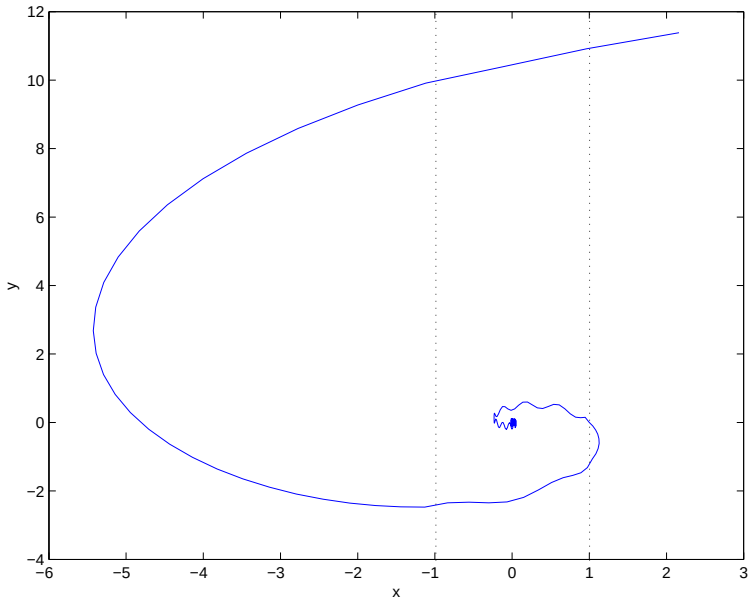


Figure 2.5: Graphical representation of $\gamma(t)$.

For $\dot{x}(t_0)$ enough high, denoted V_{min}, trajectory γ_{min} et γ_{max} leave the compact τ and $\nu = \{\dot{x} < V_{min}\}$, thus γ leave the compact τ.

Denoting $\nu = \{\dot{x} < V_{min}\}$. Outside the compact $\Gamma = \tau \bigcup \nu$, x is a spiral. A graphical representation of the trajectory is given in Figure 2.5. When time goes to $-\infty$, x cross all the phase shift verticals, thus:

$\forall (x_1, x_2) \in \mathbb{R}^2, \exists (x_0, \dot{x}_0) \in \mathbb{R}^2$ such that, solution of system (2.9) pass through x_1 and x_2.

□

Another result was also demonstrated for the multidimensional case, this work was performed in collaboration with Patrick Redont (University of Montpellier II). This theorem proves, considering particular hypothesis, the existence of an initial condition $x_{\zeta,0}$ solving numerically BVP (2.8):

Theorem 4 *Let $J : \mathbb{R}^n \to \mathbb{R}$ be a C^2-function such that $\min_{\mathbb{R}^n} J$ exists and is reached at $x_m \in \mathbb{R}^n$. Then for every $(x_0, \delta) \in \mathbb{R}^n \times \mathbb{R}^+$, exists $(\sigma, t) \in \mathbb{R}^n \times \mathbb{R}^+$ such that the solution of the following dynamical system:*

$$\begin{cases} \eta x_{\zeta\zeta}(\zeta) + x_\zeta(\zeta) = -\nabla J(x(\zeta)) \\ x(0) = x_0 \\ x_\zeta(0) = \sigma \end{cases} \tag{2.18}$$

with $\eta \in \mathbb{R}$, passes at time $\zeta = t$ into the ball $B_\delta(x_m)$.

Proof :

We assume $x_0 \neq x_m$. Let $\epsilon > 0$, we consider the dynamical system:

$$\begin{cases} \eta y_{\tau\tau}(\tau) + \epsilon y_\tau(\tau) = -\epsilon^2 \nabla J(x(\tau)) \\ y(0) = x_0 \\ y_\tau(0) = \varrho(x_m - x_0) \end{cases} \tag{2.19}$$

with ϱ in $\mathbb{R}^+\backslash\{0\}$.

- Assume that $\epsilon = 0$, we obtain the following system :

$$\begin{cases} \eta y_{\tau\tau,0}(\tau) = 0 \\ y_0(0) = x_0 \\ y_{\tau,0}(0) = \varrho(x_m - x_0) \end{cases} \tag{2.20}$$

  System (2.20) describes a straight line of origin x_0 and passing at time θ_ϱ by the point x_m, i.e. $y_0(\theta_\varrho) = x_m$.

- Assume that $\epsilon \neq 0$. System (2.19) could be rewritten as:

$$\begin{cases} \begin{pmatrix} y(\tau) \\ \eta y_\tau(\tau) \end{pmatrix}_\tau = \begin{pmatrix} y_\tau(\tau) \\ -\epsilon y_\tau(\tau) - \epsilon^2 \nabla J(y(\tau)) \end{pmatrix} \\ y(0) = x_0 \\ y_\tau(0) = \varrho(x_m - x_0) \end{cases} \tag{2.21}$$

  System (2.21) is of the form $y_\tau = f(\tau, y, \epsilon)$, with f satisfying the Cauchy-Lipschitz conditions. Applying the Cauchy-Lipschitz Theorem [17]:

  $|y_\epsilon(\theta_\varrho) - y_0(\theta_\varrho)| \rightarrow_{\epsilon\rightarrow 0} 0$ uniformly.

  Thus for every $\delta \in \mathbb{R}^+\backslash\{0\}$, there exists ϵ_δ such that for every $\epsilon < \epsilon_\delta$:
  $|y_\epsilon(\theta_\varrho) - x_m| < \delta$ (T.1)

Let $\delta \in \mathbb{R}^+\backslash\{0\}$. We consider the following variable changing $\zeta = \epsilon_\delta \tau$ and $x(\zeta) = y_{\epsilon_\delta}(\frac{\zeta}{\epsilon_\delta})$. System (2.19) becomes:

$$\begin{cases} \eta x_{\zeta\zeta}(\zeta) + x_\zeta(\zeta) = -\nabla J(x(\zeta)) \\ x(0) = x_0 \\ \dot{x}(0) = \frac{\varrho}{\epsilon_\delta}(x_m - x_0) \end{cases} \tag{2.22}$$

Let $\vartheta = \epsilon_\delta \theta_\varrho$. Under this assumption, $x(\vartheta) = y_{\epsilon_\delta}(\theta_\varrho)$. Thus, due to (T.1) : $|x(\vartheta) - x_m| < \delta$. We have found $\sigma = \frac{\varrho}{\epsilon_\delta}(x_m - x_0) \in \mathbb{R}^n$ and $t = \vartheta \in \mathbb{R}^+$ such that the solution of system (2.18) passes at time t into the ball $B_\delta(x_m)$.

□

Algorithmic implementation

In order to determine a suitable x_0 or $x_{\varsigma,0}$ solving BVP (2.8), we can consider, for instance, the same algorithms A_1 and A_2 introduced in section 2.2.3.

Difference between 1^{st} and 2^{nd} order system approaches

We consider two versions of the algorithm A_2, the first one uses a steepest descent algorithm as core optimization method and the second one uses an heavy ball based method. We apply both algorithms to the minimization of two functions, the first function f_1 is a perturbed version of a convex function and the second one, f_2 has no particular property. We use central differences for the discretization of the systems and a constant step size. Trajectory obtained during optimization processes are presented in Figure 2.6.

For f_1, minimization algorithms based on the discretization of either first or second order dynamical systems are efficient. While for f_2, only the algorithm based on the discretization of second order dynamical system detects the infimum, the other one searches the initial condition in a bad direction.

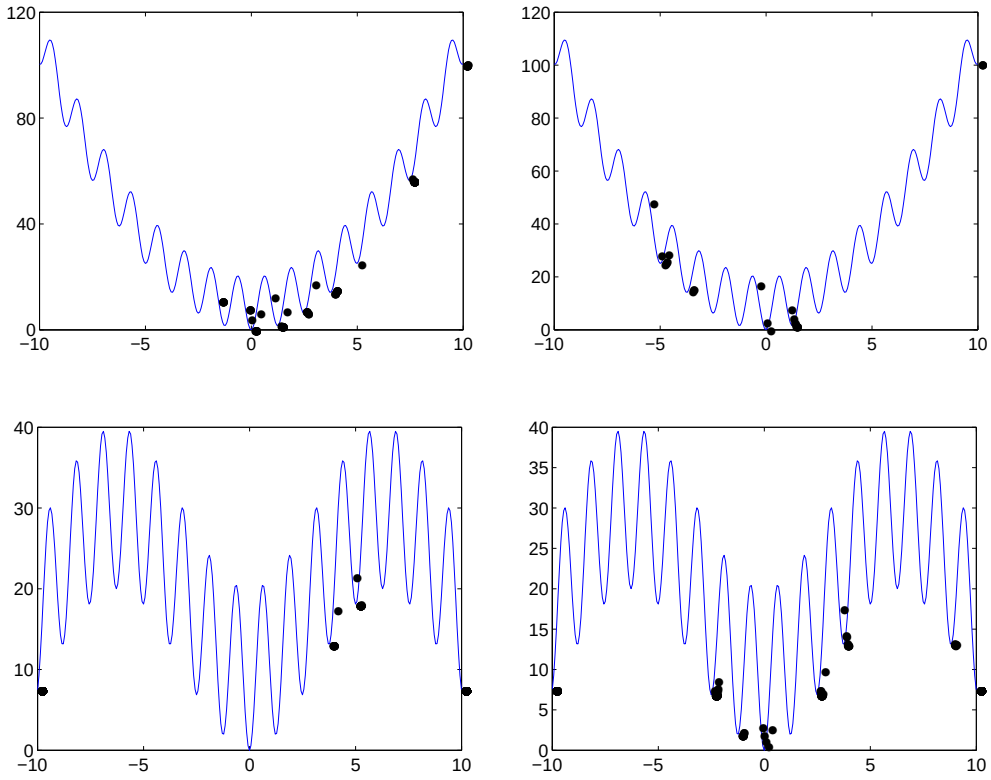


Figure 2.6: Behavior of the recursive algorithm for two non-convex functionals f_1 (a perturbed version of a convex function)(**Up**) and f_2 (a general non-convex function) (**Bottom**). The trajectories of (**left**) 1^{st} order and (**right**) 2^{nd} order dynamical systems based SDA are also reported (•).

2.2.5 Genetic algorithms and stochastic dynamical systems

Genetic algorithms approximate the global minimum of J, called in this case fitness function, through a stochastic process based on an analogy with the Darwinian evolution of species [9]: a first family, called 'population', $X^0 = \{x_l^0 \in \Omega, l = 1, ..., N_p\}$ of N_p possible solutions of the optimization problem, called 'individuals', is randomly generated in the search space Ω. Starting from this population, we build recursively N_{gen} new populations, called generations, $X^i = \{x_l^i \in \Omega, l = 1, ..., N_p\}$ with $i = 1, .., N_{\text{gen}}$ through three stochastic steps, called selection, crossover and mutation.

More precisely we present here a non usual matrix-form approach for GAs. This presentation was written in collaboration with Laurent Dumas (from Jacques-Louis Lions Laboratory - University of Paris VI):

We first rewrite X^n using the following (N_p, N)-real valued matrix form:

$$X^i = \begin{bmatrix} x_1^i(1) & \dots & x_1^i(N) \\ \vdots & \ddots & \vdots \\ x_{N_p}^i(1) & \dots & x_{N_p}^i(N) \end{bmatrix} \tag{2.23}$$

Selection: Each individual, x_l^i is ranked with respect to its fitness value $J(x_l^i)$ (i.e. lower is the fitness value better is the ranking). Then N_p individuals are randomly selected (individuals with better ranking have higher chances to be selected), with eventual repetitions, to become 'parents'.

Introducing $\mathcal{S}^i$ a binary (N_p, N_p)-matrix, generated according to previous ranking and selection processes, with $\mathcal{S}_{j,k}^i = 1$ if the kth individual of X^i is the 'parent' selection number j and $\mathcal{S}_{j,k}^i = 0$ if not. We define:

$$X^{i+1/3} = \mathcal{S}^i X^i. \tag{2.24}$$

Crossover: This process leads to a data exchange between two 'parents' and the apparition of two new individuals called 'children'. We determine, with a probability p_c, if two consecutive parents in $X^{i+1/3}$ should exchange data or if they are directly copied into the intermediate population $X^{i+2/3}$.

Introduce $\mathcal{C}^i$ a real-valued (N_p, N_p)-matrix where for each couple of consecutive lines $(2j-1, 2j)$ $(1 \leq j \leq \frac{N_p}{2}$ in case N_p is a even number or $1 \leq j \leq \frac{N_p-1}{2}$ in case N_p is a odd number), the coefficients of the $2j-1$th and $2j$th rows are given by:

$$\mathcal{C}_{2j-1,2j-1}^i = \lambda_1, \quad \mathcal{C}_{2j-1,2j}^i = 1 - \lambda_1, \quad \mathcal{C}_{2j,2j-1}^i = \lambda_2, \quad \mathcal{C}_{2j,2j}^i = 1 - \lambda_2$$

in this expression:

- $\lambda_1 = \lambda_2 = 1$ if parents are directly copied (with a probability $1 - p_c$).
- λ_1 and λ_2 are randomly chosen in $]0, 1[$ if a data exchange occurs between the two parents (with a probability p_c).

Other coefficients of $\mathcal{C}^i$ are set to 0. If N_p is a odd number, the N_pth parent is directly copied, i.e $\mathcal{C}^i_{N_p,N_p} = 1$.

This step can be summarized as:

$$X^{i+2/3} = \mathcal{C}^i X^{i+1/3} \tag{2.25}$$

Mutation: This process leads to new parameters values for some individuals of the population. More precisely, each children is modified (or mutated) with a fixed probability p_m.

Introduce for instance a random perturbation matrix $\mathcal{E}^i$ with a i-th line equals to:

- a random vector $\epsilon_i \in \mathbb{R}^N$, according to the admissible space Ω, if a mutation is applied to the ith children (with a probability p_m).
- 0 if no mutation is applied to the ith children (with a probability 1-p_m).

This step can then take the following form:

$$X^{i+1} = X^{i+2/3} + \mathcal{E}^i \tag{2.26}$$

Therefore, the new population can be written as:

$$X^{i+1} = \mathcal{C}^i \mathcal{S}^i X^i + \mathcal{E}^i \tag{2.27}$$

which is a particular discretization of a set of nonlinear 1st order ODEs [18]:

$$\dot{X}(t) = \Lambda_1(t, p_s, p_c)X(t) + \Lambda_2(t, p_m) - X(t) \tag{2.28}$$

where the construction of $\Lambda_l, l = 1, 2$ has been described above.

More precisely, from (2.28), GAs can been seen as solving the following BVP problem:

$$\begin{cases} \dot{X}(t) = \Lambda_1(t, p_s, p_c)X(t) + \Lambda_2(t, p_m) - X(t) \\ X(0) = X^0 \\ \min_{t\in[0,Z]}(|\widehat{J}(X(t)) - J_m|) < \epsilon \end{cases} \tag{2.29}$$

where:

-$\widehat{J}(X) = min(J(x_i)/x_i \in X)$

-$\{p_s, p_c, p_m\}$ is a set of fixed parameters.

- X of the form: $X = \{x_i/ \;\; i = 1, ..., N_p \;\; x_i \in \Omega_{ad}\}$

With these three basic evolution processes, it is generally observed that the best individual obtained is getting closer after each generation to the optimal solution of the problem [9, 8].

Engineers like GAs because these algorithms do not require sensitivity computation, perform global and multi-objective optimization and are easy to parallelize. Their drawbacks remain their weak mathematical background, their computational complexity, their slow convergence and their lack of accuracy. As a fine convergence is difficult to achieve with GAs based algorithms, it is recommended when it is possible, to complete the GA iterations by a descent method. This is especially useful when the functional is flat around the infimum [8].

Hybridization with SDA

In order to reduce the GA computational complexity keeping the efficiency of the method, we think that we need to couple it with our SDA technique:

- Semi-Deterministic nested minimization providing information on the choice of the population.
- GA for global optimization with this population.

SDA can be used to determine an initial population X^0 solving (2.29) (the existence of solution is trivial: we consider a point included in $argmin_{x\in\Omega_{ad}} J(x)$ into X^0). In this case, instead of using the algorithm A_1 introduced in section 2.2.3, we prefer to consider the following algorithm B_1, better adapted to the GA case:

- **Input:** X^0, N, ϵ

For i going from O to N
- $o^i = GA(X^i, P, \epsilon)$
- **If** $\min\{J(o^k), k = 1, ..., i\} < J_m + \epsilon$ **EndFor**
- We construct X^{i+1}: $\forall x^i \in X^i \; x^{i+1} = x^i - J(o_i)\frac{o^i - x^i}{J(o^i) - J(x^i)}$

EndFor
- **Output:** $B_1(X^0, N, P, \epsilon) = \text{argmin}\{J(o^k), k = 1, ..., i\}$

where $GA(X^i, P, \epsilon)$ corresponds to the output given by GA, P is the set of parameters of GA, X^0 is the GA's initial population, $N \in \mathbb{N}$ is the iteration number and ϵ is the stopping criterion.

We can build recursively algorithm B_2, B_3,... by replacing respectively $GA(X^i, P, \epsilon)$ by $B_1(X^i, N, P, \epsilon)$, $B_2(X^i, M, N, P, \epsilon)$,...

This algorithm is suitable for the GA case as before each GA execution it generates an initial population closer to the solution found at previous iteration. The zone near the current solution is better explored (this phenomenon is depicted by Figure 2.7). In cases when there is no evolution of the best element, the secant method coefficients allow to redistribute the initial population far from the current solution.

We call this approach **HGSA** (Hybrid Genetic/Semi-deterministic Algorithm).

2.2.6 Other hybridizations with SDA

In practice, any user-defined, black-box or commercial minimization package starting from an initial condition can be used to build the core optimization sequences in the SDA presented in section 2.2.2. In that sense, the algorithm permits the user to exploit his knowledge on his optimization problem and to improve it. In the same way, preconditioning can be introduced at any layer, and in particular at the lowest one.

2.3 Validation on benchmark functions

In this section we apply algorithms, included in the class of semi-deterministic methods exposed previously, to various benchmark optimization problems. Results are compared

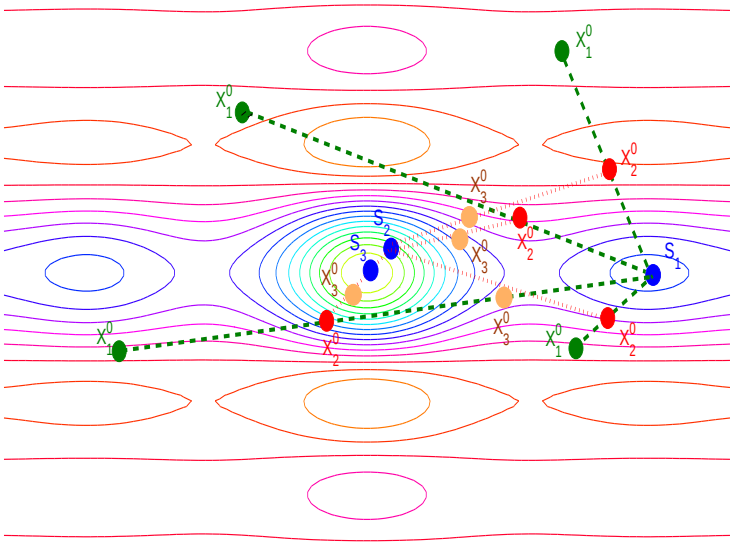


Figure 2.7: Graphical representation of two iterations of the algorithm B_1 in GA case. A first initial population X_1^0 is generated giving, with the considered GA, a solution S^1. A second initial population X_2^0 is generated, starting from X_1^0 and looking for S^1, the zone around S^1 is better studied. GA gives the solution S^2. A third population X_3^0 is generated, which explore the zone near S^2. The last execution of GA gives a solution S^3 closes to the infimum.

with those obtained with classical genetic methods.

2.3.1 Semi-deterministic algorithms (SDA)

We consider various versions of our SDA algorithms A_i (where i equals to 1,2 or 3) presented previously. The core optimization method is selected among the following list:

1- Steepest Descent method: It's defined by the following algorithm with an output called $D(x_0, I, \epsilon)$, where the inputs $x_0 \in \Omega$, $\epsilon \in \mathbb{R}^+$ are respectively the initial condition and the stopping criterion:

- **Input:** x_0, I, ϵ
- $x_1 = x_0$

For n going from 1 to I

- Determine $\rho_{opt} = argmin_\rho(J(x_n - \rho \nabla J(x_n)))$ using dichotomy
- $x_{n+1} = x_n - \rho_{opt} \nabla J(x_n)$
- **If** $J(x_{n+1}) < J_m + \epsilon$ **EndFor**

EndFor

• **Output:** $D(x_0, I, \epsilon) = x_{n+1}$

where the input I, the iteration number, is set to 10.

This algorithm denoted by **SD2A** (Steespest Descent based SDA) is used with different numbers of layer. We denotes:

- **SD2A-1L** when we consider the one-layer algorithm A_1.
- **SD2A-2L** when we consider the two-layers algorithm A_2.
- **SD2A-3L** when we consider the three-layers algorithm A_3.

2- Heavy Ball dynamical system based algorithm: We discretize system (2.8) [13]. The initial condition x_0 and iteration number of the core algorithm $I = 10$ are fixed, $x_{\zeta,0}$ becomes the parameter of SDA method A_2.

This algorithm will be denoted by **HBSDA** (Heavy Ball method based SDA)

3- A simplex search method: This is a direct search method that does not use numerical or analytic gradients. If n is the length of x, a simplex in n-dimensional space is characterized by the n+1 distinct vectors that are its vertices. In two-space, a simplex is a triangle; in three-space, it is a pyramid. At each step of the search, a new point in or near the current simplex is generated. The function value at the new point is compared with the function's values at the vertices of the simplex and, usually, one of the vertices is replaced by the new point, giving a new simplex. This step is repeated until the diameter of the simplex is less than the specified tolerance. This method described in [19].

This algorithm will be denoted by **SXSDA** (SimpleX based SDA)

4- Interior-reflective Newton method: It is a subspace trust region method. Each iteration involves the approximate solution of a large linear system using the method of preconditioned conjugate gradients. This algorithm is detailed in [20, 21].

This algorithm is denoted by **NSDA** (Newton based SDA)

5- BFGS Quasi-Newton method: It's mixed quadratic and cubic line search procedure. This quasi-Newton method uses the BFGS [20, 21] formula for updating the approximation of the Hessian matrix.

This algorithm is denoted by **QNSDA** (Quasi-Newton based SDA)

6- Sequential quadratic programming method: In this method, the function solves a quadratic programming (QP) subproblem at each iteration. An estimate of the Hessian of the Lagrangian is updated at each iteration using the BFGS formula. An estimate of the Hessian of the Lagrangian is updated at each iteration using the BFGS formula [22]. A line search is performed using a merit function similar to that proposed by [23, 24, 25]. The QP subproblem is solved using an active set strategy similar to that described in [26].

This algorithm is denoted by **SQPSDA** (SQP based SDA).

7- A genetic algorithm: In this case, we consider algorithm B_2 presented in section 2.2.5, instead of A_i, combined with the GA described in section 2.2.5. GA parameters are set to:

- The population size is set to $N_p = 10$.
- The selection is a roulette wheel type [9] proportional to the rank of the individual in the population.
- The crossover is barycentric in each coordinate with a probability of $p_c = 0.45$.
- The mutation process is non-uniform with a probability of $p_m = 0.35$.
- A one-elitism principle, that consists in keeping the current best individual in the next generation, has also been imposed.
- At the end of the algorithm, we perform a steepest descent method starting from the optimized result.

This algorithm is denoted by **HSGA**.

All those algorithms are applied with the following parameters:

- For HBSDA SXSDA, NSDA, QNSDA and SQPSDA the maximum functional evaluation number of the core algorithm is set to 200.
- For HSGA, SD2A-L2, HBSDA SXSDA, NSDA, QNSDA and SQPSDA the first layer iteration number is set to 10.
- For SD2A-L3 the first and second layers iteration numbers are set to 5.

2.3.2 Classical genetic algorithms

GA is applied with the same stochastic processes than the HSGA algorithm, with the two following set of parameters:

- First set denoted by **GA-S1**: $N_p = 180$, $p_c = 0.45$,$p_m = 0.15$.
- Second set denoted by **GA-S2**: $N_p = 50$, $p_c = 0.5$,$p_m = 0.3$.

N.B.: All parameters of sections 2.3.1 and 2.3.2 are fixed and used in all computations of this chapter.

2.3.3 Benchmark functions

Algorithms presented previously are validated on several global minimization problems available in [27]. The objective is to minimize benchmark functions with characteristics interesting in numerical analysis. Results with the description of each test are presented in Tables 2.8-2.13.

For those test cases, excepting cases when it's specified, the generation number N_{gen}, for GAs and the iteration number of external layer of semi-deterministic methods are unfixed.

Due to the stochastic aspect of all methods (provoked by the choice of initial conditions and search directions) results presented here are in fact a mean of several optimization processes.

N.B.: For all problems presented here, all considered core optimization methods fail to find a reasonable solution when they are applied alone (i.e. without hybridization with our SDA).

Considered benchmark functions to minimize are:

Large-isocontour function (LIF):

$$J(x) = \sum_{j=1}^{n} x_j^{2j}, x \in [-10, 10]^n \tag{2.30}$$

with $n = 10, 100$ and 1000. It's minimum $J_m = 0$ is reached at the origin. A two-dimensional representation of this function is presented by Figure 2.8.

This function is interesting as it's convex. However, in high dimensional cases, many optimization methods fail to minimize it correctly due to the flattening of the iso-contours. As we can see on Table 2.1 combining those algorithms with our technique allows to perform a satisfactory minimization. This benchmark test also point the difficulty of HBSDA to find accurate results, this is due to the perturbation created by the 2^{nd} order term.

HSGA and classical GAs are compared on Table 2.2. As we can see for the same computational complexity HSGA provides an accurate result than other GAs.

Generalized Rastringin function (GRF):

$$J(x) = \sum_{j=1}^{n} (x_j^2 - \cos(18x_j)), x \in [-5, 5]^n \tag{2.31}$$

with $n = 10, 100$ and 1000. It's minimum $J_m = 0$ is reached at the origin. A two-dimensional representation of this function is presented by Figure 2.9.

This function is a perturbed version of a convex function with a large number of local minima. We can see on Tables 2.3-2.4 the necessity to have a multi-layers search of the initial conditions. The three and two layer structures are equivalents. Furthermore, the

LIF	SD2A-1L	SD2A-2L	SD2A-3L	HBSDA
n=10 / $\epsilon = 10^{-6}$	600	1500	2000	8000
n=100 / $\epsilon = 10^{-7}$	800	1800	2200	Fail: 10^{-6}
n=1000 / $\epsilon = 10^{-8}$	900	2000	2500	Fail: 10^{-7}
	SXSDA	NSDA	QNSDA	SQPSDA
n=10 / $\epsilon = 10^{-6}$	6000	1000	250	400
n=100 / $\epsilon = 10^{-7}$	7000	7000	100	800
n=1000 / $\epsilon = 10^{-8}$	40000	12000	100	1000

Table 2.1: LIF results: Iteration number needed to obtain a reduction of ϵ of the initial value of the cost function. from (**top**) to (**bottom**): $n = 10$ with a reduction $\epsilon = 10^{-6}$,$n = 100$ with a reduction $\epsilon = 10^{-7}$ and $n = 1000$ with a reduction $\epsilon = 10^{-8}$. In cases of failure, the reduction order is specified.

LIF	HSGA	GA-S1	GA-S2
n=10	10^{-7}	10^{-5}	10^{-4}
n=100	10^{-5}	10^{-3}	10^{-3}
n=1000	10^{-5}	10^{-3}	10^{-3}

Table 2.2: LIF results: Functional reduction after 10000 algorithmic iterations. From (**top**) to (**bottom**) n=10,100 and 1000.

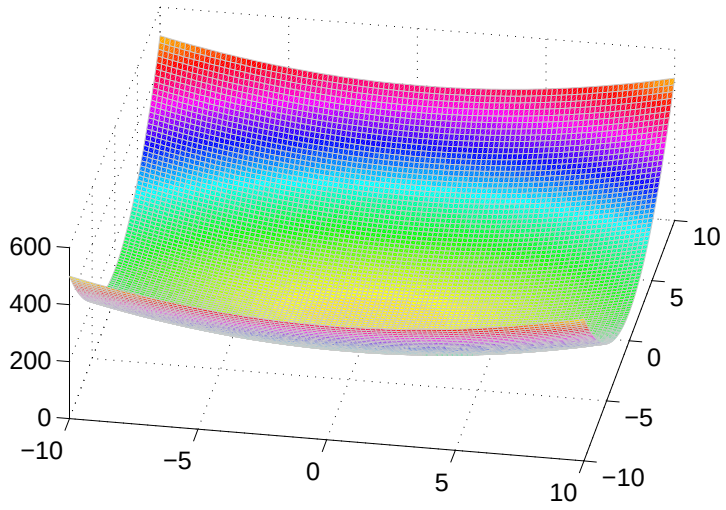


Figure 2.8: Large-isocontour function: two dimensional representation.

GRF	SD2A-1L	SD2A-2L	SD2A-3L	HBSDA
n=10 / $\epsilon = 10^{-6}$	Fail: $\epsilon = 10^{-3}$	1500	1500	Fail: 10^{-5}
n=100 / $\epsilon = 10^{-7}$	Fail: $\epsilon = 10^{-3}$	1500	1500	4000
n=1000 / $\epsilon = 10^{-8}$	Fail: $\epsilon = 10^{-2}$	1500	1500	10000
	SXSDA	NSDA	QNSDA	SQPSDA
n=10 / $\epsilon = 10^{-6}$	10000	10000	2500	3000
n=100 / $\epsilon = 10^{-7}$	20000	Fail: 10^{-5}	8000	5000
n=1000 / $\epsilon = 10^{-8}$	Fail: 10^{-3}	Fail: 10^{-3}	Fail: 10^{-3}	Fail: 10^{-1}

Table 2.3: GRF results: Iteration number needed to obtain a reduction ϵ of the initial value of the cost function. from (**top**) to (**bottom**): $n = 10$ with a reduction $\epsilon = 10^{-6}$,$n = 100$ with a reduction $\epsilon = 10^{-7}$ and $n = 1000$ with a reduction $\epsilon = 10^{-8}$. In cases of failure, the reduction order is specified.

GRF	HSGA	GA-S1	GA-S2
n=10	10^{-3}	10^{-2}	10^{-2}
n=100	10^{-1}	10^{-1}	10^{-1}
n=1000	10^{-1}	10^{-1}	10^{-1}

Table 2.4: GRF results: Functional reduction after 10000 algorithmic iterations. From (**top**) to (**bottom**) n=10,100 and 1000.

steepest descent method is better adapted to this case, due to its respect of the attraction basins.

All genetic based methods fail to find the global attraction basin due to the high number of minima.

Modified Rastringin Function (MRF):

$$J(x) = \sum_{j=1}^{n} (sin(x_j)^2 - \cos(18x_j)), x \in [-2, 2]^n \tag{2.32}$$

with $n = 10, 100$ and 1000. The minimum of J, $J_m = 0$, is reached at the origin. A two-dimensional representation of this function is presented by Figure 2.10.

In this case, we have modified previous function in order to create a general non-convex function with a large number of minima. As we can observe on Tables 2.5-2.3.3 using a second order dynamical system as core optimization method seems to be more adapted to this type of function.

As previously genetic methods doesn't give a satisfactory result.

Generalized Griewank function (GGF):

$$J(x) = \sum_{j=1}^{n} \frac{cos(x_j) - 100)}{\sqrt{j}} + \sum_{j=1}^{n} (x_j - 100)^2, x \in [-600, 600]^n \tag{2.33}$$

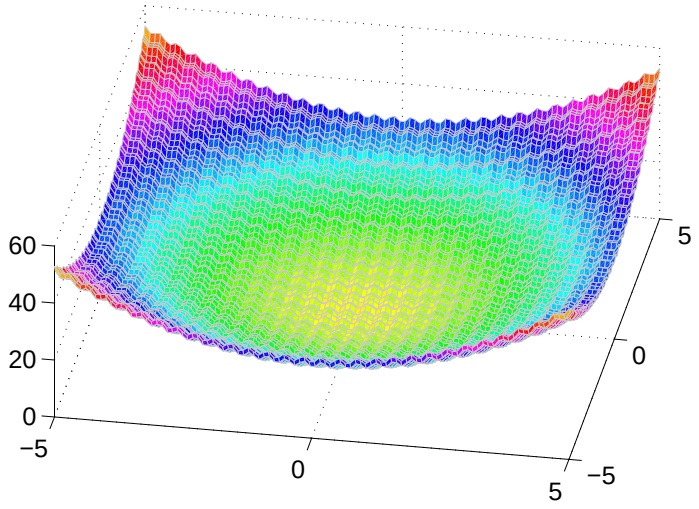


Figure 2.9: Generalized Rastringin function: two dimensional representation.

MRF	SD2A-1L	SD2A-2L	SD2A-3L	HBSDA
n=10 / $\epsilon = 10^{-6}$	Fail	1000	1500	1000
n=100 / $\epsilon = 10^{-7}$	Fail	1200	2500	1000
n=1000 / $\epsilon = 10^{-8}$	Fail	1500	5000	1000
	SXSDA	NSDA	QNSDA	SQPSDA
n=10 / $\epsilon = 10^{-6}$	Fail	2500	300	1000
n=100 / $\epsilon = 10^{-7}$	Fail	Fail	Fail	1000
n=1000 / $\epsilon = 10^{-8}$	Fail	Fail	Fail	1000

Table 2.5: MRF results: Iteration number needed to obtain a reduction ϵ of the initial value of the cost function. from (**top**) to (**bottom**): $n = 10$ with a reduction $\epsilon = 10^{-6}$,$n = 100$ with a reduction $\epsilon = 10^{-7}$ and $n = 1000$ with a reduction $\epsilon = 10^{-8}$.

MRF	HSGA	GA-S1	GA-S2
n=10	10^{-6}	10^{-4}	10^{-4}
n=100	10^{-2}	10^{-1}	10^{-1}
n=1000	10^{-2}	10^{-1}	10^{-1}

Table 2.6: MRF results: Functional reduction after 10000 algorithmic iterations. From (**top**) to (**bottom**) n=10,100 and 1000.

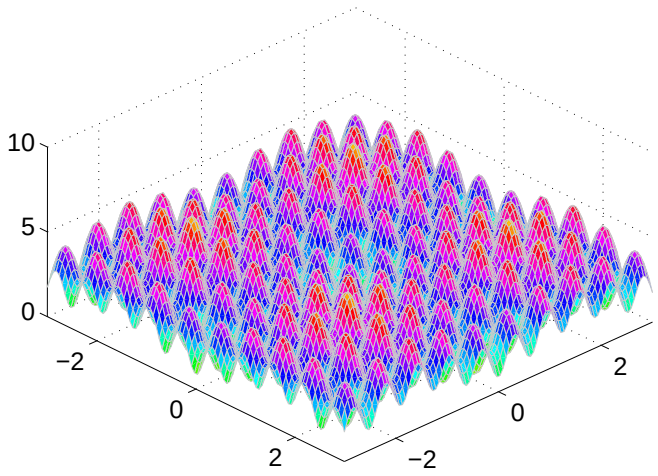


Figure 2.10: Modified Rastringin Function: two dimensional representation.

with $n = 10, 100$ and 1000. The minimum $J_m = 0$ is reached at $[100]^n$. A two-dimensional representation of this function is presented by Figure 2.11.

This function is a perturbed version of a convex function with a large number of minima. It's an interesting benchmark test function as the search space is larger and oscillation amplitudes are higher than in Rastringin case. As we can see on Table 2.7 minimization is difficult to perform. SD2A algorithms are the most efficients. Furthermore, HSGA also allows to find the good attraction basin.

Non-convex function (NCF):

$$J(x) = -exp(\frac{sin(x_1)}{x_1} * \frac{sin(x_2)}{x_2}) - e, x \in [-10, 10]^2 \tag{2.34}$$

The minimum $J_m = 0$ is reached at the origin. A two-dimensional representation of this function is presented by Figure 2.12.

This function contains several minima, each one having a large attraction basin. All algorithms, including GAs, succeed finding a good approximation of the optimum. However, as we can observe on table 2.8, HSGA is faster than other genetic methods.

Modified Rosenbrock function (MRF):

$$J(x) = 40 + 100 * (x_2 - x_1^2)^2 + (1 - x_1)^2 - 400 * e^{-10((x_1+1)^2+(x_2+1)^2)} \\ , x \in [-2, 2]^2 \tag{2.35}$$

The minimum $J_m = 0$ is reached at $[1, 1]$. A two-dimensional representation of this function is presented by Figure 2.13.

This function is compound by a large attraction basin of a local minimum and a small attraction basin of the global minimum. In this case only genetic based methods are efficient. Furthermore, HSGA still faster than other methods.

GGF	SD2A-1L	SD2A-2L	SD2A-3L	HBSDA
n=10	21	$1e^{-8}$	4	4
n=100	$1e^{-8}$	$1e^{-8}$	$1e^{-8}$	$1e^{-4}$
n=1000	$1e^{-8}$	$1e^{-8}$	$1e^{-8}$	$1e^{-4}$
	SXSDA	NSDA	QNSDA	SQPSDA
n=10	.21	18.19	0.18	0.14
n=100	0.1611	1.75	0.18	0.4
n=1000	1.0827	1.6	0.17	0.58
	HSGA	GA-S1	GA-S2	
n=10	10^{-8}	6	9	
n=100	10^{-8}	0.13	0.46	
n=1000	10^{-8}	0.7	0.8	

Table 2.7: GGF results: Distance between optimized solution obtained after 10000 algorithmic iterations and exact solution. Norm is given by $\|.\|_2/N$

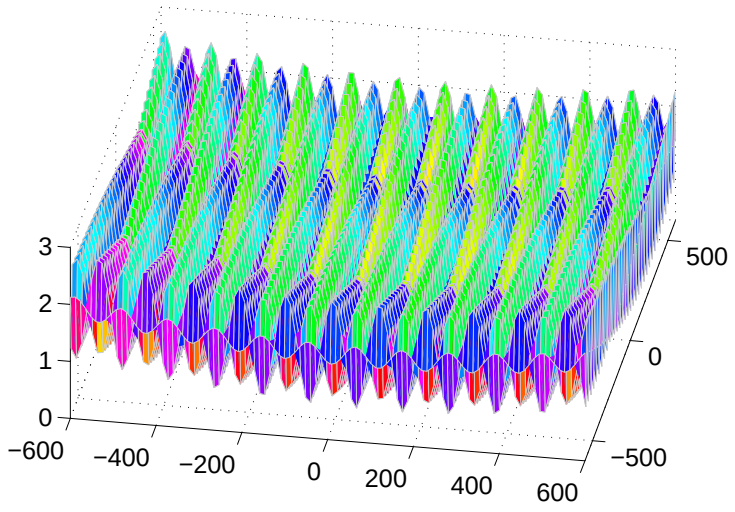


Figure 2.11: Generalized Griewank function: two dimensional representation.

SD2A-1L	SD2A-2L	SD2A-3L	HBSDA
2000	1500	1000	800
SXSDA	NSDA	QNSDA	SQPSDA
500	100	300	3000
HSGA	GA-S1	GA-S2	
600	1500	2700	

Table 2.8: NCF results: Number of functional iterations needed to find the optimum with a precision of $\epsilon = 10e^{-6}$.

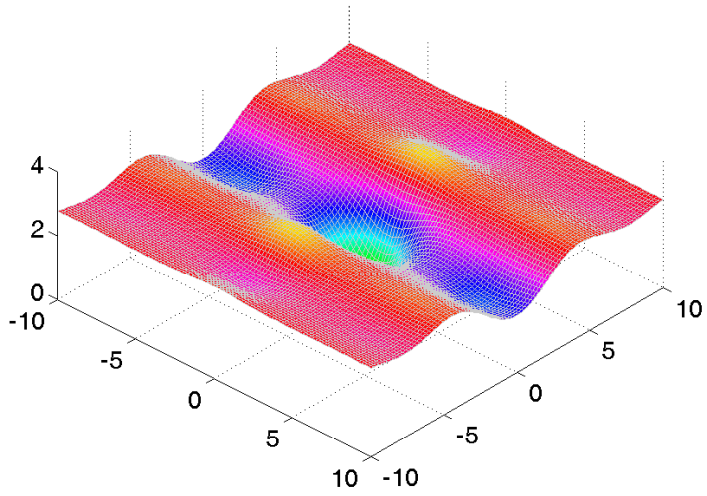


Figure 2.12: Non-Convex function: two dimensional representation.

SD2A-1L	SD2A-2L	SD2A-3L	HBSDA
Fail	Fail	Fail	Fail
SXSDA	NSDA	QNSDA	SQPSDA
Fail	Fail	Fail	Fail
HSGA	GA-S1	GA-S2	
1000	8000	5400	

Table 2.9: MRF results: Number of functional iterations needed to find the optimum with a precision of $\epsilon = 1e^{-6}$.

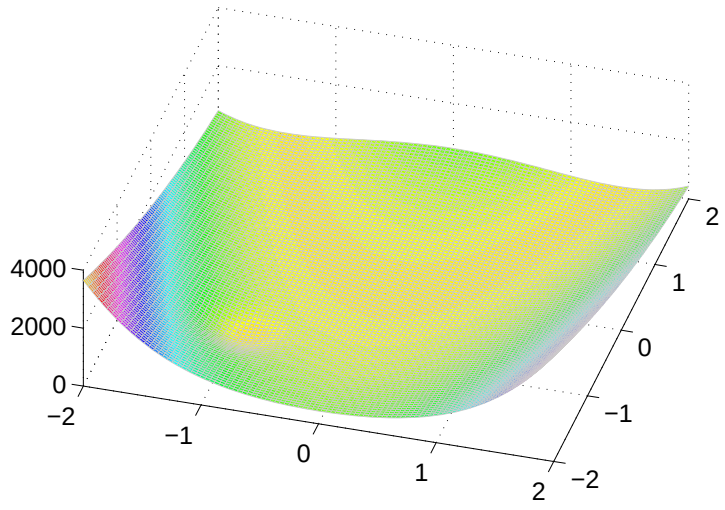


Figure 2.13: Modified Rosenbrock function: two dimensional representation.

2.3.4 Algorithms selection

In all future computations, according to previous benchmark results, we decided to use the following algorithms:

- SD2A-L2: it's the more stable algorithm. We use the following parameters $(M, N, I, \epsilon) = (5, 5, 10, 1.e^{-6})$. These values give a good compromise between computation complexity and result accuracy. In order to simplify notations, it will be denoted by **SD2A**.

  In order to reduce computational time during the optimization process, when it's possible we will use a Low-cost sensitivity technique [14]. Here we give a short presentation:

  Consider a general simulation loop, leading from shape parametrization x to the cost functional J:

  $$J(x) : x \rightarrow q(x) \rightarrow U(q(x)) \rightarrow J(x, q(x), U(q(x))) \tag{2.36}$$

  where q is the shape geometry and U is the state equation solution.

  The Jacobian of J is given by:

  $$\frac{dJ}{dx} = \frac{\partial J}{\partial x} + \frac{\partial J}{\partial q}\frac{\partial q}{\partial x} + \frac{\partial J}{\partial U}\frac{\partial U}{\partial q}\frac{\partial q}{\partial x} \tag{2.37}$$

  The last term $\frac{\partial J}{\partial U}\frac{\partial U}{\partial q}\frac{\partial q}{\partial x}$ is the more expensive to compute as it requires the linearization of the state equations. In order to reduce computational effort of sensitivity evaluations we could use reduced complexity models which provide an inexpensive approximation of the last term in (2.37).

A way to define low-complexity models is to use a different level of discretization for U with the same state equation. We can look for state sensitivity on coarse meshes while the state is evaluated on much finer discretizations:

$$\frac{dJ}{dx} = \frac{\partial J}{\partial x}(U_f, q_f) + \frac{\partial J}{\partial q}\frac{\partial q}{\partial x}(U_f, q_f) + \frac{\partial J}{\partial U}\frac{\partial U}{\partial q}\frac{\partial q}{\partial x}(I_f^c U_f, q_c)$$

where f and c subscripts denote fine and coarse meshes, I_f^c is an interpolation operator between the fine and coarse meshes. By fine mesh we mean a mesh enough fine for the solution to become independent from the mesh. This means that the linearization is performed on a coarse mesh, however around an accurate state variable computed on a fine mesh. In that case, obviously if the coarse mesh tends to the fine one, the approximate gradient tends to the gradient on the fine mesh. In addition, to the two levels of refinements used for state and sensitivity calculations, state evaluations for gradient calculation can be only made partially. Hence, only partial convergence of solver method in the solution of the state equations is required starting from $I_f^c U_f$. This corresponds to the fact that SD2A, like other descent methods, only needs a descent direction d such that $d.\nabla J > \varepsilon > 0$ [12].

- **HSGA** is also selected and applied in cases where sensitivity computations are not possible with the following parameters: $N_p = 10$,$N_{gen} = 10$, $p_c = 0.55$, $p_m = 0.45$. At the end of the algorithm, in order to improve the result accuracy, when it's possible, we perform a steepest descent method starting from the optimized result.

- In order to compare SD2A and HSGA with a classical **GA** method, we also use the previous GA algorithm with the following set of parameters: $N_p = 180$,$N_{gen} = 30$, $p_c = 0.45$, $p_m = 0.15$. At the end of the algorithm, a steepest descent method is performed.

For all algorithms, to reduce computational complexity, points already computed are stocked in memory in order to avoid recomputations. Indeed:

- For GA and HSGA each individual can be present and repeated in various generations.

- For SD2A, if two consecutive initial conditions give the same minimum, dynamical system trajectory is projected on the maximum or minimum boundary border. This can occur various time during the optimization process.

2.3.5 Conclusions

A new class of Semi-Deterministic methods has been introduced. This approach allows us to improve both deterministic and non-deterministic optimization algorithms. Various algorithms included in former class have been validated on various benchmark functions. Obtained results over-perform those given by a classical genetic algorithm in term of computational complexity and precision.

These algorithms will be applied in next chapters to various industrial optimization problems:

- Multichannel Optical Filters Design.
- Shape optimization of a fast-microfluidic protein folding device.
- Portfolio optimization under constraints.

Part II

Industrial applications

Chapter 3

Multichannel Optical Filters Design

3.1 Introduction

The use of new optical fiber devices in the telecommunication sector has known an important development in the last few years. Among them, Fiber Bragg Gratings (FBG) based devices represent an attractive and cheap alternative for applications such as multichannel filtering, multichannel optical add/drop multiplexing, multichannel dispersion compensation and multiwavelength laser sources.

Focusing on this last domain, the FBG functionality is to ensure the choice of desired wavelengths by forming a step-tunable laser [28, 29]. It may be incorporated either in all fiber structures such as erbium-doped fiber ring lasers or hybridized with an external Fabry-Perot cavity [30]. One way to perform their realization is to use Sampled FBGs (SFBG). Such filters are made of a spatial periodic distribution of identical sampling gratings. There is an important demand for optimization methods for the design of sampling patterns giving a desired reflectivity spectrum [31]. In addition, the method needs to perform global optimization as it's observed that considered functionals have multiple minima [10].

In this chapter we will study three optimization problems linked to the design of FBG filters:

- Pass-Band Filter Devices.
- Multichannel Filter Devices.
- Code Division Multiple Access (CDMA) Devices.

In order to solve those problems, we apply and compare SD2A, HSGA and GA methods presented in chapter 2. We show that superior results can be found with SD2A method from both industrial realizability and computational complexity point of view.

Section 3.2 presents the Fiber Bragg Gratings devices and their mathematical modelling. Section 3.3 describes our three optimization problems and presents the results.

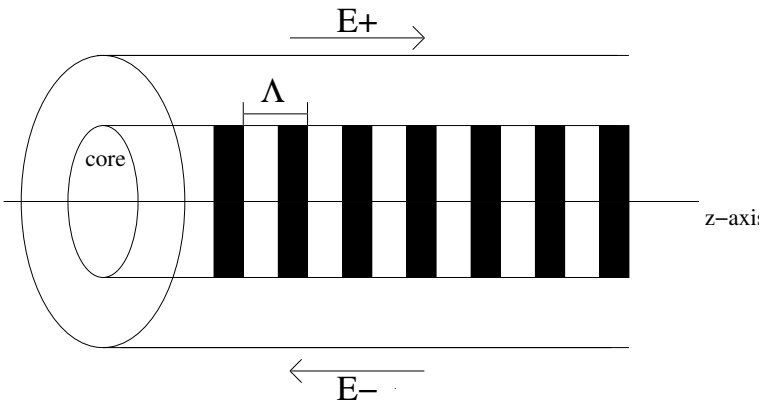


Figure 3.1: Fiber Bragg grating diagram: Dark stripes correspond to core zone where the refractive index is modulated. The fiber reflects a certain wavelength band, E-, and allows other one, E+, to pass. Λ represents the grating period.

3.2 Fiber Bragg Gratings (FBG)

Fiber Bragg Gratings are based on the perturbation of the effective refractive index of an optical guide in order to reflect a predetermined wavelength band and to let other bands to pass (see Figure 3.1). To write a FBG in an optical fiber, we expose it to an UV Laser radiation. This radiation modify the refractive index of the optical guide core in a periodic or an aperiodic way [32].

3.2.1 Mathematical modelling

In order to derive a mathematical model of a FBG, we assume there exists only two counterpropagating guided modes in the FBG of respective amplitudes $A(z,\lambda)$ and $B(z,\lambda)$ for any wavelength λ in the transmission band. We denote by n_{eff} the unperturbed refractive effective index and by $\beta = \frac{2\pi n_{eff}}{\lambda}$ the corresponding propagation constant. The perturbation by UV exposure, denoted by n_{eff}, of the refractive effective index along the fiber axis z is given by [32]:

$$\delta n_{eff}(z) = \overline{\delta n_{eff}}(z)(1 + \nu \cos\left[\frac{2\pi}{\Lambda} z + \phi(z)\right]) \quad z \in [0, L] \tag{3.1}$$

where L is the fiber length, Λ is the nominal grating period, $\overline{\delta n_{eff}}(z)$ is the slowly varying index amplitude change over the grating (also called *apodization*), ν is the fringe visibility and $\phi(z)$ is the slowly varying index phase change (also called *chirp*).

In the present work we consider a fiber with no chirp, i.e. $\phi = 0$, and the fringe of visibility $\nu = 1$.

While the modes are orthogonal in an ideal waveguide and therefore do not exchange energy, the presence of a dielectric perturbation causes the modes to be coupled. Introducing the detuning parameter:

$$\zeta(\lambda) = \beta - \frac{\pi}{\Lambda} = \frac{2\pi n_{eff}}{\lambda} - \frac{\pi}{\Lambda}$$

and the new unknowns:

$$R(z,\lambda) = A(z,\lambda)\exp(i\zeta(\lambda)z)$$
$$S(z,\lambda) = B(z,\lambda)\exp(-i\zeta(\lambda)z)$$

the coupled equations can be written as:

$$\frac{dR}{dz}(z,\lambda) = i\hat{\sigma}(z,\lambda)R(z,\lambda) + i\kappa(z)S(z,\lambda) \tag{3.2}$$

$$\frac{dS}{dz}(z,\lambda) = -i\hat{\sigma}(z,\lambda)S(z,\lambda) - i\overline{\kappa}(z)R(z,\lambda) \tag{3.3}$$

When δn_{eff} is small compared to n_{eff}, the two coupling coefficients can be approximated by [32]:

$$\hat{\sigma}(z,\lambda) = \zeta(\lambda) + \sigma(z) - \frac{1}{2} = \zeta(\lambda) + \beta\frac{\overline{\delta n_{eff}}(z)}{n_{eff}} - \frac{1}{2}$$

and

$$\kappa(z) = \frac{\overline{\delta n_{eff}}(z)}{\pi\lambda}$$

σ is called the 'dc' (demi coupling) and κ the 'ac' (associated coupling) coefficient.

The two modes coupling model is then completed by the following boundary conditions:

$$R(0,\lambda) = 1 \tag{3.4}$$

(the forward-going wave is incident from $-\infty$) and

$$S(L,\lambda) = 0 \tag{3.5}$$

(there is no backward-going wave for $z \geq \frac{L}{2}$).

The main characteristics of a FBG is then expressed through its complex spectral response in the transmission band given by:

$$\lambda \in [\lambda_{min}, \lambda_{max}] \mapsto \rho(\lambda) = \frac{S(0,\lambda)}{R(0,\lambda)} \tag{3.6}$$

from which we deduce its spectral response (power reflection function) at the entrance of a grating given by:

$$\lambda \in [\lambda_{min}, \lambda_{max}] \mapsto r(\lambda) = |\rho(\lambda)|^2 \tag{3.7}$$

There are two principal methods for calculating (3.6) and (3.7) that result from two-mode coupling in non uniform grating [32]: direct numerical integration and transfer matrix method. The former approach is the fastest and is adopted in this work.

3.2.2 Transfer matrix method

In this method we approximate the amplitude reflection coefficient (3.6) and the power reflection coefficient (3.7) by decomposing the FBG into a set of N uniform elementary fibers of length $\Delta = \frac{L}{N-1}$.

If the grating is uniform along z, then coefficients $\overline{\delta n_{eff}}$,$\kappa$ and σ are constant. Thus, equations (3.2) and (3.3) reduce to a system of coupled first-order ordinary differential equations with constant coefficients that can be exactly solved.

More precisely, we find that the amplitude reflection coefficient (3.6) and the power reflection coefficient (3.7) are respectively equal to:

$$\rho(\lambda) = \frac{-\kappa \sinh(\gamma L)}{\hat{\sigma}\sinh(\gamma L) + i\gamma \cosh(\gamma L)} \tag{3.8}$$

and

$$r(\lambda) = \frac{\sinh^2(\gamma L)}{\cosh^2(\gamma L) - \frac{\hat{\sigma}^2}{\kappa^2}} \tag{3.9}$$

where $\gamma^2 = \kappa^2 - \hat{\sigma}^2$.

The expressions (3.8) and (3.9) are used on each elementary fiber in order to find the overall coefficients of the complete FBG.

To do so, we first compute the exact solution of the system (3.2)-(3.3) assuming that the coupling coefficients are constants ($\kappa(0) = \kappa_0$ and $\hat{\sigma}(0) = \hat{\sigma}_0$). We found the following spectral response:

$$\rho_R(\lambda) = \frac{-q_0 sh(\gamma L)}{\gamma ch(\gamma L) - i\beta sh(\gamma L)} \tag{3.10}$$

where $\gamma_0^2 = \kappa_0^2 - \hat{\sigma}_0^2$.

Thus, we deduce that for the j^{th} section, we have:

$$\begin{bmatrix} R((j+1)\Delta, \lambda) \\ S((j+1)\Delta, \lambda) \end{bmatrix} = T_j \begin{bmatrix} R(j\Delta, \lambda) \\ S(j\Delta, \lambda) \end{bmatrix} \tag{3.11}$$

where T_j is a transfer matrix, given by:

$$T_j = \begin{bmatrix} ch(\gamma_j\Delta) + i\frac{\hat{\sigma}_j}{\gamma_j} sh(\gamma_j\Delta) & \frac{\overline{\kappa}_j}{\gamma_j} sh(\gamma_j\Delta) \\ \frac{\kappa_j}{\gamma_j} sh(\gamma_j\Delta) & ch(\gamma_j\Delta) + i\frac{\hat{\sigma}_j}{\gamma_j} sh(\gamma_j\Delta) \end{bmatrix} \tag{3.12}$$

with $\kappa_j = \kappa(j\Delta)$, $\hat{\sigma}_j = \hat{\sigma}(j\Delta)$ and $\gamma_j^2 = \kappa_j^2 - \hat{\sigma}_j^2$.

Using the following notation: $T = T_{N-2} \dots T_0 \equiv [T_{i,j}]_{(i,j)}$, we have the approximation of the spectral response:

$$\tilde{\rho}(\lambda) \simeq -\frac{T_{2,1}}{T_{2,2}} \tag{3.13}$$

the cost of this method is $O(NM)$ times the cost of the evaluation of the matrix (3.12) with M being the number of points of the spectral discretization and N the number of sections.

In order to decrease the computational time, we use a simplified version of this method where we approximate the matrix (3.12) T_j by:

$$T_j = T_\Delta + T_j^\rho \tag{3.14}$$

where

$$T_\Delta = \begin{bmatrix} \exp(i\hat{\sigma}\Delta) & 0 \\ 0 & \exp(-i\hat{\sigma}\Delta) \end{bmatrix} \tag{3.15}$$

represents the pure propagation matrix obtained when $\kappa \to 0$ in T_j and

$$T_j^\rho = \begin{bmatrix} ch(|\kappa_j|\Delta) & -sh(|\kappa_j|\Delta)\frac{\overline{\kappa_j}}{|\kappa_j|} \\ sh(|\kappa_j|\Delta)\frac{\kappa_j}{|\kappa_j|} & ch(|\kappa_j|\Delta)) \end{bmatrix} = \frac{1}{\sqrt{1-|\rho_j|^2}} \begin{bmatrix} 1 & -\overline{\rho}_j \\ -\rho_j & 1 \end{bmatrix} \tag{3.16}$$

With $\rho_j = -th(|\kappa_j|\Delta)\frac{\kappa_j}{|\kappa_j|}$.

(3.16) represents the reflection matrix obtained when $|\kappa_j| \to +\infty$ with $|\kappa_j|\Delta$ fixed in T_j.

Thus considering (3.15), $\tilde{\rho}$ becomes:

$$\tilde{\rho}(j\Delta, \lambda) = \frac{\rho_j + S((j+1)\Delta, \lambda)\exp^{2i\zeta\Delta}}{1 + \overline{\rho_j} S((j+1)\Delta, \lambda)\exp 2i\zeta\Delta} \quad 0 \le j \le N-3 \tag{3.17}$$

We can approximate $\rho(\lambda) \equiv \tilde{\rho}(\lambda)$ starting from $S(L, \lambda) = 0$.

3.2.3 Sampled FBG (SFBG)

A Sampled FBG (SFBG) is a superstructure made of a periodic distribution of a sampling pattern. This is an efficient and simple technique to construct a grating that exhibits periodic maxima in its spectrum. Using the coupled mode theory [32], it can be shown that a periodic spatial distribution of the sampling pattern leads to a comb of peaks with identical response and with a wavelength separation $\Delta\lambda$ varying as:

$$\Delta\lambda = \frac{2n_{eff}\Lambda^2}{P} \tag{3.18}$$

with P the period of the sampling patterns distribution.

The computation of the reflectivity coefficients of a sampled FBG remains unchanged compared to what we have exposed in previous paragraph. But this calculation is more time consuming.

3.3 Optimization problems

In this section we define and solve three optimization problems based on the design of FBG filters.

3.3.1 Pass-band filter design

GA and SD2A are first applied to the design of pass-band filters compound by an unique FBG. The objective is to build an optical filter which reflect a wavelength band $[\lambda_1, \lambda_2]$ (i.e. a spectral response equals to one in $[\lambda_1, \lambda_2]$ and equals to zero elsewhere). The spectral response of the perfect passband-filter, denoted by $F_{[\lambda_1,\lambda_2]}$, is presented on Figure 3.2. This problem was already studied using various optimization algorithms, such as GAs [33].

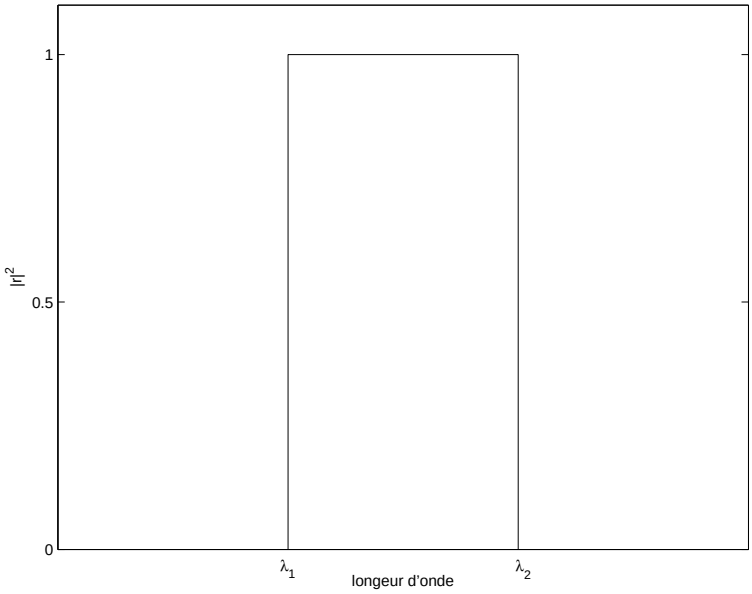


Figure 3.2: spectral response of an ideal pass-band filter between λ_1 and λ_2

Description of Pass-band filter inverse problem

We consider a FBG with the following characteristics $n_{eff} = 1.45$, $L = 31mm$ and $\Lambda = 0.53\mu m$.

We have seen, in previous section, that a FBG with no chirp can be characterized by its effective refractive index modulation. In order to find a FBG with the desired spectral response and associated with an index modulation satisfying given industrial constraints (it's difficult to build a fiber with a refractive index modulation containing too many variations and oscillations in its amplitude [10]), apodization profiles are generated by spline interpolation through a reduced number of N_S points equally distributed along the fiber. Also for technical reasons, the index variation is included into a bounded interval $[-\overline{n}_{max}, +\overline{n}_{max}]$. Thus the corresponding search space of the optimization problem is an hypercube:

$$\Omega_{N_S} = [-\overline{n}_{max}, +\overline{n}_{max}]^{N_S} \tag{3.19}$$

Here $\overline{n}_{max} = 5 \times 10^{-4}$ and $N_S = 8$.

	GA	SD2A
J_{N_c} value	3.9	2.
Evaluation number	5400	1400

Table 3.1: Pass-band filters: Results for GA and SD2A optimization.

The functional on Ω_{N_S} to minimize is an error-type function that computes the difference between the spectral response of a FBG associated to the apodization $x \in \Theta_{N_S}$, denoted by $Spec_x$, and the spectral response of the target filter $F_{[\lambda_1,\lambda_2]}$. It's evaluated on $N_c = 100$ wavelengths equally distributed on the transmission band $[\lambda_1, \lambda_2]$:

$$J_{N_c}(x) = \sum_{i=1}^{N_c} (|Spec_x(\lambda_i)|^2 - |F_{[\lambda_1,\lambda_2]}(\lambda_i)|^2)^{\frac{3}{4}} d\lambda \tag{3.20}$$

where :

$$F_{[\lambda_1,\lambda_2]}(\lambda) = \begin{cases} 1 \text{ if } \lambda \in [\lambda_1, \lambda_2] \\ 0 \text{ elsewhere} \end{cases} \tag{3.21}$$

The choice of the norm in J is important. Indeed, it determines if a filter is more penalized by its side lobes (i.e. parasitical side peaks) or by the difference between its passband and the ideal passband. The norm chosen was proposed by Skaar J. (see [10]) to obtain the best compromise between those two characteristics.

Results and discussion

The two-level SD2A and GA algorithms, with parameters fixed in chapter 2, are applied to the minimization of previous functional J_{N_c}, in order to design a pass band filter reflecting the wavelength band between $\lambda_1 = 1.55\mu m$ et $\lambda_2 = 1.5503\mu m$.

For the GA optimized filter, the error function J_{N_c} is equal to 3.9. Result was found after 5400 error function evaluations. The SD2A optimized filter has an error function equals to 2, the result was found after 1400 evaluations. Our semi-deterministic algorithm permits to reduce the computational time and the value of J_{N_c} by two in comparison to GA. Furthermore the SD2A-filter has smaller side-lobes and his passband is closer to the ideal passband. SD2A leads, for this problem, to a better solution than GA. Apodization, Spectral responses and convergence histories for GA and SD2A optimizations are presented in Figure 3.3. Results are summarized in Table 3.1.

We are now interested to apply both algorithms to the synthesis of a more complex optical filter compound by SFBG.

3.3.2 Multichannel Filter design

SD2A, HSGA and GA algorithms are applied to solve two examples of inverse problems relative to the design of Multichannel SFBG Filters. The objective is to construct two multichannel filters centered around $1552.5nm$ that consist either of $N_{peaks} = 16$ or $N_{peaks} = 38$ totally reflective identical channels spaced of 0.8 nm. The SFBG characteristics are set to $n_{eff} = 1.45$, $L = 100mm$, $P = 1.039mm$ and $\Lambda = 0.53\mu m$.

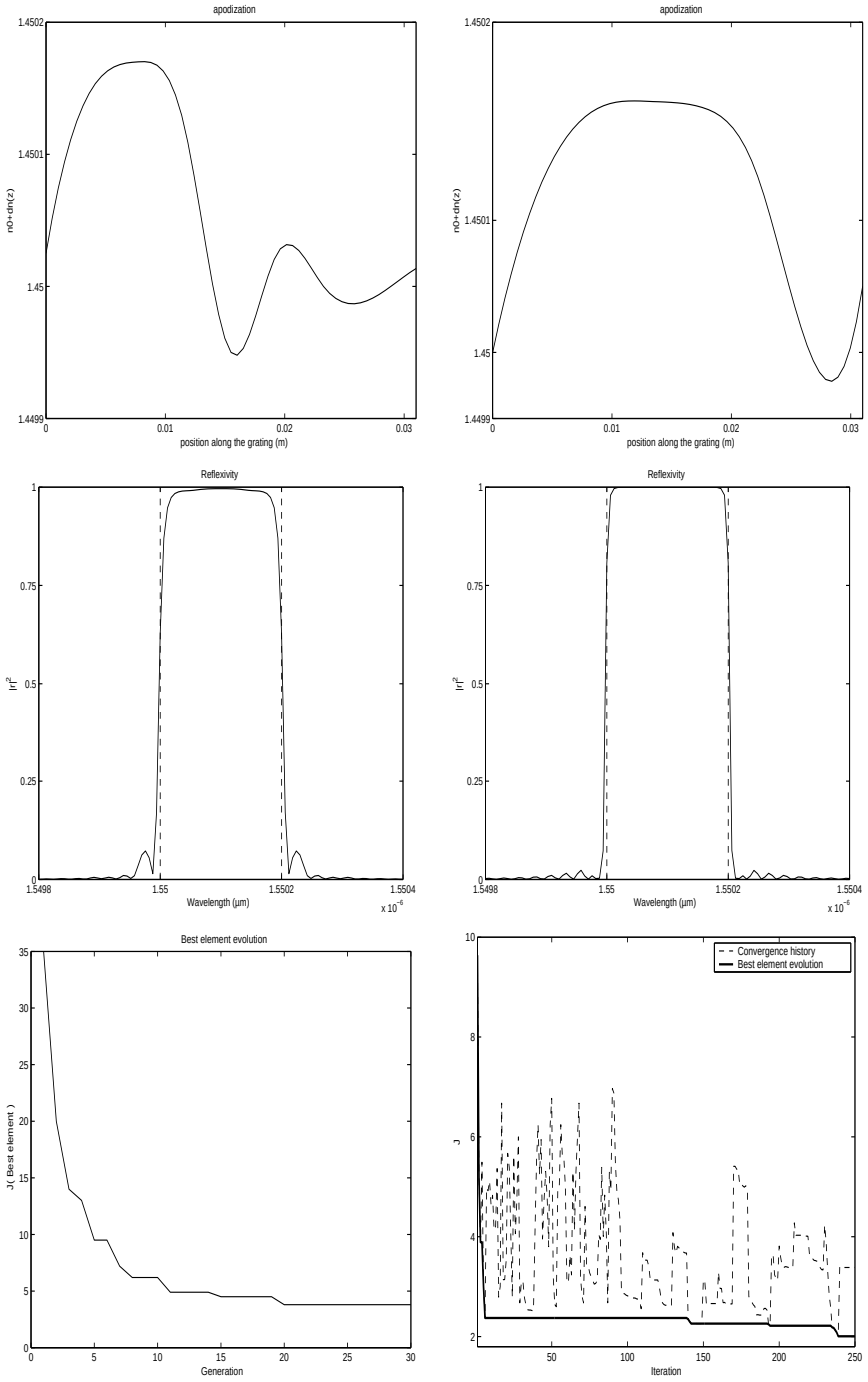


Figure 3.3: Pass-band filters: The first column presents the result obtained with the GA, the second column the result obtained with SD2A. The first line shows the effective index modulation of optimized filters, the second line the associated spectra. Finally the third line presents the convergence histories.

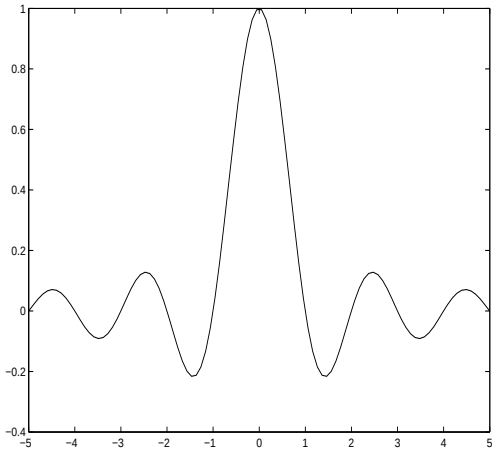


Figure 3.4: Example of a Sinc function graph.

Current design of multichannel filters based on Sampled FBGs

Pioneer works with this approach have reported results obtained with amplitude only sampling through the writing of periodic transmitting and opaque regions inside the fiber core [34]. But, the main drawback in this case is the non-uniformity of the spectral response due to the square index envelope of every transmitting region. The uniformity of the response is then limited to only a few nanometers. In order to overcome this non-uniformity, it was proposed to insulate the opaque regions with sets of interleaved amplitude-only sampling patterns for yielding interleaved groups of channels [35]. However, a predictable number of peaks inside an overall square envelope is still difficult to reach in this case. A lot of work was then performed for optimizing the sampling pattern either in amplitude [36] and/or in phase [37, 31]. In the case of amplitude optimization, the index modulation amplitude of the sampling was optimized following a Fourier transform approach. Ibsen et al. [36] have shown the possibility to inscribe inside the fiber core a Sinc-shaped apodization function with a continuous writing technique:

$$\overline{\delta n_{eff}}(z) = (\delta n_{eff})_{max} \frac{\sin(\frac{2N_p \pi z}{L})}{\frac{2N_p \pi z}{L}} \tag{3.22}$$

The number of secondary side loops of this function, $(2 * (N_p - 1))$, being directly proportional to the total number N_p of targeted channels, a 16-peaks filter was demonstrated with a profile exhibiting 14 secondary side loops between the main lobes (Example of Sinc graph is given by Figure 3.4. To physically express the negative values of the Sinc function, discrete π-phase shifts were inserted between each side loop. However, this type of profile suffers several drawbacks. The mathematical Sinc function consists of a main lobe and secondary side loops which proportion must be perfectly respected and a great number of π-phase shifts must be inserted as the number of channels increases. Therefore, a perfect control and mastering of the writing process is required. Moreover, as the number of channels increases, the duty cycle, ratio between the central lobe and the rest of the grating, decreases proportionally to $\frac{1}{N}$. Therefore, most of the energy deposited onto the

fiber finally concerns the writing of the central lobe so that the strength of reflectivity of the grating strongly decreases. In this way, a 16-channels count FBG with a reflectivity of 0.95 requires a maximum index variation up to 6.10^{-4}. Such values are difficult to reach in practice and they imply to use very photosensitive fibers which, in turn, may give rise to a non linear regime during the UV-writing. In the case of phase-only optimization, only the relative phase of the channels to be generated is continuously modified and the amplitude modulation of the sampling is minimized. Such an approach was first demonstrated with semiconductor laser Bragg reflectors [38] and it was recently applied to FBGs [31]. An optimized phase-sampling profile was proposed and experimentally demonstrated for generating a 9-channels count filter. In this case, a customized lithographically prepared phase mask was used but the total number of channels remained relatively low.

As we can see, there is an important demand for new optimization methods in order to solve this kind of problem.

Description of SFBG inverse problem

As previously, we can reformulate this problem considering that each SFBG is characterized by its apodization. In order to find a SFBG with the desired spectral response and associated with an index modulation with some 'interesting' industrial characteristics (smooth enough, slowly varying, reduced sign change number and symmetric), apodization profiles are generated by spline interpolation through N_S points equally distributed along the first half of the sampling pattern and completed by parity. We will thus choose a value of N_S high enough to ensure a large number of peaks in the spectral response but small enough to ensure improvement in the index modulation profile in comparison with the classical Sinc profile. More precisely, the corresponding search space of the optimization problem is the hypercube:

$$\Omega_{N_S} = [-\overline{n}_{max}, \overline{n}_{max}]^{N_S} \tag{3.23}$$

where $\overline{n}_{max}$, the maximum amplitude variation, is a design constraint. Here $\overline{n}_{max} = 5\times10^{-4}$ and $N_s = 9$ or 20 respectively for the two considered inverse problems ($N_{peaks} = 16$ and $N_{peaks} = 38$).

The function on Ω_{N_S} to minimize is defined by:

$$J_{N_c}(x) = \sum_{i=1}^{N_c} (r(x)(\lambda_i) - r_{target}(x)(\lambda_i))^2 \tag{3.24}$$

In the above expression, the power reflection function, $r(x)$ of the filter with an apodization associated with $x \in \Omega_{N_S}$ is evaluated on N_c (namely 1200 and 3000 respectively) wavelengths equally distributed on the transmission band by using the simplified transfer matrix method described previously [32]. $r_{target}(x)$ denotes the nearest perfect power reflection function to $r(x)$ matching the desired requirements, namely N_{peaks} transmitted wavelengths with a perfect transmission rate and separated by $\Delta\lambda$:

$$r_{target}(x)(\lambda) = \begin{cases} 1 \text{ if } \lambda \in \Lambda \\ 0 \text{ elsewhere} \end{cases} \tag{3.25}$$

with $\Lambda = \{\lambda_x, \lambda_x + \Delta\lambda, \ldots, \lambda_x + (N_{peaks} - 1)\triangle\lambda\} \subset [\lambda_{min}, \lambda_{max}]$.

Results and discussion

The two-level SD2A and GA algorithms have been used for the minimization of the previous functional J_{N_c} in order to design two non-chirped apodized SFBG exhibiting 16, respectively 38, totally reflective channels spaced of $0.8nm$ (or 100GHz). We discuss their characteristics and compare them to those generated by a classical sinc-type apodization profiles.

N.B.: For the 16 peaks filter, we have also used the HSGA algorithm. In fact, when this work was performed HSGA algorithm was not yet implemented. Later, we have decided to apply this algorithm to one optical fiber application. HGSA is applied with parameters fixed in chapter 2.

16 peaks filter Figure 3.5 and 3.6 show the apodization profiles and the associated power reflection functions between 1540 and 1560 μm of four different filters: Sinc, HSGA,GA and SD2A. Convergence histories are given in Figure 3.7.

The Sinc filter has a cost function's value of 6.59.

The GA filter gives 5.9. The total number of functional evaluations is 5400. Evaluation time is close to 24 hours.

For the SD2A filter, J_{N_c} is equal to 3.09. The total number of functional evaluations is about 3000. Furthermore, in order to reduce computational time during the SD2A optimization process, gradient computation are performed on a reduced number of point $N_c = 300$. SD2A optimization takes about 3 hours.

The HSGA filter gives 4.9. The number of functional evaluations is 2400. Evaluation time is near to 10 hours.

This is important as Sinc profiles are considered as efficient and realistic for the realization of these kind of filters. The four values of the cost function show that HSGA,GA and SD2A algorithms have led to better solutions in terms of reflection characteristics than the Sinc profile. This is visible in the second column of Figure 3.5 and 3.6 on the respective interchannel and out-of-band rejection (undesirable side peaks) which is reduced to 0.2 in the SD2A profile and to 0.3 in the GA and HSGA ones to be compared to the value of 0.4 for the Sinc profile. In addition, optimized profiles are more suitable for industrial realization as the number of necessary phase shifts (when $\overline{\delta n_{eff}}(z) = 0$) is only 6 for the GA and HSGA profiles and 4 for SD2A one against 14 for the Sinc-profile. Furthermore the index modulation of optimized profiles is more homogeneously distributed along the pattern and does not exhibit any dominant lobe. A last improvement concerns the maximum amplitude of the profiles which has been reduced to $\Delta\overline{n}_{max} = 1.910^{-4}$ for optimized profiles. These results are summarized in Table 3.2.

38 peaks filter Here we see why Sinc profiles are appreciated in industry. Indeed, the classical Sinc-type profile exhibits better spectral characteristics than both SD2A and GA optimized profiles as it can be observed in Figure 3.8 or in their corresponding values by J_{N_c} (9.1 against 10.3 and 13.9). However, the sinc-type profile is difficult to built because of its large amount of phase shifts (36) and its large maximum index amplitude (6×10^{-4}). GA have failed to find a filter with a 38 totally reflective peaks spectrum.

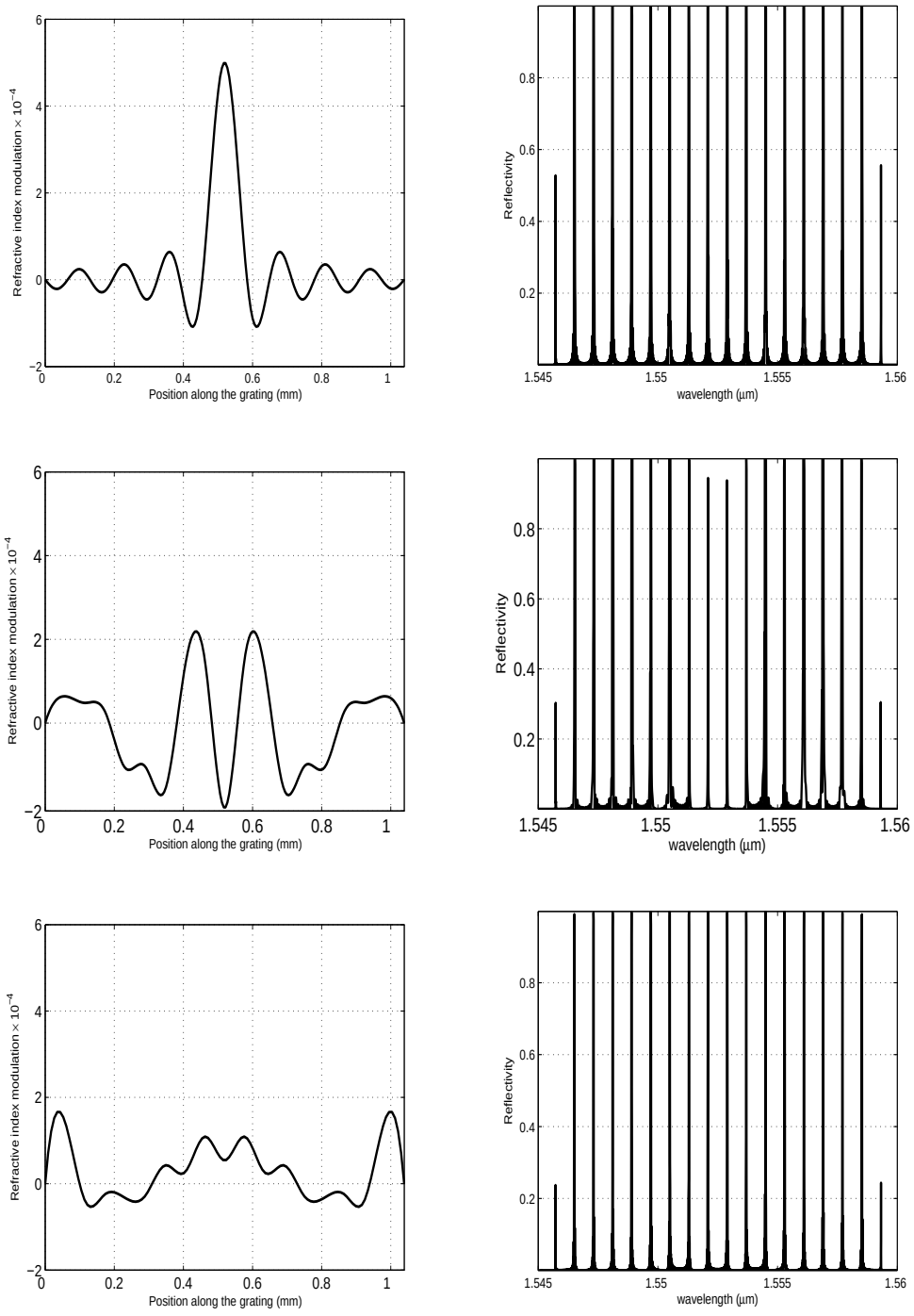


Figure 3.5: 16 peaks multichannel filters: from (**Top**) to (**Bottom**) with Sinc, GA and SD2A. (**Left**) Apodization profiles : $\overline{\delta n_{eff}}(z)$ ($\times 10^{-4}$) vs. length (mm). (**Right**) Associated power reflection function or reflexivity vs. wavelength (μm). SD2A results is easier to build and gives better spectrum.

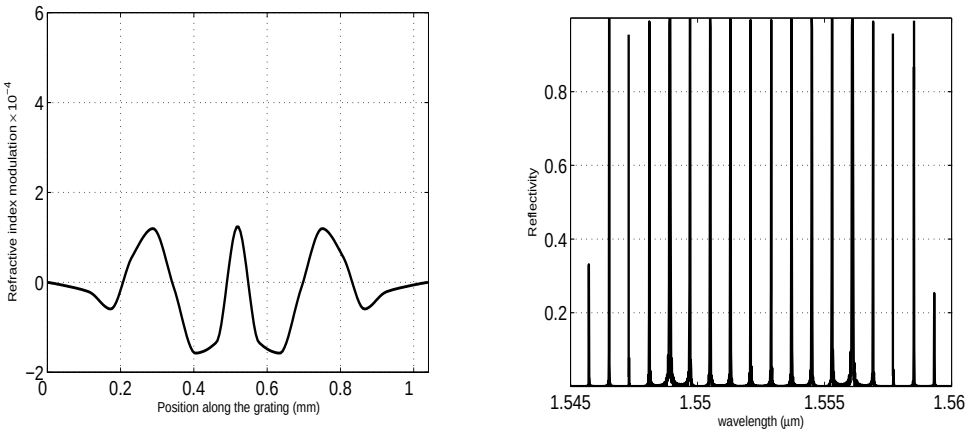


Figure 3.6: 16 peaks multichannel filters. (**Left**): refractive index modulation profile. (**Right**): Associated spectrum: reflexivity vs. wavelength (μm).

	Sinc	GA	SD2A	HSGA
Cost function value	6.6	5.9	3.1	4.9
Total evaluation number	$\cdots$	5400	3000	2500
Computational time	$\cdots$	24h	3h	10h
Out-of-band rejection	0.5	0.3	0.2	0.3
$\Delta\overline{n}_{max}$	5 10^{-4}	1.9 10^{-4}	1.9 10^{-4}	1.8 10^{-4}
Phase shifts	14	6	4	6

Table 3.2: 16 peaks multichannel filters: (**Left**) to (**Right**), results with Sinc, GA, SD2A and HSGA techniques.

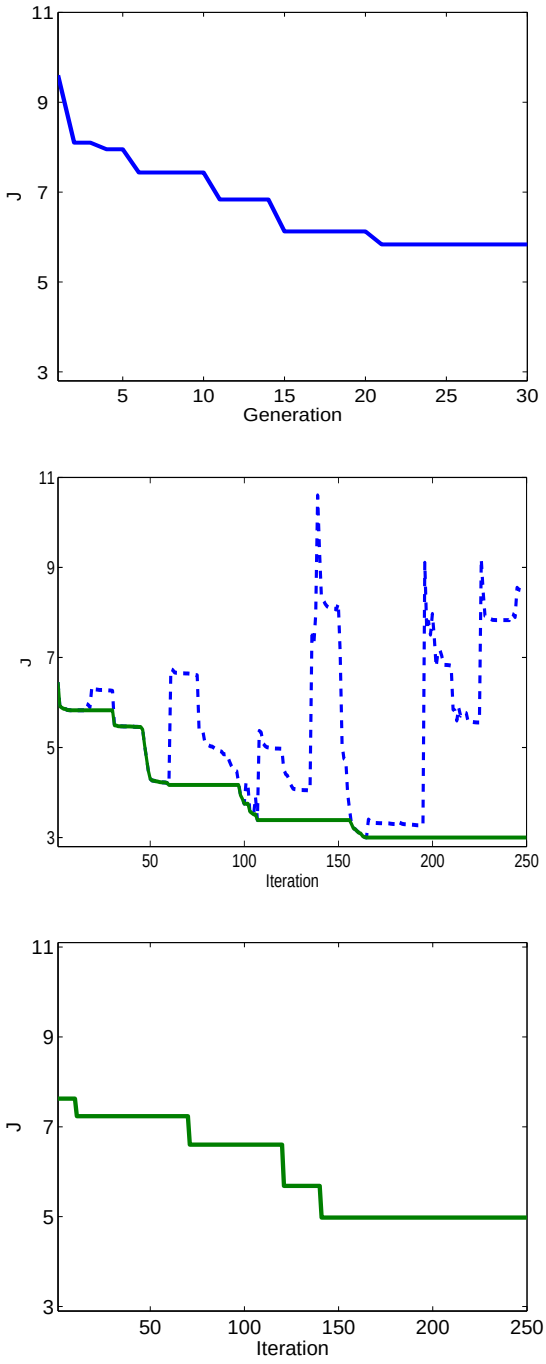


Figure 3.7: 16 peaks multichannel filters. (**Up**) AG, (**Center**) SD2A and (**Bottom**) HGSA histories: Best element evolution (**solid line**) and global convergence (**dashed line**).

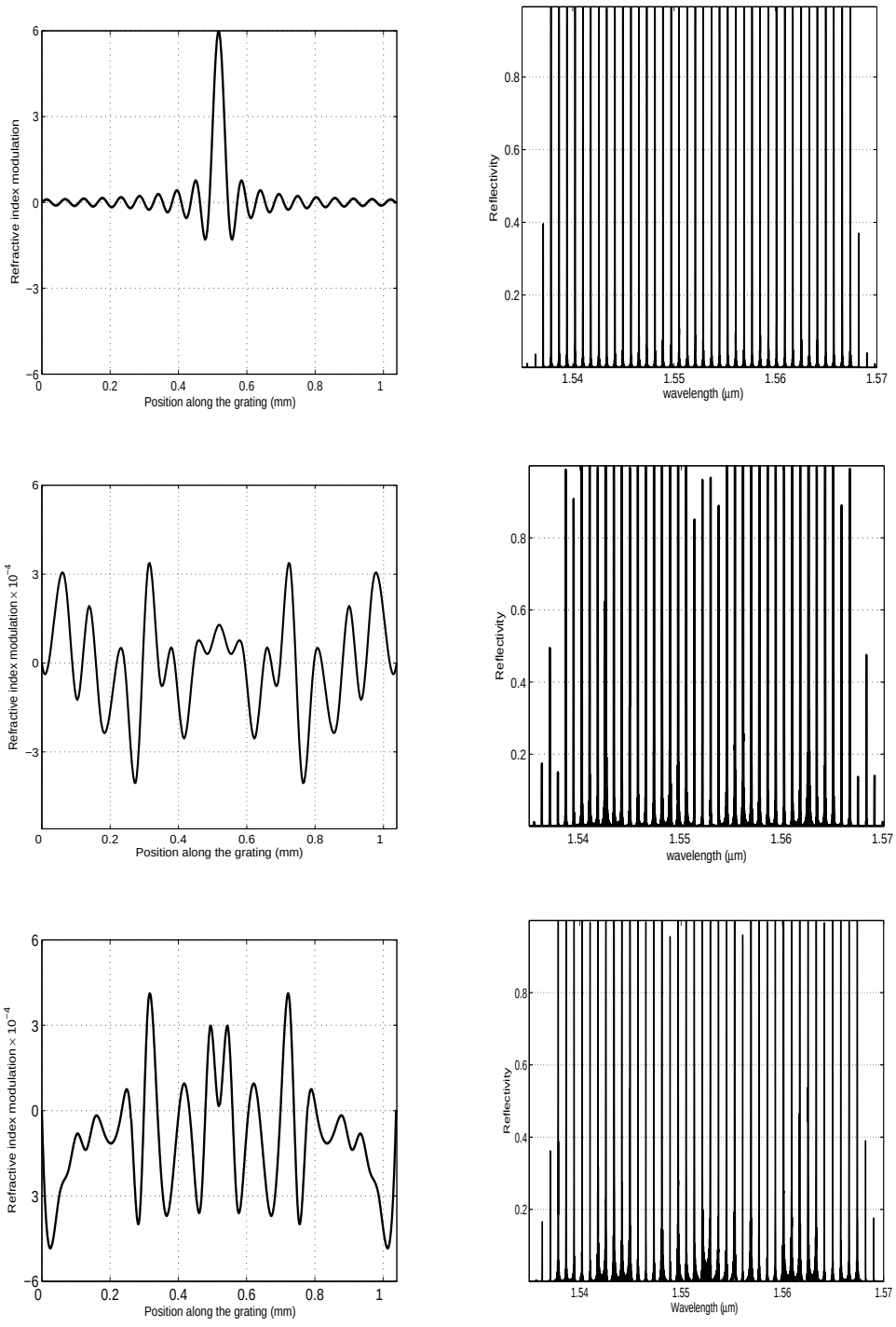


Figure 3.8: 38 peaks multichannel filters: from (**Top**) to (**Bottom**) with Sinc, GA and SD2A. (**Left**) Apodization profiles : $\overline{\delta n_{eff}}(z)$ ($\times 10^{-4}$) vs. length (mm). (**Right**) Associated power reflection function or reflexivity vs. wavelength (μm).

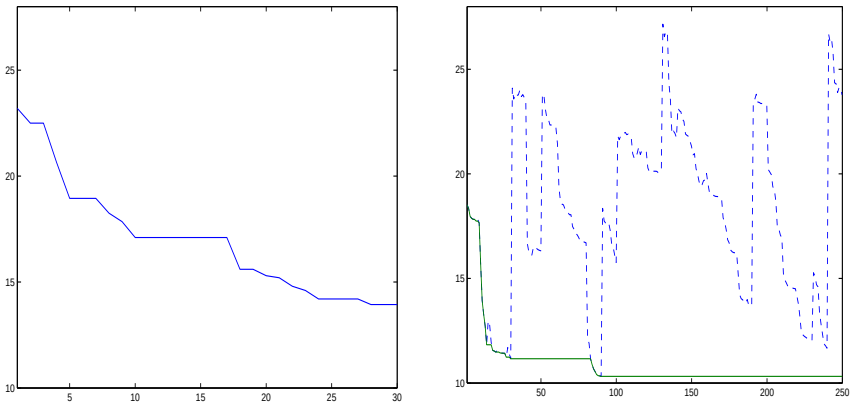


Figure 3.9: 38 peaks multichannel filter. **Left:** Best element convergence history vs. generation iterations for GA. **Right:** Best element convergence history vs. accumulation of optimization iterations (solid line) and global convergence history (dashed line) for SD2A. function vs. iteration.

	Sinc	GA	SD2A
Cost function value	9.8	13.9	10.3
Total evaluation number	$\cdots$	5400	4500
Computational time	$\cdots$	60h	12h
Perfect Peak number	38	34	38
$\Delta\overline{n}_{max}$	$6\ 10^{-4}$	$5.5\ 10^{-4}$	$3.5\ 10^{-4}$
Phase shifts	37	21	16

Table 3.3: 38 peaks multichannel filters: (**Left**) to (**Right**), results with Sinc, GA and SD2A techniques.

The total number of functional evaluations for GA is almost unchanged at 5400. GA optimization time is about 60 hours.

Actually, the SD2A algorithm could have found equivalent or better results than the sinc-type profile if a larger value N_s had been taken (e.g. $N_s > 40$). But to avoid the above-mentioned experimental difficulty, we explicitly removed such unsuitable profiles with the choice of a reduced dimension of the search space. Thus, the obtained SFBG is easier to implement as the number of phase shifts is reduced to 14, the index modulation is more homogeneously distributed and the maximum amplitude of the profile is reduced to $\Delta\overline{n}_{max} = 4.8 \times 10^{-4}$. Furthermore, the associated spectrum still industrially applicable due to the fact that in practice we only need at least 95%-reflective peaks [28].

the computational time for the SD2A optimization is here equal to 12 hours for a total number of 4500 evaluations of the cost function J_{N_c}. Gradient computation are performed on a reduced number of point $N_c = 1200$.

Convergence histories are presented in Figure 3.9. Results are summarized in Table 3.3.

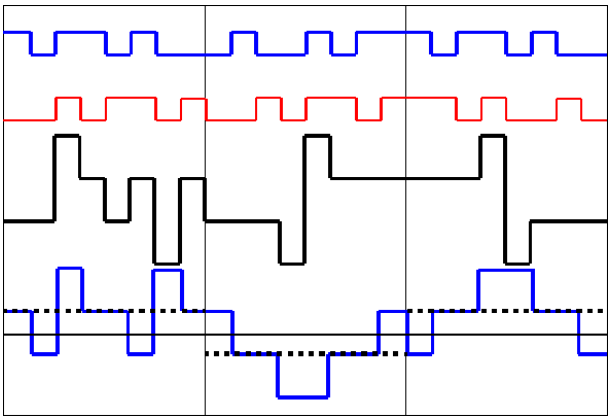


Figure 3.10: From (**Top**) to (**Bottom**): **1-** coded signal for a three bit message (A) [1,0,1] where "1" is replaced by an 8 chip sequence, and "0" its complement **2-** coded signal for a three bit message (A) [0,0,1] where "1" and is replaced by another 8-chip sequence orthogonal with the chip sequence for A. **3-** transmitted signal (sum of coded signals), **4-** transmitted signal is "multiplied" by sequence for A, thus the message A is recovered by averaging(- - -).

3.3.3 CDMA Filter design

In this section only SD2A algorithm is applied to the design of a SFBG Filter that will be used into a CDMA device. As previously, the objective is to construct a multichannel filter centered around $1552.5\mu m$ that consist either of $N_{peaks} = 8$ channels spaced of 0.8 nm. But in this new application, we need to introduce gap in the spectrum, i.e. we need to set the reflexivity of predetermined channels to 0. The SFBG still have the same characteristics: $n_{eff} = 1.45$, $L = 100mm$, $P = 1.039mm$ and $\Lambda = 0.53\mu m$.

CDMA Principle

Optical fibers, which offers a large bandwidth (about five TeraHertz per telecommunication window), can be fully used only if the techniques of multiple access are sufficiently effective. Three schemes of access are possible to share the large bandwidth of an optical communication link: Time sharing (TDMA) which authorizes a great number of users but requires fast synchronization, wavelength sharing (WDM) [39], which sometimes requires precise adjustments, and sharing with code division (CDMA) [40] which primarily allows a great flexibility in multiple accesses. Naturally those techniques can be combined.

In the basic technique of Code Division Multiple Access (CDMA), each bit "1" or "0" of a binary message is replaced at the level of the transmitter by a code attributed to each user. This message code is a pseudo-random time sequence of "chips" (i.e. positive or negative impulsions: [-1, 1, 1, -1,...]) and its complement for "0" (i.e. [1, -1, -1, 1,...]).

In reception the correlation with the specific codes makes it possible to extract the messages (Figure 3.11). The codes are to be orthogonal (orthogonal codes of Walsh, or m-sequences, or Gold codes) to ensure a negligible correlation when the wrong code is used (See Figure 3.10 for an example of codes).

The application in optics of the techniques of spectrum to share transmission medium cannot be a simple transposition of the techniques used in radio frequencies (mobile telephony of third generation or GPS system), because detection rests a priori only on the intensity of the signal. The principles were formulated by Salehi [41] which suggested the use of unipolar orthogonal codes [0,1] so-called "optical codes". The symbol "1" is coded by a time sequence of short pulses (10-100 psec i.e. spread spectrum) obtained with a network of parallel lines with delays ("tapped delay lines").

The decoder consists of combined delay lines which realign the pulses issued from the transmitter chosen, which are added to rebuild an impulse of strong amplitude: it needs a non-linear element to discriminate simultaneous pulses from a sequence of close pulses. The pulses which don't correspond appear as noise. Because only positive intensity is exploitable, the system is unipolar, the quantity of pulses ("weight")/symbol of message is weak, the number of orthogonal codes, thus the number of users is weak. Many alternatives were explored to improve this technique.

Bipolarity is necessary to optimize the number of orthogonal codes: the sums of the pulses of both polarities are compared :

- equal if there is no correlation between the received signal and selected code.
- unequal if the message is "1" or "0".

Bipolarity may be introduced by using - two phases [0, π], separated with detection, - two wavelengths, two physical channels, two polarizations or temporal multiplexing.

Zaccarin and Kavehrad [42], then Lam [43] suggested a spectral variant of CDMA: the code is a spectrum characterized by a set of wavelengths. It is possible to design devices where the code is defined in the wavelengths-domain. A broad source - controlled electronically at the rate of the symbols is filtered: each bit " 1 " or " 0 " is a selected definite unit among 2xN wavelengths. Since there is no cyclic risk of correlation, one can use orthogonal codes like the codes of Walsh. If one is satisfied with codes with weak but non-null periodic autocorrelation, the number of users can be about N ("Gold codes") by considering N possible shifts distributed among the users.

Problem formulation

We propose here to design a part of a spectral CDMA transmitter/receptor which are depicted in Figure 3.11. More precisely, we will design the code separator device of a particular user 'A'. The objective of this code separator is to separate the spectrum corresponding to the bit '1' and the bit '0' of the user A. It's compound by:

- An Optical Isolator: When a signal enter in the forward sense the signal is redirected to a SFBG fiber. When a signal enter in the backward sense it is redirected to a classical fiber.
- A SFBG which will reflect the spectrum corresponding to bit '1'. This part will be designed using SD2A.
- A classical optical fiber.

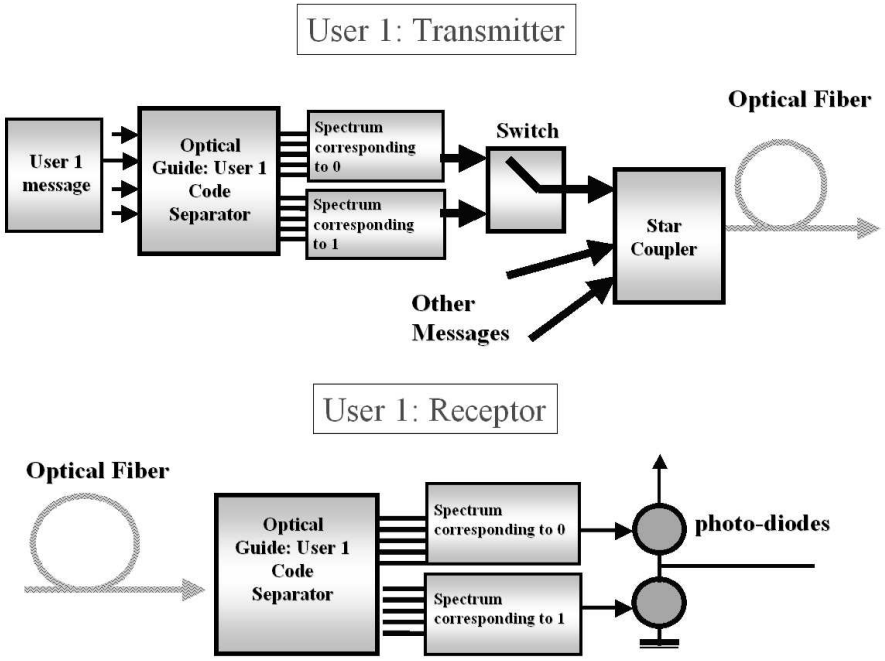


Figure 3.11: CDMA transmitter and receptor devices.

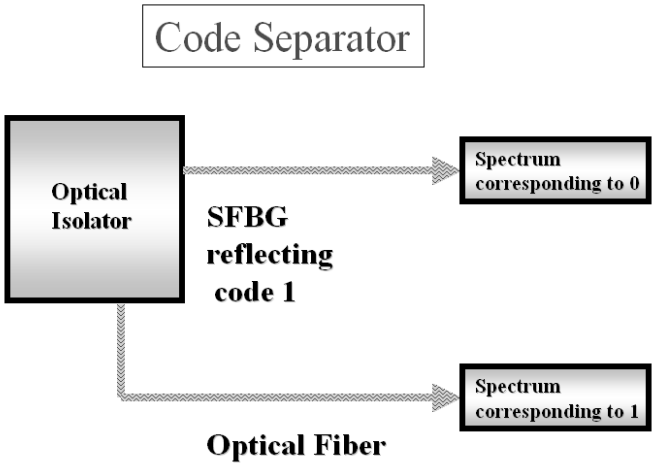


Figure 3.12: Code separator device.

	SD2A
Cost function value	2.0
Total evaluation number	3000
Computational time	3h

Table 3.4: CDMA multichannel filter: result obtained with SD2A optimization technique.

This device is described by Figure 3.12.

We consider a spectral code generated by $N_{code} = 8$ wavelengths:

$\lambda_1 = 1.5521\mu m$, $\lambda_2 = 1.5473\mu m$, $\lambda_3 = 1.5481\mu m$, $\lambda_4 = 1.5489\mu m$, $\lambda_5 = 1.5497\mu m$, $\lambda_6 = 1.5505\mu m$, $\lambda_7 = 1.5513\mu m$, $\lambda_8 = 1.5521\mu m$.

The code corresponding to '1' for user A is defined by: $[\lambda_1, \lambda_3, \lambda_4, \lambda_7, \lambda_8]$.

Due to the fact that considered SFBGs can only generate symmetrical spectra (this was observed numerically using the previous model combined with various symmetric and non-symmetric apodization profiles), we will generate a SFBG filter that reflects:

$$\Lambda = [\lambda_1, \lambda_3, \lambda_4, \lambda_7, \lambda_8, \lambda'_1, \lambda'_3, \lambda'_4, \lambda'_7, \lambda'_8]$$

where λ'_i for $i = 1, ..., 8$ correspond to the symmetrical wavelength of λ_i centered around $1.5525\mu m$.

As previously, the SFBG apodization profile are generated by $N_{peaks} = 9$ interpolation points. The functional on Ω_{N_S} to minimize is (3.24) with $N_c = 1200$ and $r_{target}(x)$ defined by:

$$r_{target}(x)(\lambda) = \begin{cases} 1 \; if \; \lambda \in \Lambda \\ 0 \text{ elsewhere} \end{cases} \tag{3.26}$$

Results and discussion

Due to the efficiency of the two-level SD2A used in previous applications, in comparison to GA, this algorithm is the unique used in order to design the SFBG that corresponds to the spectrum Λ.

Figure 3.13 shows the apodization variation profile, the associated power reflection function between 1,545 and 1,560 μm, and the convergence history obtained with SD2A optimization.

The error function J_{N_c} is equal to 2.09. The total number of functional evaluations is about 3000. Gradient computation still be performed on a reduced number of point $N_c = 300$. SD2A optimization takes about 3 hours. These results are summarized in Table 3.4.

The optimized spectrum corresponds to the considered user A code '1'. The optimized apodization profile is suitable for industrial implementation. The number of necessary phase shifts is 5 and the maximum amplitude of the profile is $\Delta\overline{n}_{max} = 2.10^{-4}$.

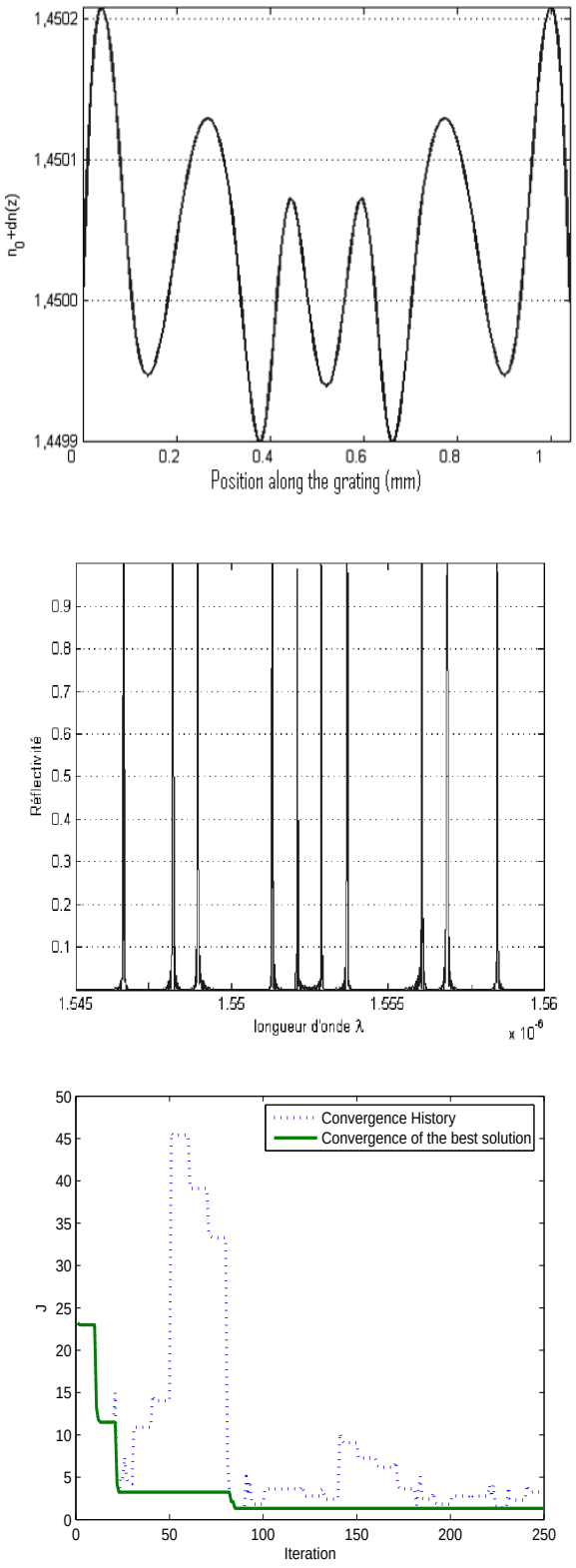


Figure 3.13: CDMA filter: from (**Top**) to (**Bottom**) SD2A optimized apodization profile, associated spectrum and convergence histories.

3.4 Conclusions

Various kind of optical filters have been synthesized using three global optimization approaches: our original SD2A and HSGA optimization algorithms and a genetic algorithm. When it was possible results have been compared to a classical industrial method. The SD2A technique gives superior results: the grating solutions produced are easier to build from an industrial point of view and the spectra are industrially exploitable.

A next step could be the study, for CDMA device synthesis, of the effect of combined apodization and phase profiles optimization in order to avoid the symmetry in spectra mentioned previously. During this work, we have been interested only by apodization optimization in order to keep a grating easy to implement by any optical laboratory. Indeed, phase variation requires more complex and expansive materials.

Chapter 4

Shape Optimization of a Fast-Microfluidic Mixer Devices

4.1 Introduction

Microfluidic channel systems used in bio-analytical applications are fabricated using technologies derived from microelectronics industry including lithography, wet etching and bonding of substrates. Industrial applications of these techniques concern DNA sequencing, new drug molecules trials, pollution detection in water or food and protein folding (See figure 4.1).

Focusing on the last domain, the problem described in this chapter is to reduce the mixing time between a given protein and solvent (buffer) of a considered micromixer by finding an optimum channel geometry. The optimization begins with a previous mixer design [44] and varies the channel shapes at the mixer intersection.

In section 2, we recall the state of the art and previous works on fast microfluidic protein folding devices design. Section 3 introduces a short presentation of the physic of the problem and its numerical solution. Section 4 shows optimization results obtained with the three optimization algorithms SD2A, HSGA and GA presented in chapter 2. Finally, in section 5 we compare numerical values with those obtained by experimental

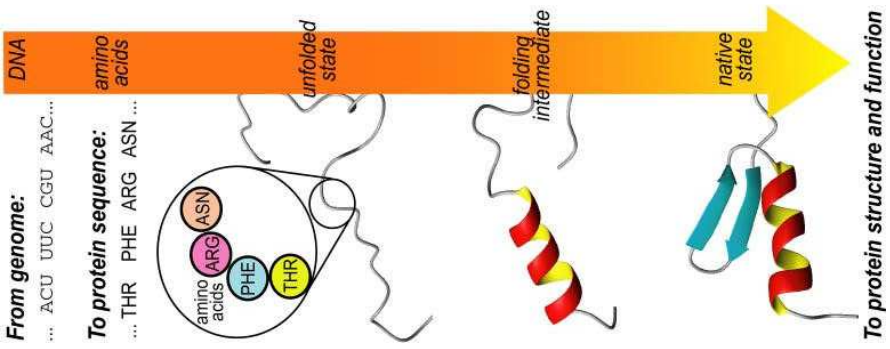


Figure 4.1: Folding process scheme: The geometrical structure of the unfolded protein change.

implementation.

4.2 Fast Microfluidic Protein Folding device

Important structural events occur on a microsecond time scale [45]. To resolve folding events of order 10 microseconds, mixer designs are required that effect mixing in a few microseconds or less. This can be performed, for instance, using photochemical initiation [46] and changes in temperature [47], pressure [48, 49] or chemical potential [50, 51, 52]. All these techniques provide perturbations of protein conformational equilibrium necessary to initiate folding. In comparison to temperature and pressure-jump relaxation technique, folding experiments based on changes in chemical potential, via rapid mixing of protein solutions into and out of chaotrope solvents, are more versatile. The technique is applicable to a wide range of proteins as most unfold reversibly in the presence of chemical denaturants such as urea and guanidine hydrochloride (GdCl) [50].

Until recently, the main limitation of mixer-based experiments was their inability to access very short timescales. Mixing time is ultimately limited by the time required for molecular diffusion across a finite length scale, and diffusion time scales as the square of diffusion length. Brody et al. [53] first proposed rapid mixers based on hydrodynamic focusing as a way to address the issue of reducing diffusion lengths under laminar flow conditions while minimizing sample consumption. Hydrodynamic focusing has been used to measure protein and RNA folding [54], with mixing times of a few hundreds of microseconds.

This chapter discusses specific shape optimization for a new microfluidic mixer based on a continuous flow principle originally proposed by Knight et al. [5], and improved and demonstrated by Hertzog et al. [44]. The design leverages hydrodynamic focusing on the micron scale to reduce diffusion lengths. This mixer uses about eight orders of magnitude less labelled protein sample mass flow than a previously reported ultra-fast protein folding mixer [55], with flow rates of 3 nl/s and protein concentrations of tens of nanomolar.

The original mixer design, presented in Figure 4.7-Left, was found using a parametric optimization method in which the system was simplified with five nondimensional parameters to describe the problem. These five nondimensional parameters included two length scale ratios, a flowrate ratio, a Peclet number, and a Reynolds number. The shape of the mixing region was fixed to that of an intersection at 90 degrees channels of varying width, with only variations in the inlet and exit widths permitted. General trends were found by independently varying each of the five parameters while keeping the other four fixed. These trends guided the geometry towards small inlet nozzles and high Reynolds numbers, which were bounded by a variety of physical constraints. These constraints included minimum features sizes set by the microfabrication techniques, variations in flow fields from the two-dimensional approximation, and microchannel clogging issues. The previous optimization enabled us to find only a local minimum in the cost function and was bounded heavily by the small number of parameters, especially in the mixer geometry.

The optimization technique used in this chapter is aimed at minimizing the mixing time by investigating complex variations in the channel shapes. This technique is more robust than the previous method and allows for non-intuitive designs to emerge.

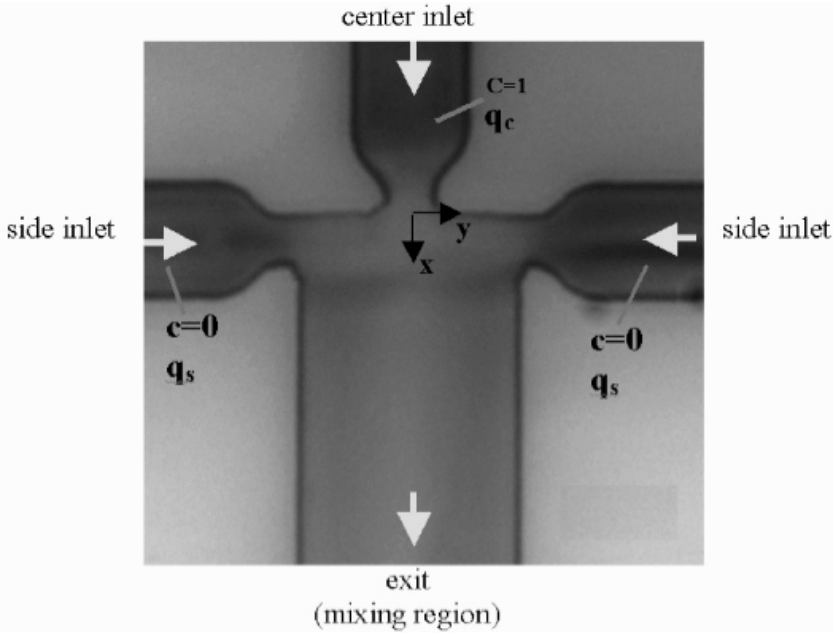


Figure 4.2: Typical fast-micro-mixer geometry. q_s and q_c are respectively the side/center injection velocities. c is the denaturant concentration.

4.3 Fast-Microfluidic mixer modelling

The SD2A, HGSA and GA algorithms are used to optimize the shape of a given fast-micro-mixer in order to reduce its mixing time. There are no general definitions of mixing time and several definitions of the degree of mixing exist in the literature [56]. In most cases, the best definition of mixing time is an ad hoc rule that takes into account the figures of merit of a specific application. In section 4.3.3, we propose a mixing definition for our mixer that well characterizes the temporal resolution of subsequent macromolecular folding kinetics measurements.

4.3.1 Shape design

The mixer shape considered is a typical three-inlet/single-outlet channel architecture proposed by Knight [5] (see Figure 4.2). Due to the fact that our model is symmetric we only study half of the mixer [44] (see Figure 4.3-*Left*). Our model is a 2D approximation of the physical system [57]. Experiments show a 5 % deviation from a 3D modelling which is satisfactory for a 2D model to be used as low-complexity model in optimization.

Our aim is to optimize the corner shapes. We parameterize the corner regions by cubic splines (see Figure 4.3-*Right*). The total number of parameters is 8: 4 for each corner. In addition, we account for the following constraints:

- The considered fast-micro-mixer is $22\mu m$ long and $10\mu m$ large.
- The lithography step in fabrication limits the minimum feature size to a minimum of 1 to 2 μm.

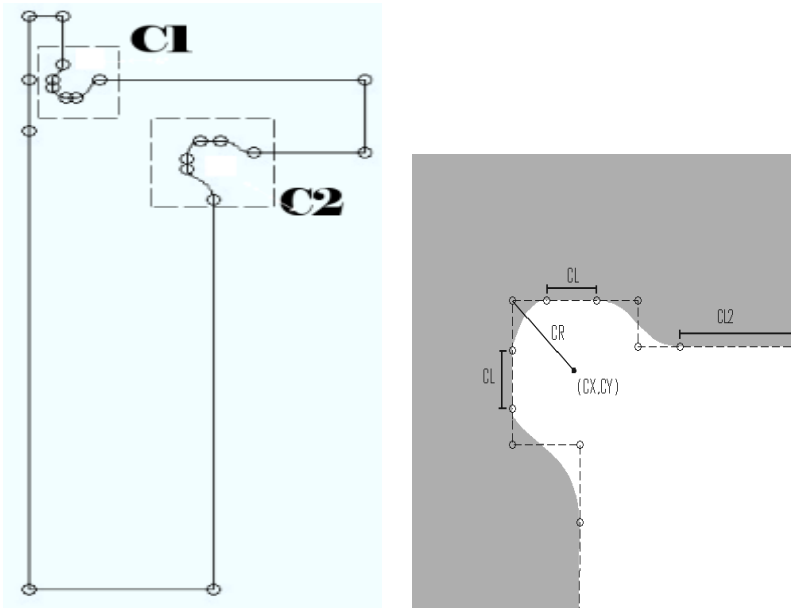


Figure 4.3: **Left**: Half-Shape parameterization. The corners are denoted by C_1 and C_2. The mixers are symmetric about the vertical centerline so only one symmetric half is modeled to save computation time. **Right**: Typical parameterization of a corner. Here C_2. We consider 4 parameters: C_x, C_r, C_l, C_{l2}. C_y is fixed.

- The width of the side channel nozzles is set to 3 μm and the width of the center channel nozzles to 2 μm to mitigate clogging issues.

- The depth of the channels is set to ~ 10 μm to optimize the fluorescence signal with a confocal system. In addition, it is difficult to etch deeper on fused silica [44].

- The physical properties of buffers and guanidine hydrochloride denaturant used here for protein folding studies have known parameters such as density, viscosity, and diffusivity.

- Finally, the maximum side velocity is $U_s = 10^{-4} m/s$. Hence, a typical flow Reynolds number based on sides channel thickness and flow inlet is $Re_w = \frac{U_s w_s}{\eta} \sim 15$.

- The device length scales required for the continuum assumption to hold are different for the flow of liquids and gases [58]. For gases, the appropriate length scale is typically the mean free path of the gas. In liquids, molecules are tightly packed and we use the characteristic size of the molecule. For water at standard conditions, the molecular diameter is $310^{-10} m$ which justify the continuum assumption for this problem as it gives a Knudsen number of order 10^{-4}.

Thus, the corresponding search space of the optimization problem is $\Omega = [x_i^{min}, x_i^{max}]_{i=1}^{8}$ where x_i^{min} (resp. x_i^{max}), the minimum (resp. maximum) value of the i^{th} parameter, are fixed by the previous constraints.

4.3.2 State equations

The mixer flow was analyzed using numerical solutions of the full Navier-Stokes fluid flow equations and a convection diffusion equation describing concentration fields c of the guanidine hydrochloride denaturant. Only steady configurations have been considered as we are not interested in the behavior of the device during its transient set up.

These flow simulations were used to explore the guanidine hydrochloride performance of a variety of mixer designs with systematically varied flow and geometric parameters. The model is applied to mixer shape designs described in 4.3.1. Mixer flow and concentration field c are computed using the following system of equations:

$$\begin{cases} -\nabla.(\eta(\nabla u + (\nabla u)^\top)) + \rho(u.\nabla)u + \nabla p = 0 \\ \nabla.u = 0 \\ \nabla.(-D\nabla c + cu) = 0 \end{cases} \tag{4.1}$$

where (u, p) is the flow velocity vector and pressure field, $\rho = 1013kg/m$ is the density, $\eta = 8.10^{-4}Pa.s$ the dynamic viscosity and $D = 2^{-9}m^2/s$ is the diffusion coefficient.

Finally, the following boundary conditions are assumed: $u = 3.210^{-4}m/s$ on side inlets, $u = 3.210^{-6}m/s$ on center inlet, $u.t = 0$ on the exit, $u.n = 0$ on the center symmetry line, $u = 0$ elsewhere on the shape border. (t, n) is the local orthonormal reference frame along the boundary. c is prescribed at inlet and normal zero gradient is assumed for all other boundaries. $c = 0$ at side inlet and $c = 1$ at center inlet.

The Incompressible Navier-Stokes equations are solved iteratively. Velocity and pressure spacial discretization is based on P2-P1 Lagrange finite elements to stabilize to realize the Ladyzhenskaya, Babuska and Brezzi (LBB) stability condition. The convective diffusion equation is solved using a streamline-upwind/Petrov-Galerkin (SUPG) method [59]. A direct damped Newton method is then used to solve the nonlinear system coming from (4.1) [60].

4.3.3 Cost Function

The cost function to minimize is the mixing time of the considered Lagrangian fluid particle travelling along the centerline. The mixing time is defined as the time required to change local concentration from 90% to 30% of the inlet value c_0:

$$J(x) = \int_{y_{90}}^{y_{30}} \frac{dy}{v} \tag{4.2}$$

Where y_{90} and y_{30} denote respectively the points along the symmetry line where the concentration is at 90% and 30% of c_0. v is the normal component of the velocity. 30% has been chosen as an average value between 10% (numerically difficult to reach) and 50% (insufficient mixing).

This modeling has been validated by a first prototyping [44].

4.4 Results and discussion

SD2A, HSGA and GA algorithms are applied with parameters fixed in chapter 2

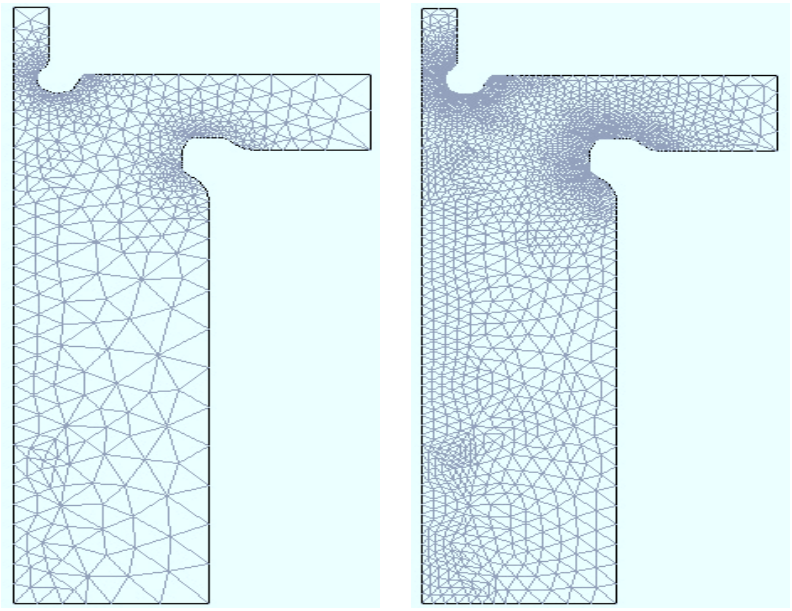


Figure 4.4: **Left**: For gradient calculation the state is partially evaluated on a coarse mesh. **Right**: On each new shape the state is calculated on a fine mesh. The meshes are finer near the intersection and along the high concentration and velocity gradient regions of the focused stream in order to resolve them.

GA and HGSA start from a random initial population X^0 including the initial mixer design [44]. SD2A starts from the initial mixer design.

For GA (resp. HGSA) the total number of functional evaluations is 5500 (resp. 2600). For SD2A the evaluation number is 3500 with more than 90 % of the evaluations performed on a coarse mesh with incomplete state evaluations (See Figure 4.4). Evaluating the gradient on the coarse mesh is equivalent to two evaluations of the cost function on the fine mesh. Furthermore, as we can notice on Figure 4.5, for each search space dimension, the coarse gradient direction is the same than the fine one. Hence, GA requires about 4 days, HGSA 2 days and SD2A less than 4 hours to reach the same minimum. Convergence histories are given in Figure 4.6. As we can see on this Figure, SD2A has visited several attraction basins before exploring the best element basin. Each evaluation on the fine (resp. coarse) level requires about one minute (resp. 20s) on a 3Ghz PC computer. In addition, with SD2A the infimum is reached sooner in the optimization.

The three optimization techniques have lead to the same optimized shape. Initial and optimized shapes are presented in Figure 4.7.

The main feature of the optimized geometry is a constriction of the exit channel after the intersection. The constriction results in high fluid velocities just after the intersection of the three inlet streams. Figure 4.9-Left shows the simulated velocity fields for both mixers. In the initial design, the velocity along the center streamline of the exit channel is relatively small as the convective diffusion problem develops. In contrast, the optimized design has a flow constriction immediately after the intersection which maintains high velocity magnitudes and high velocity gradients which approximately coincide with the

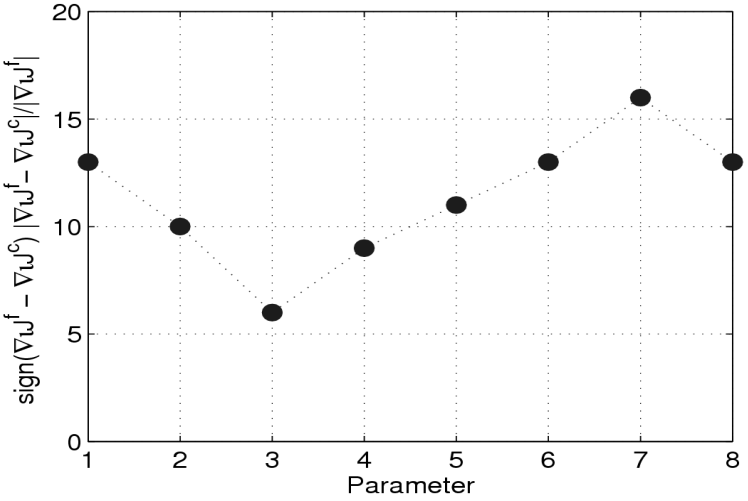


Figure 4.5: Comparison of gradients computed on the fine f and coarse c meshes. Average value of $(\text{sign}(\nabla_i J^f - \nabla_i J^c)\frac{|\nabla_i J^f - \nabla_i J^c|}{|\nabla_i J^f|})$ where i=1,..,8 denotes the parameter number. We notice that the sign is always correct.

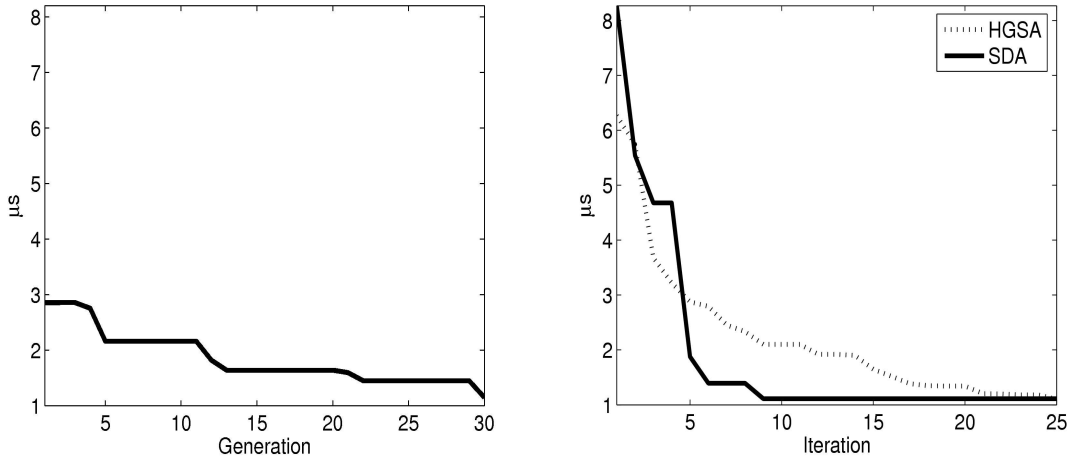


Figure 4.6: **Left**: Best element convergence history vs. generation iterations for GA. **Right**: Best element convergence history vs. iterations for SD2A and HGSA.

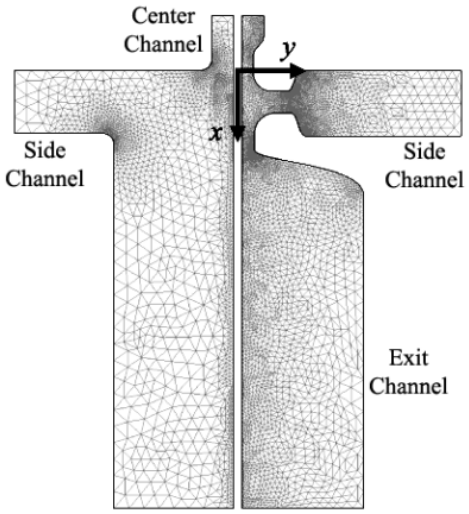


Figure 4.7: The design of Initial (**left**) and the current fully shape-optimized geometry (**right**) with mesh. The channels are labelled as center channel (top or North channel), side channels (horizontal or East/West channels), and exit channel (bottom or South channel).

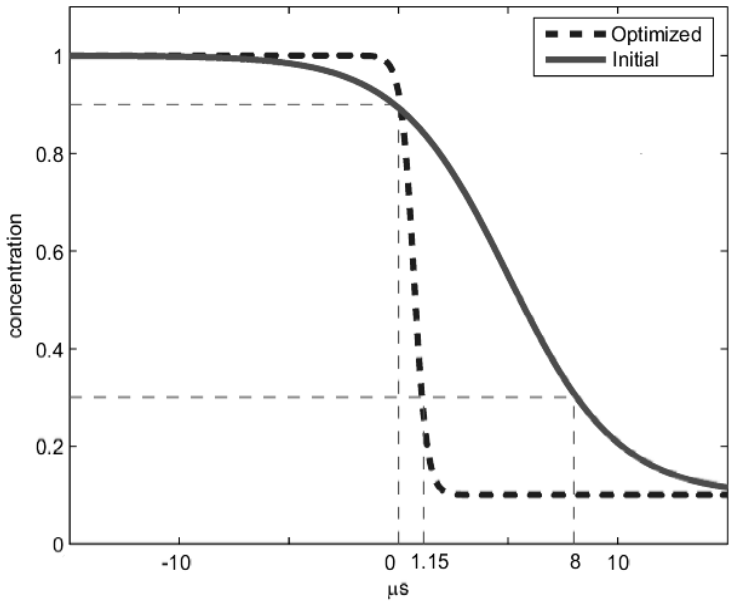


Figure 4.8: Predicted normalized concentration versus time. Dashed line is the initial mixer design and solid line is the current design. The older design takes 8 μs to go from 90 % to 30 % of the initial concentration. The current design takes only 1.2 μs.

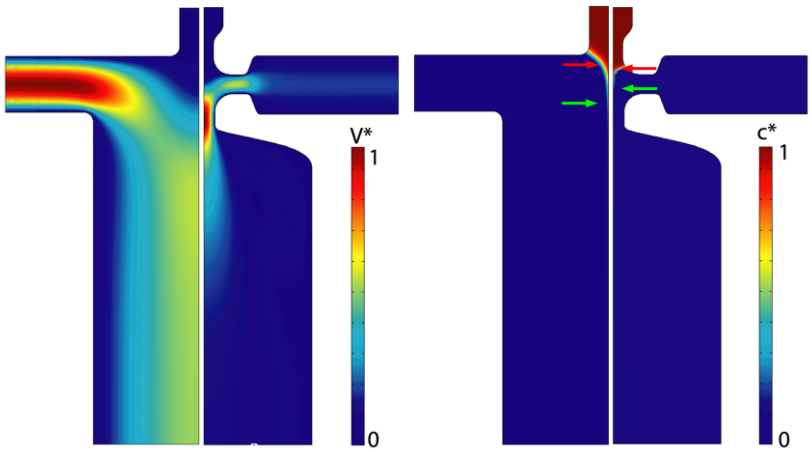


Figure 4.9: (**Left**) Velocity field for the initial design (*left*) and the optimized design (*right*). Color indicates magnitude of normalized velocities. In the initial design, the velocity colormap is normalized by Vmax=3.25 m/s. In optimized design, the velocity colormap is normalized by Vmax=17.3 m/s (**Right**) Concentration field for design of Initial (left) and the optimized design (*right*). Arrows point to the locations of c = 90% and c = 30% along the center line, respectively. Color map indicates normalized concentration.

region where the local Lagrangian concentration drops from 90% to 30%. Figure 4.9-Right shows a detailed view of the simulated concentration fields of the two mixer designs. Superposed on each of the concentration fields are arrows denoting the locations along the center line of 90% and 30% denaturant concentration, respectively.

Comparison of Figures 4.9 shows how mixing occurs in approximately the same region for the initial and optimized mixer designs, but the current mixer has approximately 10-fold higher local velocities (15 m/s versus about 2 m/s for the initial mixer). The results are summarized with center-line flow profiles shown in Figure 4.10. Figure 4.10-Left shows the concentration along the center-streamline for both mixers. The concentration drops from unity to about 0.05 in the region x = 0 to 5 mm in both cases. In Figure 4.10-Right, we show center-streamline velocity and relation between time and axial location for a Lagrangian particle. t = 0 is chosen at the point where the concentration reaches 90 % of the initial value. The $t(x)$ curves shows clearly how the new design maintains higher absolute velocity in the mixing region.

Figure 4.8 shows the Lagrangian concentration (i.e., local denaturant concentration experienced by a particle travelling along the flow centerline) versus time for both mixers. These curves are calculated by combining the concentration versus x of Figure 4.10-Left with the time versus x of Figure 4.10-Right. Again, t = 0 denotes the instance at which the concentration reaches 90% of the initial value. As per Equation (4.2), the cost function is then the time at which the concentration reaches 30% of co. The initial design has a mixing time of approximately 8 μs while the current shape optimized design reduces this to 1.2 μs. The concentration asymptotes to a final value determined by the flowrate ratio. The process is similar to a one-dimensional transient diffusion problem with a delta initial

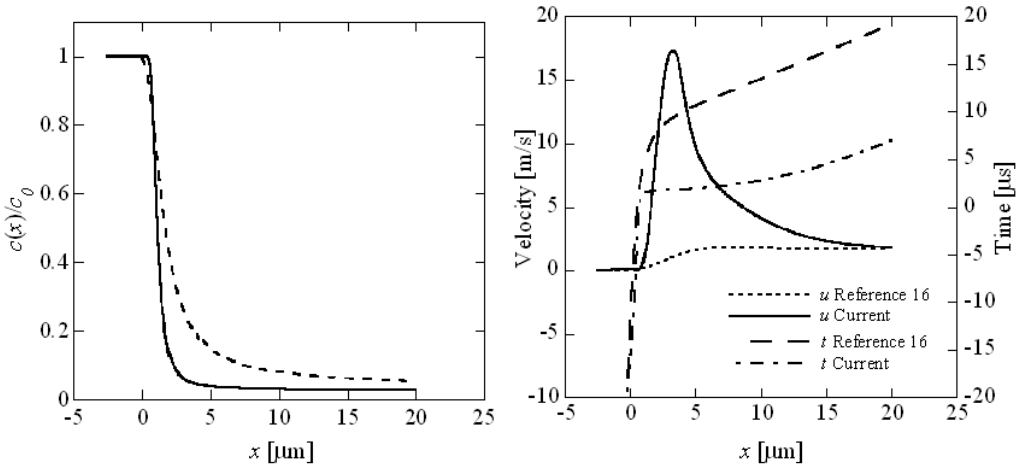


Figure 4.10: (**Left**) Normalized concentration versus distance from nozzle, from simulations. Dashed line is for initial mixer design, solid line is optimized design. (**Right**) Velocity, u, along centerline of mixer and integrated Lagrangian time, t, versus distance from nozzle. Initial mixer results are shown as dotted (u) and dashed (t) lines and current results are shown as solid (u) and dash-dot (t) lines.

condition.

Proteins travelling along streamlines other than the center streamline undergo a different Lagrangian concentration history. They experience different velocity and concentration fields and therefore have different mixing times. A detection volume (larger in characteristic length than the stream but much smaller than the outlet channel dimensions) along the focused stream at some $x = x_{detection}$, will sample proteins with a distribution of mixing times. We can quantify this non-uniformity of mixing times by calculating the mixing time of all the streamlines across the focused stream. We again use the mixing time definition of Equation (4.2), but with u(x) replaced by u_{SL}(s) where u_{SL} represents the local fluid velocity along the streamline of interest and s represents the contour distance along the streamline.

Figure 4.11 shows the variation of mixing times across the focused stream for both designs. We used a custom confocal microscope setup with a depth of field of approximately dz = 1.5 μm and centered at z = 0, so that we are only interested in streamlines in the z = 0 midplane of the mixer. The cross-stream coordinate, y, is nondimensionalized by the width of the focused stream, w_{fs}. The latter dimension is defined as the distance between the two streamlines separating the center and side channel flows at $x_{detection}$. $x_{detection}$ is taken at x = 5 μm (downstream of the nozzle exit) for this mixing uniformity analysis, at a location where the center stream velocity is uniform (less than 1 % variation along the center stream from y = -0.1 to 0.1 μm). The term "nozzles" refer to the narrow channel sections near the intersection of the center and side channels. Streamlines originating upstream of the center nozzle and flowing near the outside wall of the center channel (these become the periphery of the focused stream) have longer mixing times than the center streamline. At the wall, the velocity is zero, so we define here the outer streamlines as

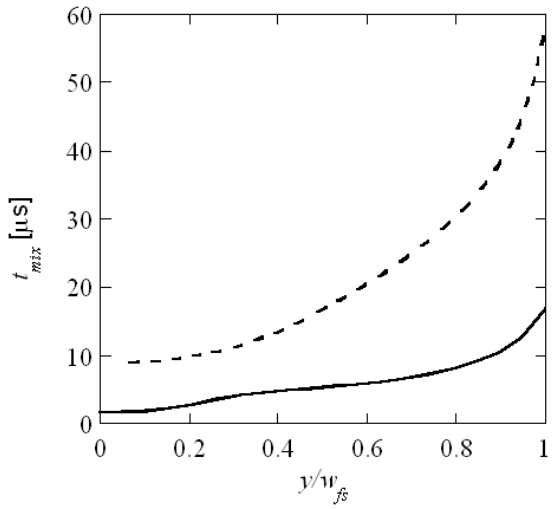


Figure 4.11: Predicted variations in mixing time for different streamlines in focused stream. Dashed line is the initial mixer design and the solid line is the current design. Abscissa is y coordinate at $x = -5\mu m$, normalized by width of focused stream, w_{fs}. The current design demonstrates shorter and more uniform mixing times across the focused stream.

those within 0.5 μm of the wall in the x < 0 μm region above the nozzle. These outer streamlines have a mixing time of 58 and 17 μs for the initial and optimized designs, respectively. The average mixing time across the focused stream is 24.5 μs for the older design and 6.1 μs for the new design. The standard deviation of mixing times for each design is 14.7 and 3.9 μs for the initial and optimized mixers, respectively.

These results show a significant improvement in mixing time uniformity for the current optimized design over the older design. They have been obtained numerically considering the model presented in previous section, we will now validate and compare them to those obtained by an experimental implementation performed at the Mechanical Engineering department of the Stanford University.

4.5 Experimental Validation

We measured the mixing times of the two designs with dye-quenching experiments in silicon-glass devices to validate the numerical simulations and optimization analysis. This experimental implementation was performed at the Mechanical Engineering department of the Stanford University under the direction of David E. Hertzog and Juan G. Santiago.

4.5.1 Experimental Techniques

Device Fabrication The microfluidic mixers were fabricated on silicon substrates using contact photolithography, Deep Reactive Ion Etching (DRIE) and the channels sealed

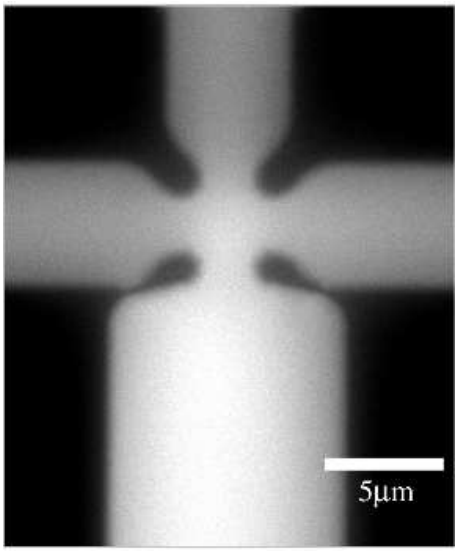


Figure 4.12: Brightfield reflection images of the fabricated shape-optimized mixer. Focal plane is approximately coincident with the inner surface of the optical access window for each channel.

by anodic bonding of a 170 μm-thick Pyrex 7740 wafer. Figure 4.12 shows brightfield-reflection CCD camera images of the mixing regions for the shape-optimized design. The channels are 10 μm deep with vertical side walls. The lithography step limits feature sizes to approximately 2 μm which imposes a physical constraint on the optimization problem. The devices have built in filters (arrays of vertical posts with 2 μm spacing) upstream of the mixing regions to reduce clogging. The channels leading to and away from the mixing regions shown in Figure 4.12 are identical to those used in our previous mixer [44].

Pressure Control and Buffer/Reagents Solutions were pumped through the mixers with the same computer-controlled pneumatic pressure system we used for our first mixer. The pressure system consists of two 0 - 100 psi regulators with 0.1 % accuracy controlled by an analog output board. A custom designed acrylic jig holds the mixer and connects it via o-rings to fluid reservoirs where the pneumatic lines interface to create a pressure head. The two regulators allow precise and independent control of flowrates in the center and side channels. We used a dye solution of 2 MDa dextran-conjugated fluorescein at 10 mM, diluted in 100 mM Phosphate Buffered Saline (PBS) for the dye-quenching experiments. This dye solution was mixed with either 500 mM potassium iodide prepared in 100 mM PBS for quenching, or pure buffer (100 mM PBS) for unquenched mixing. All solutions were filtered with 0.1 μm syringe driven filters prior to use.

Experimental Setup We obtained two image scans with exposure times of 3.9 μs per pixel for each experiment to obtain the intensity ratio for dye-quenching: one image with a dye center stream mixing into buffered salt solution sheath streams (quenched intensity, $I_{quenched}$) and one image with the dye mixing into buffer solutions (unquenched

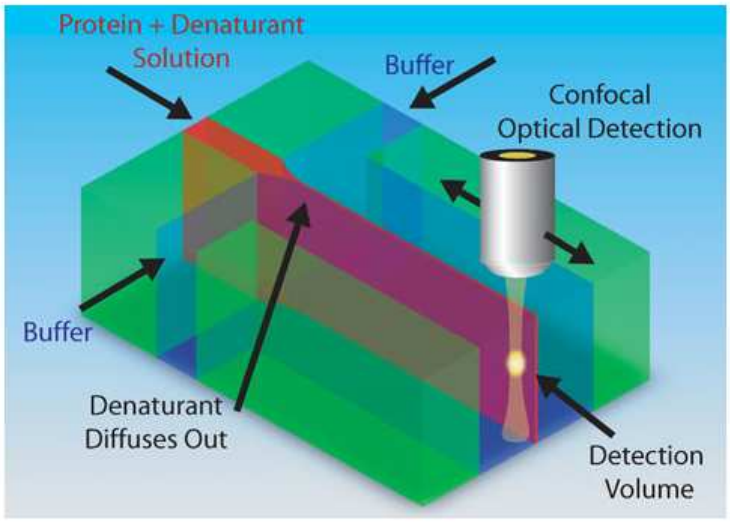


Figure 4.13: Confocal microscope: The fast micro-mixing devices designed consist of channels etched into silicon chips equipped with a confocal optical detector. The protein enters with a denaturant, which relaxes the protein strands. Travelling down the length-wise channel, the denaturant diffuses into a buffer during travel, and the proteins fold in view of the optical detector.

intensity, $I_{unquenched}$). The intensity ratio is obtained from these two images, RI = Iquenched/Iunquenched. Registration of the two images is achieved by aligning notches etched into the side walls of the center inlet 10 μm upstream of the intersection (not shown).

Confocal Optical System Experiments were performed on a custom-built confocal microscope (See Figure 4.13). An argon-ion laser provided excitation at 488 nm through a 488 nm notch filter and a 100X 1.4 NA objective. Fluorescence emission was collected with the same objective, passed through a dichroic mirror, 50 μm pinhole (Newport Co), a 500 nm long pass filter, and detected with an avalanche photodiode. Images were integrated by scanning with a three-axis piezo-scanning stage controlled with a commercial scanning controller and software. Typical x-y scans were 7.5 μm wide and 60 μm long with 0.1 μm resolution and taken at the vertical midpoint of the channels where the flow is approximately two-dimensional.

Fluorescence Quenching Typical mixer characterization techniques to determine mixing times use the quenching of a slowly diffusing fluorescent dye mixed with a buffer containing a fast-diffusing quencher [6]. This approach decouples the kinetics associated with the diffusion of the dye from those of the quencher. To compare the mixing times of our original and shape-optimized designs we quench a high molecular weight fluorescent dye (10 mM fluorescein conjugated with dextran) with iodide ions from a 500 mM potassium iodide solution. Iodide ions reduce the quantum yield of the dye molecules through

collisional quenching, and the resulting decrease in intensity is used to measure the local concentration of iodide.[44]

The ratio of intensities of quenched dye solutions to unquenched solutions (RI) were calibrated with salt concentration using a spectrofluorometer in a preliminary calibration study in which reactants were in chemical equilibrium. The empirical transform between RI and salt concentration is shown in Figure S-1. This curve was obtained from equilibrium measurements using the spectrofluorometer. At concentrations near 500 mM, dye intensity is reduced to 25%.

4.5.2 Experimental results

Confocal Images Figure 4.15 shows typical scanned confocal images from dye-quenching experiments in the initial (a and b) and current (c and d) mixer designs. As expected, the fluorescence intensity of the unquenched streams (a, c) are higher than that of the quenched streams (b, d). Note that the velocity field is the same between the quenched and unquenched images of a given geometry, even though the intensity field is different. The concentration of dye molecules is also the same between the quenched and unquenched cases. Fluorescence intensity is, of course, reduced in the quenched experiments due to the presence of iodide ions. The shape optimized mixer shows a dip in fluorescent intensity in the narrow "neck" region for both the unquenched and quenched dye experiments (an explanation of this phenomenon is given by Figure 4.14). This inflection in centerline intensity is an artifact of the imaging procedure. The vertical aspect ratio for the "neck" region is high enough (AR = 1/5) such that the opaque silicon channel side walls partially block the paths of light of excitation and emission. This artifact, however, does not affect our quantification of quenching rate as our measurements are based on unquenched-to-quenched intensity ratios, and not on the absolute values of intensity.

Both mixer designs showed typical hydrodynamic focusing of the dye solution into a thin stream. The shape-optimized design improved mixing time by increasing velocity gradients in the region of highest concentration gradients. The higher velocity gradients in the focusing region of the optimized design more quickly stretch the interfaces between the center stream and the sheath streams. This reduces the time Lagrangian particles reside in the two-dimensional convective diffusion region of the early phase mixing, and more quickly achieves the "focused stream" state where mixing time is limited mostly by spanwise (y-direction) diffusion.

Data Analysis and Calibration We normalized images by subtracting background images of unseeded buffer flow, and the quenched and unquenched scans were aligned in x, y and q (rotation). We extracted intensity versus x profiles by averaging $\Delta y = 0.5\mu m$ wide strips at $\Delta x = 0.1\mu m$ spacing along the focused stream. The ratio of intensity of the quenched image to unquenched image, $RI = I_{quenched}/I_{unquenched}$, is shown in Figure 8a for both mixer designs. For x < 0, the dye and salt solution are not yet in contact and there is no quenching. After the two streams intersect, iodide ions begin to diffuse into the dye stream and RI rapidly decreases to 35 % of its initial value at $x = 5\mu m$ (RI drops to 25 % at $x = 25\mu m$, as expected from the equilibrium measurements). We used simple linear interpolations of the transform data to convert the raw RI data of Figure

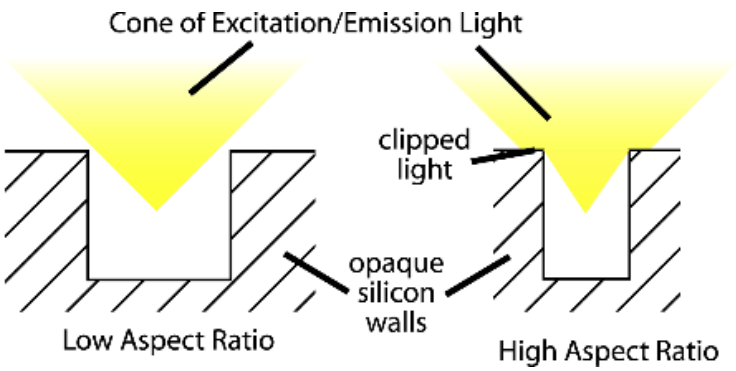


Figure 4.14: On the left side we have a low aspect ratio channel (showing cross section with no coverslip for clarity). The silicon side walls (hatched) do not interfere with the cone of light (yellow) when focused in the vertical center of the channel. This is the case when the channel is wide such as in the exit channel of the mixers. On the right we have the case of a high aspect ratio channel (such as at the center nozzle where the channel is narrow). Here, part of the excitation and emission light is blocked by the opaque silicon side walls. This causes the total intensity of the fluorescent dye in the narrow parts of the mixer to look dimmer than in wide channels. Because our measurement signal is a ratio, however, ($I_{quenched}$ / $I_{unquenched}$) the reduced total intensity has no effect because both the quenched and unquenched signals would be reduced equivalently.

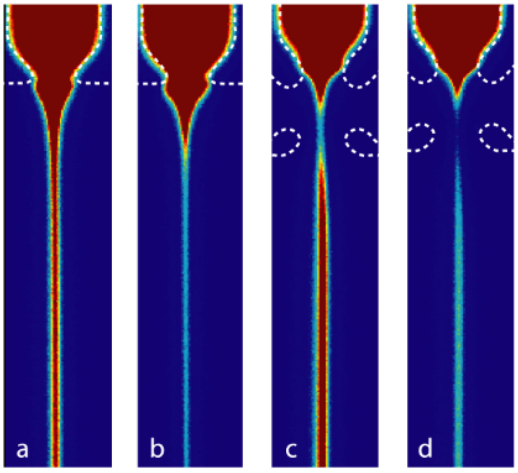


Figure 4.15: Confocal images from dye quenching experiments. Flow is from top to bottom and false colormap indicates fluorescence intensity (red is high intensity from fluorescein, blue is low intensity). Dashed lines show approximate wall locations. Scans are approximately 4 μm wide and 40 μm tall. (**a**) Mixer design of initial with buffer only in side channels, no quenching is present. (**b**) Initial design with 500 mM potassium iodide in side channels, fluorescein is quenched. (**c**) Current mixer design with buffer in side channels, and (**d**) current design with potassium iodide in side channels.

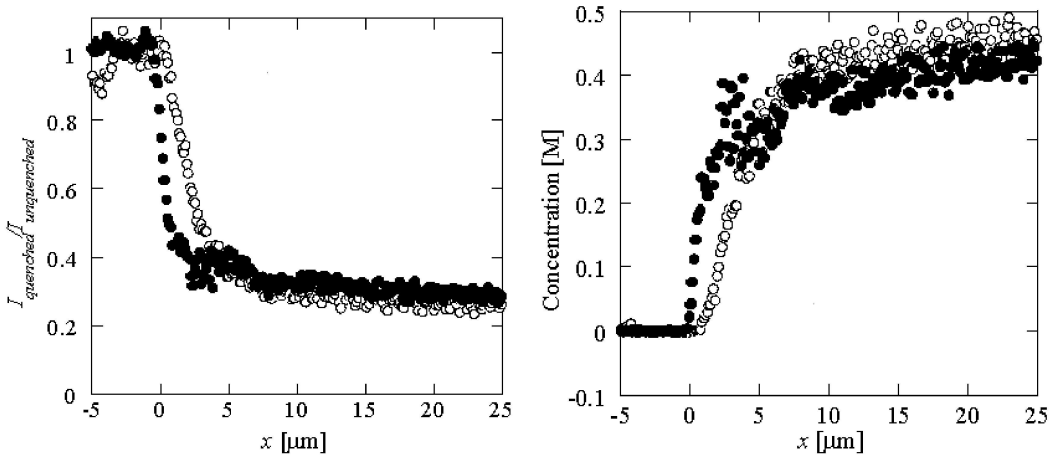


Figure 4.16: (a) Intensity ratio versus distance from nozzle from experiments shown in Figure 4.15. Intensity ratio = Iquenched/Iunquenched. Open circles are data from the initial mixer design, closed circles are data from the new design. (b) Concentration of potassium iodide versus distance from nozzle (same experiment as Figure 4.15). Open circles are initial design, closed circles are the optimized design.

4.16-a to the concentration data of Figure 4.16-b. We used the same calibration method for all experiments.

Lagrangian Time History Next, we wish to estimate the concentration versus time history that a Lagrangian particle would experience given our measurements of the Eulerian concentration field and our predictions of Eulerian velocity field. The Eulerian frame of reference is static with respect to the device. In the Eulerian description, a steady-state velocity field is used which defines fluid velocity at all points within the fluid (this vector field is thus a function only of space). In a Lagrangian frame of reference, we describe the velocity vector of an identifiable fluid particle as it travels through space (this vector quantity is thus only a function of time). In the latter frame, fluid velocities and concentrations change in time as the fluid element and frame of reference moves through gradients in the Eulerian fields. The numerical models solve for the Eulerian velocity and concentration fields, but it is more appropriate to describe the mixer performance in terms of the conditions that a protein would experience as it flows through the mixer (a Lagrangian frame of reference). To this end, we first integrate the function $1/u(x)$ along the focused stream centerline using the numerical velocity field prediction (see the form of Equation (4.2)). This conversion from the spatial coordinate, x, to a Lagrangian time coordinate, t was given as Figure 4.10-b. We can then map the parameterized Lagrangian time function t(x) onto the streamwise distance for the measured potassium iodide (KI) concentration field (which was determined directly from RI with equilibrium measurements from the spectrofluorometer, Figure 4.18). Figure 4.17 shows the concentration of KI versus (Lagrangian particle) time for both the original mixer design (circles) and the shape-optimized design (triangles). The shape optimization reduces mixing time from 7 μs in the initial design to about 4 μs in the new design. As we discuss below, the

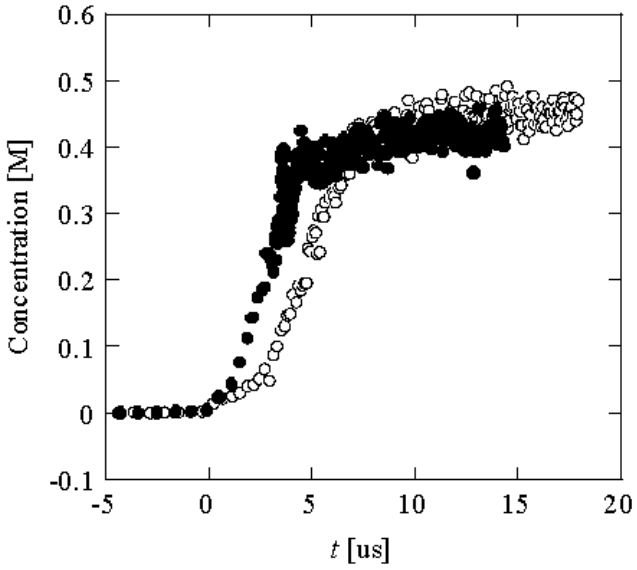


Figure 4.17: Potassium iodide concentration versus time. The current mixer optimization improved the mixing time by approximately 3 μs over the initial design. Open circles correspond to initial design and closed circles to optimized design.

performance of the new mixer matches predictions within our experimental uncertainty.

4.6 Uncertainty

The discrepancies between the simulated and the measured mixing times for the initial and optimized mixers (predicted 8 μs versus the measured 7 μs for the old, and predicted 1.2 μs versus the measured 4 μs for the optimized) may result from several sources of uncertainty:

- The fabricated geometry was different than the geometry suggested by the optimization.
- There are uncertainties in the applied pressures and flowrates.
- The finite length of the diffraction spot resulting from the confocal imaging limits measurement resolution.
- Image noise in the detection system.
- Errors associated with the conversion from quenching ratio to salt concentration.
- The two-dimensional numerical models are only approximations to the full three-dimensional flow fields.

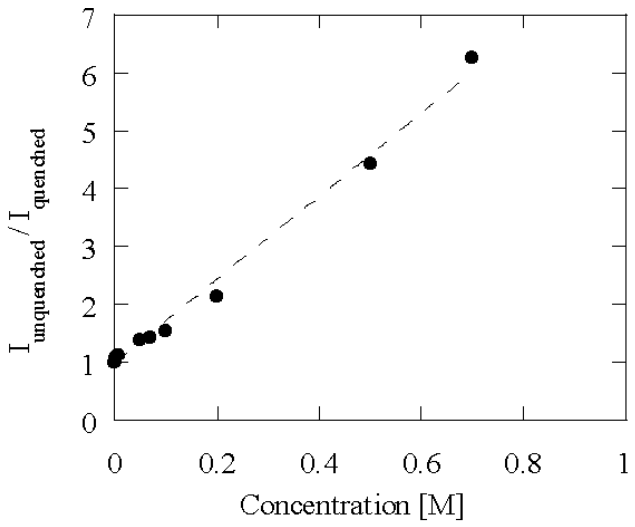


Figure 4.18: Measured intensity ratio versus potassium iodide concentration of dextran conjugated fluorescein. Intensity ratio is $I_{unquenched}/I_{quenched}$. Dots indicate intensity ratio measured with a spectrofluorometer. Dashed line is fit to Stern-Volmer equation $I_{unquenched}/I_{quenched} = 1 + KSV * c_{salt}$. The Stern-Volmer constant is $K_{-SV} = 7.17 \pm 0.16 M^{-1}$. The variation in intensity is used to calculate salt concentrations in quenching experiments.

4.6.1 Numerical Uncertainty

We used our simulations and estimates of the performance of our imaging system to explore the relative importance of each of these factors. Table 4.1 lists the sensitivity of the modeled mixing time to various channel dimensions and flowrates. By analyzing microscopy images such as those shown in Figure 4.13, we found feature sizes and feature locations of the fabricated geometry were within about $\pm$ 0.5 μm of the modeled geometry. Typically the fabricated device had slightly larger channel and nozzle widths than the model. These differences are mostly a result of the lithography step of fabrication. As shown in Table 4.1, the mixing time is most sensitive to the width of the center inlet channel. A $\pm$ 0.5 μm change in center channel width increases mixing time by a factor of about 4 (a $\pm$ 1 μm error results in a mixing time increase of about 6 - 9 fold). This sensitivity implies that the focusing region (where the center channel enters the intersection to the point downstream where the focused stream width stops changing) largely determines the mixing time. Narrower center nozzles (with appropriate changes in the flowrate ratio) can achieve shorter mixing times, however, clogging issues and the photolithography process limit how small we can make the center nozzles.

We estimate our flow control system supplies source pressures with 0.1 % full-scale accuracy. As shown in Table 4.1, even changes in flowrates of up to $\pm$10 % do not change the mixing time by more than 100 %, so that significant deviations in measured mixing time due to applied pressure variations are unlikely. We also estimated the sensitivity of flow rates to applied pressures using analytical models for flow in the supply channels. For these calculations, we used channel widths measured from optical and scanning electron

Parameter	t_{mix} [μs]	Percent Change
Flowrate:		
Center Channel +10%	0.8	-33
Center Channel -10%	2.1	+75
Side Channel +10%	1.2	0
Side Channel -10%	0.8	-33
Ratio Center/Side +25%	1.2	0
Ratio Center/Side -25%	2.3	+92
Geometry:		
Exit Channel Width +1μm	1.2	0
Exit Channel Width -1μm	1.2	0
Side Channel Width +1μm	1.0	-17
Side Channel Width -1μm	1.1	-8
Center Channel Width +1μm	12.2	+900
Center Channel Width -1 μm	8.3	+600

Table 4.1: Numerical Parameter Sensitivities. (Note: decreases in mixing time from the optimum found in this paper result from violations of the constraints imposed on the current problem)

microscope (SEM) images. We measured channel depths with a profilometer prior to bonding the glass coverslip (we measured depth variations of approximately $\pm 0.5\ \mu m$ from the etching process). The uncertainty in channel dimensions results in an uncertainty in the analytical flowrate calculations of about ± 9 %, and an uncertainty in the flowrate ratio of about ± 0.5 %. Again, such flow rate ratio errors result in mixing time changes of less than 100 %.

4.6.2 Experimental Uncertainty

Noise in the detection system (due to the photodiode, counting hardware, stray light, etc) and the conversion from intensity ratio to salt concentration will introduce uncertainty in the mixing time measurement. We estimated this uncertainty by measuring the standard deviation of the time data (see Figure 4.18) as the concentration reaches a steady value (chosen as 0.4 M here). The optimized mixer has a mixing time standard deviation of 0.28 μs as the concentration reaches 0.4 M, and so we conclude that the effects of random measurement noise on (the time averaged) mixing time are negligible. Our confocal detection system uses diffraction-limited optics with a confocal spot size estimated at 0.5 μm. Scanned images of dye intensity are therefore essentially low-pass filtered via convolution with a Gaussian kernel. As shown in Figure 4.10-a, we expected high concentration gradients in the region within about 2 μm of the nozzle exit, partic-

ularly in the case of the optimized mixer. The finite spatial resolution of the confocal detection system will tend to smooth these sharp gradients resulting in artificially longer measured mixing times. We can estimate the error in measured mixing time due the confocal spot size by approximating the imaging as a convolution between the simulated concentration field with a 0.5 μm wide Gaussian kernel. This optical convolution alone increases the estimate of mixing time from 1.2 μs to 6.1 μs.

4.7 Conclusions

We have applied various shape optimization methods, based on SD2A, GA and HSGA, to the design of fast microfluidic mixers in order to reduce mixing time of a given protein solution and solvent. The optimization reduced the expected mixing time from 8 μs in the previous mixer design to 1.2 μs in the shape-optimized design. This mixing time improvement was achieved solely by varying geometry and flowrates of the mixer within the minimum feature size and maximum flowrate constraints.

Both mixers were fabricated. We measured experimentally a reduction in mixing time from 7 μs in the initial mixer design to 4 μs in the current shape-optimized design. The measured mixing times fall within the estimated experimental uncertainty (a few microseconds) of the predicted mixing time.

One of the major barriers to reducing the mixing time further is the appearance of three-dimensional flows at higher Reynolds number which make the flow field difficult to model. Dean's vortices begin to develop in the mixing region at higher flowrates. A next step is to solve this problem by building full three-dimensional numerical models as well as developing new geometries that produce less curvature in the velocity field. Note that 2.3X and 3.9X reductions in mixing time are possible by enabling an increase of the flowrates by 2X and 10X, respectively.

Mixers presented during this chapter are currently used to measure the folding kinetics of a benchmark proteins using Forster Resonance Energy Transfer (FRET), and are currently using them to study protein collapse of a large variety of proteins [6].

Chapter 5

Portfolio Optimization Under Constraints

5.1 Glossary

In order to allow readers unfamiliar with financial vocabulary to understand the following chapter, we list here the principal definitions used in next sections (indicated using bold characters when it's the first time they appear) [61]:

- **Asset**: A resource having economic value that an individual, corporation or country owns or controls with the expectation that it will provide future benefit.
- **Asset-Backed Security (ABS)**: A financial security backed by a loan, lease or receivables against assets other type of backed securities. As an investor, asset-backed securities are an alternative to investing in corporate debt.
- **Barrier Option:** A type of option whose payoff depends on whether or not the underlying asset has reached or exceeded a predetermined price.
- **Base Point (b.p.)**: A percent of percent.
- **Bond**: A loan entity such as a government or company to an individual. In exchange for the loan the entity pays an interest rate. Bonds are valued by how much interest they pay and how stable the issuer is.
- **Capital**: Financial assets or the financial value of assets such as cash.
- **Counterparty**: Refers to the party with whom a swap or options deal has been struck.
- **Credit**: A contractual agreement in which a borrower receives something of value now, with the agreement to repay the lender at some date in the future.
- **Credit derivative**: Privately held negotiable bilateral contracts that allow users to manage their exposure to credit risk. Credit derivatives are financial assets like forward contracts, swaps, and options for which the price is driven by the credit risk of economic agents (private investors or governments)

- **Credit event**: Defined by ISDA (International Swaps and Derivatives Association). It's one of the following counterparty event:
  - Bankruptcy
  - Obligation Acceleration
  - Obligation Default
  - Failure to Pay
  - Repudiation/Moratorium
  - Restructuring
- **Default**: The risk that companies or individuals will be unable to pay the contractual interest or principal on their debt obligations.
- **Equity**: Stock or any other security representing an ownership interest.
- **Facility**: In banking parlance, means the arrangement with a borrower to provide a particular kind of credit support to the borrower.
- **Liability**: A company's legal debts or obligations that arise during the course of business operations. These are settled over time through the transfer of economic benefits including money, goods or services
- **Liquidity**: The degree to which an asset or security can be bought or sold in the market without affecting the asset's price.
- **Loan**: Borrowed funds that have a fixed interest rate.
- **Loss**: The difference between the revenue received from the sale of an output and the opportunity cost of the inputs used.
- **Maturity or Horizon**: The length of time until the principal amount of a bond must be repaid.
- **Migration**: Credit rating/quality migration describes the possibility that a firm or obligor (An entity that has an obligation to pay all principal and interest payments on a debt) with some credit rating today may move to a range of possible other ratings/qualities by the risk horizon.
- **Nominal value**: The stated value of an issued security that remains fixed, as opposed to its market value, which fluctuates.
- **Obligation**: The legal responsibility to meet the terms of a contract.
- **Portfolio**: The group of assets - such as stocks, bonds and mutuals - held by an investor.

Moody's	S&P	Grade	Risk
Aaa	AAA	Investment	Highest Quality
Aa	AA	Investment	High Quality
A	A	Investment	Strong
Baa	BBB	Investment	Medium Grade
Ba, B	BB, B	Junk	Speculative
Caa/Ca/C	CCC/CC/C	Junk	Highly Speculative
C	D	Junk	In Default

Table 5.1: Moody's and S&P rating

- **Position**: (**Long**) The buying of a security such as a stock, commodity or currency, with the expectation that the asset will rise in value. (**Short**) The selling of a borrowed security, commodity or currency, with the expectation that the asset will fall in value.
- **Pricing**: The evaluation of something in terms of its price, usually based on market demand and competition.
- **Profit and Loss Statement (P&L):** A financial statement that summarizes the revenues, costs and expenses incurred during a specific period of time - usually a fiscal quarter or year. These records provide information that shows the ability of a company to generate profit by increasing revenue and reducing costs.
- **Rating**: An evaluation of a bond's relative safety from an investment standpoint. Basically, it scrutinizes the issuer's ability to repay principal and make interest payments. Generally there are two major rating services: **Moody**'s and **Standard and Poor**'s (**S&P**) described in table 5.1.
- **Recovery rate:** The amount that a creditor would receive in final satisfaction of the claims on a defaulted credit.
- **Return**: In stocks and bonds, the amount of money required to investors on their investments.
- **Risk**: The chance that an investment's actual return will be different than expected. This includes the possibility of losing some or all of the original investment. It is usually measured using the historical returns or average returns for a specific investment
- **Return On Prudential Equity (ROPE):** It's an asset profitability indicator between the **P&L** and the regulatory capital (here 6%). Basically it can be written as:

$$ROPE = \frac{\text{Asset's income}}{6\%} \tag{5.1}$$

- **Securitization**: The process of creating a financial instrument by combining other financial assets and then marketing them to investors.

- **Security**: An instrument representing ownership (stocks), a bond, or the right to ownership (derivatives).
- **Security interest**: The right of the creditor to take all or part of a property offered as security.
- **Spread**: The difference between the price paid for a bond (the bid) and the price sold to investors (the offering price).
- **Tranches**: A piece, portion or slice of a deal or structured financing. This portion is one of several related securities that are offered at the same time but have different risks, rewards and/or maturities.
- **Structured Finance/Product**: A generic term for financial instruments which are devised for funding on the basis of identifiable assets rather than the credit standing of the entity concerned. Includes securitization, but besides, also includes various forms of lending where the cashflows of the entity are trapped at source to pay off the lender.
- **Volatility**: A statistical measure of the tendency of a market or security to rise or fall sharply within a period of time. It's computed using variance.
- **Yield (rate of return)**: The rate of income generated from a stock in the form of dividends, or the effective rate of interest paid on a bond, calculated by the coupon rate divided by the bond's market price. Furthermore, for any investment, yield is the annual rate of return expressed as a percentage.

5.2 Introduction

Due to the continuous development of **derivative credit products** [62], risk management has became an important activity in **asset** allocation of financial structures. Basically, credit risk is the risk of a trading partners, called **counterparties**, not fulfilling their **obligations** on the due date or at any time thereafter resulting into **losses** for investors. This credit risk can be generated by many factors such as:

- Variation of counterparty's **rating**: In 2000, a general downgrade of telecommunication industries'rating occurs due to an indebtedness of this sector.
- A **credit event**: Financial fraud and bankruptcy of Enron (2001) and WorldCom (2002).

Objectives of credit risk management are multiple:

- Provide risk models and evaluation tools. One of the most important mathematical contribution in this field was the development of particular risk measures, such as Value at Risk [63].
- Apply those models to evaluate risk on financial products and intend to control it.

The main difficulty is that credit losses are characterized by a large likelihood of small earnings, coupled with a small chance of losing a consequent amount of the investment. Thus the loss distribution are heavily skewed and functions corresponding to risk measure are highly non-linear. In literature, many works deal with the convexification of those functionals, resulting on an over-estimation of the risk, and apply those methods to simple portfolio examples [64, 65].

In this chapter, we focus on the application of our HSGA method to the improvement under constraints of portfolio performances, in particular non-convex risk measures and profitability. The portfolio considered here belongs to a complex family of credit portfolio, called Collateralized Loan Obligations^2 (CLO^2), owned by the BNP-Paribas Portfolio Management team.

In Section 3, we introduce the portfolio's structure considered here. In Section 4, we describe the model used to evaluate it. In section 5, we present the considered optimization problems and analyze the results obtained with our algorithm.

5.3 Collateralized Loan Obligations^2's structure

We present here the general structure of the portfolio to be optimized. In fact we consider here a portfolio of portfolios (i.e. a portfolio (called **master**) compound by parts of other portfolios (called **inner**)). Thus we will first introduce the inner portfolio's structure and then the master portfolio's one.

5.3.1 Inner Portfolios - CLO

Collateralized Loan Obligations (CLO) are **security interests** in pools of assets (called also Collateral), that usually comprise **loan**.

The objective, for financial institutions, is to buy **securitization** in order to protect themselves from eventual **defaults** of counterparties included in the CLO. Investors bear the credit risk of the collateral but in counterpart receive, until CLO **maturity** date, a periodic remuneration proportional to the risk.

Multiple **tranches** of **securities** are issued by the CLO, offering investors various credit risk characteristics. Tranches are categorized, according to their degree of credit risk, as following:

- **Senior**: Low credit risk. It's a security that only cover high loss events. The spread paid to investors is the lower.
- **Mezzanine**: Medium credit risk. Those securities cover intermediate loss events.
- **Junior**: High credit risk. It's a security that cover first losses. The tranche spread is the higher. Generally, due to its high spread, financial institutions don't buy securitization on this tranche.

More precisely, if there is a default, investors on junior tranche first cover the loss. In cases where the loss is superior to the first tranche **nominal**, other investors, successively from mezzanine to senior, cover the remain losses.

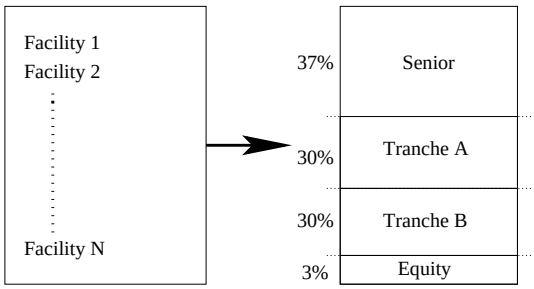


Figure 5.1: Example of CLO's structure. It's compound by N facilities and sliced in 4 tranches.

CLO's structure example : In order to illustrate previous definition we consider the following CLO's structure example, with a maturity of one year and compound by N facilities and sliced in four tranches: We consider a CLO,

- 3% of the CLO's nominal in the junior tranche. Investors of this tranche receive 600 **b.p.** of the tranche's nominal each month.
- 30% of the CLO's nominal in the first mezzanine tranche. Investors of this tranche receive 50 b.p. of the tranche's nominal each month.
- 30% of the CLO's nominal in the second mezzanine tranche. Investors of this tranche receive 20 b.p. of the tranche's nominal each month.
- 37% of the CLO's nominal in the senior tranche. Investors of this tranche receive 5 b.p. of the tranche's nominal each month.

This structure is depicted by Figure 5.1. In case of defaults, we have four different scenarios depending on the loss amount:

- **Scenario 1**: If Loss $\leq$ 3 % of the total CLO value: Only investors who have parts in junior tranche will pay to financial institution: (3% of CLO's nominal - CLO's Loss).
- **Scenario 2**: If 3 %$<$ Loss $\leq$ 33 % of the total CLO value: investors who have parts in equity tranche will pay to financial institution: 3% of CLO's nominal, and investors who have parts in the first mezzanine will pay to financial institution: (30% of CLO's nominal - (CLO's Loss-3 %) of CLO's nominal).
- **Scenario 3**: If 33 %$<$ Loss $\leq$ 63 % of the total CLO value: investors who have parts in the equity and first mezzaninie tranches will pay, respecting their tranches, 3% and 30% of CLO's nominal. Investors who have parts in the second mezzanine tranche will loose (30% of CLO's nominal - (CLO's Loss-33 %) of CLO's nominal).
- **Scenario 4**: If 63 %$<$ Loss: investors who have parts in the senior tranche will pay (37% of CLO's nominal - (CLO's Loss-63 % of CLO's nominal)). Other investors will, depending of their tranche, pay 63% of CLO's nominal.

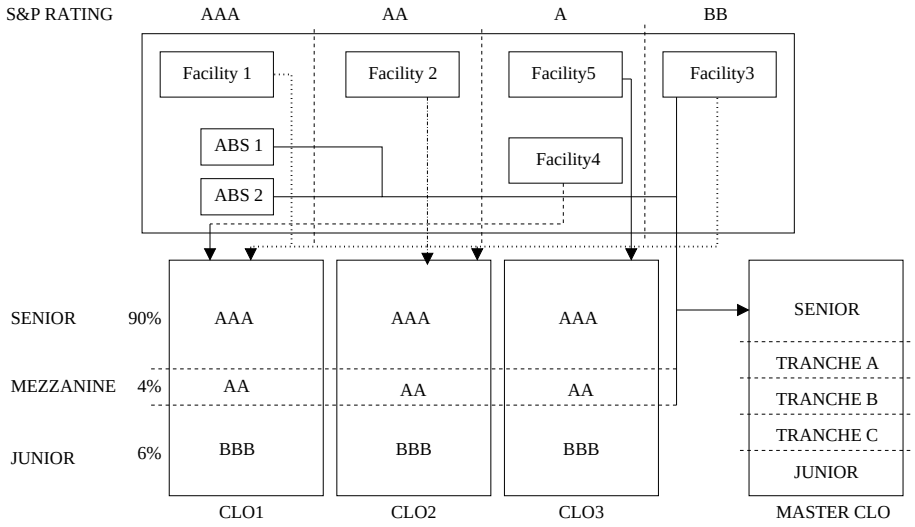


Figure 5.2: CLO^2's structure example: ICLOs are decomposed in three tranches. ICLO 1 contains facilities 1,3 and 4. ICLO 2 contains facilities 1,2 and 3. ICLO 3 contains facility 5. Master CLO is composed by the mezzanine tranches of ICLO 1,2,3 , two external ABS and one Single-Name (facility 3).

5.3.2 Master portfolio - CLO^2

During last years, the relative stability in corporate credit shifted investor interest further toward structured instruments to enhance yield, CLO^2 (CLO Square) emerged.

The CLO^2 (also called Master CLO) may be constructed on the basis of single tranche CLOs. A CLO^2 simply consists into a repackaging transaction of one or several single tranches of CLOs, called Inner CLOs (ICLO). In addition to these single tranches CLOs, a set of additional stand-alone assets, such as Single-Names (a single **Facility**) or **ABS**, can be included in the portfolio. At the end, the Master CLO is divided in sub-tranches that will be proposed to investors. The global performance of a CLO^2 is derived from the performance of the pool of single tranche CLOs. An example of CLO^2's structure is given in Figure 5.2.

Thanks to how a CLO^2 transaction is built, this type of instrument offers to investors the following attractive features :

- The investors can take exposure to a very broad and highly diversified portfolio.
- The investors benefit from two layers of subordination: a first one at the level of the ICLOs, a second one at the level of the investment tranches in the Master CLO.
- The tranches of a CLO^2 transactions are more resilient than their single tranche counterparts to a rating **migration** of the underlying credit portfolio in most stress scenarios.

On the other hand, CLO^2 clearly leaves investors exposed to more extreme systemic market conditions, in the sense that they experience no losses up to a point for about 90% of the cases, after which the loss deterioration is fairly fast and highly severe.

Analyzing CLOs is difficult. There is an entire portfolio of credits to analyze combined with the complexity of the tranching, which must be also analyzed. Sophisticated credit portfolio models should be used.

5.4 CLO^2 evaluation model

In this section, we present the methodology that allows us to evaluate a given CLO^2. Schematically, we can give the global structure of the considered method:

1- Input: collect data of each facility in the portfolio.

2- Evaluate the loss density function of the associated portfolio.

3- Compute the desired portfolio performance indicators using the loss density function and market data (Risk measures, incomes...).

We describe more precisely each step presented in previous scheme:

5.4.1 Step 1- Collect facilities' data.

We consider a portfolio compound by n facilities. Each facility is characterized by:

- Its nominal (N_i).
- Its maturity (T_i) date.
- Its spread (Sp_i).

those data are fixed by the bank during the portfolio's constitution.

In order to apply the model described in the next subsection, for each client associated to facility i, we consider:

- Its country of business (C_i).
- Its industry sector (I_i) (we will only consider 1 sector per client in the following).
- Its rating (Rat_i) (here supposed fixed).
- A Global **Recovery Rate** (GRR_i), i.e. the recoverable amount in case of default (in percent). Or equivalently, the Loss Given Default ($LGD_i = 1 - GRR_i$).
- Its *R-square* (R_i): Represents the degree of correlation between the value of a client's assets and the behavior of the global economy.
- **Trade limit (TL)**: The maximum and minimum nominal of each dependence of a company on its activity sector.

those informations are given by private institutions.

All facilities' data are stocked in a set denoted by $\boldsymbol{PORT}$.

5.4.2 Step 2- Evaluation of CLO^2's Loss density function

We present here the model used to evaluate CLO^2's Loss density function. It's a Default times model [66]. It's derived from a methodology proposed by Kealhofer, McQuown and Vasicek (KMV) [67].

Basically, the default time τ_i of a particular facility i is the time for which the associated client is in default resulting into a loss (where the time 0 corresponds to the creation date of the portfolio and the time H to the portfolio maturity). It's a random variable. The vector $T = (\tau_1, ..., \tau_n)$ is called default time vector. Each particular value of the default times vector T corresponds to a possible evolution scenario of the portfolio.

The method proposed here to evaluate Loss density function can be divided in two main steps:

- Generate values of the default time vectors using a Monte-Carlo approach.
- Compute the loss associated with each scenario and build a discrete CLO^2's Loss density function.

We can now describe each step:

Default times generation

The simulation of default times is based on the probabilistic concept of copula (See [68, 69] for more details). We give here a simplified description of the method:

The default times vector T is computed from a Gaussian vector $V = (v_1, ..., v_n)$ with zero mean and covariance matrix Σ and unit variances. Σ represent the correlation between each client in the considered portfolio.

In order to compute a such V, we first define a standard gaussian vector $G = (g_1, ..., g_n)$. V is given by taking the matrix product of the standard gaussian vector G and a transformed expression of the initial covariance matrix Σ, denoted by Ξ and defined as $\Sigma = \Xi^t\Xi$:

$$V = \Xi G = (v_1, ..., v_n) \tag{5.2}$$

Several decomposition techniques exist in order to find such a matrix. Since Σ is a symmetric, positive-definite matrix, the Cholesky decomposition is the more appropriate [70].

As it's often used, we decided to apply the KMV correlation model in order to specify the covariance matrix Σ:

The KMV model is a factor model [66]. The principal modeling lies in the correlations of default between clients. It uses a latent variable modeling, i.e. it does not model the correlations directly, but in reference to global variables that represent the global economy. Thus, 14 random variables (the factors $(G^k)_{1\leq k\leq 14}$: 2 Global Economic Risk factors, 5 Regional Risk factors and 7 Industrial Sector Risk factors) model the global economic trends. In addition, the firms i are classified by their activity sectors (61 sectors) and their geographical areas (45 areas) using random variables C_i and respectively I_i. Those factors F_i^k, C_i and I_i are supposed to be independent and normally distributed (See [67]

for detail of each random variable). Those factors G^k, C_i and I_i are supposed to be independent and normally distributed.

In this methodology, the correlation between clients i and j is given by:

$$cor(i,j) = R_i R_j \left[\sum_{k=1}^{14} \alpha_k^i \alpha_k^j + \beta^{C_i} \beta^{C_j} + \beta^{I_i} \beta^{I_j} \right] + \sqrt{1 - R_i^2} \sqrt{1 - R_j^2} \delta_i^j \tag{5.3}$$

where $\delta_i^j = 1$ if $i = j$, 0 if not. α_k^i, β^{C_i} and β^{I_i} are real coefficients that models the dependence of each client i to the factors G^k, C_i and I_i respectively and such that $\sum_{k=1}^{14} \left(\alpha_k^i\right)^2 + \left(\beta^{C_i}\right)^2 + \left(\beta^{I_i}\right)^2 = 1$.

Default times τ_i are given by:

$$\tau_i = F_i^{-1}(\Theta(v_i)) \tag{5.4}$$

where $\Theta(.)$ denotes the standard normal gaussian density function. F_i is the marginal default probability function of τ_i defined by:

$$F_i(t) = \mathbb{P}(\tau_i \leq t) \tag{5.5}$$

We are able to construct F_i thanks to the rating of the i^{th} client Rat_i (the rating gives the internal default probabilities associated with each annual horizon).

Combining this method of default times generation with a Monte-Carlo scheme we can generate a set a default time vectors.

Loss density analysis

Once the Monte-Carlo algorithm is finished, we can use all obtained default time vectors in order to compute the portfolio's loss distribution. We consider the following methodology:

We first apply to default times generated by the Monte-Carlo scheme, two filters before running the algorithm in charge of the losses allocation through the tranches. The first one only keeps safe the default times strictly smaller or equal to the maturity H of the transaction. The second one increasingly orders the remaining random variables. Thus, when we are in position to start the allocation of the losses, we are only working with the default times of interest, conveniently sorted.

For a given scenario, we first find the facility associated with a new default. Thanks to this information, we are able to know how worth is the Loss given Default (LGD) of this facility. We simply compute the loss by taking the product of the LGD (considered as stochastic or deterministic) and the facility nominal. It represents the current loss. This new current loss has to be constrained to take into account the limited liabilities property of CLOs'tranches investments. Once we have collected a new loss, we simply aggregate that current loss to the sum of the previous current losses to get the global loss

If the global loss (given a particular ICLO) is strictly larger than the amount of subordination of the ICLO tranche, then the first losses in the equity tranche of the

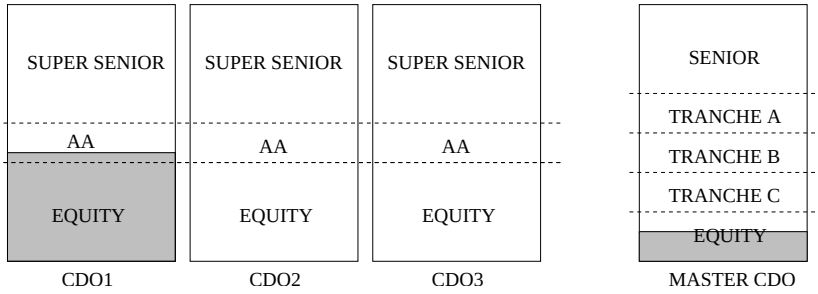


Figure 5.3: **First scenario:** We consider the portfolio presented by figure 5.2. Global loss in the inner CLO1 exceeds the subordination of the mezzanine CLO tranche. The first loss occurs in the master CLO. The equity tranche only is amortized.

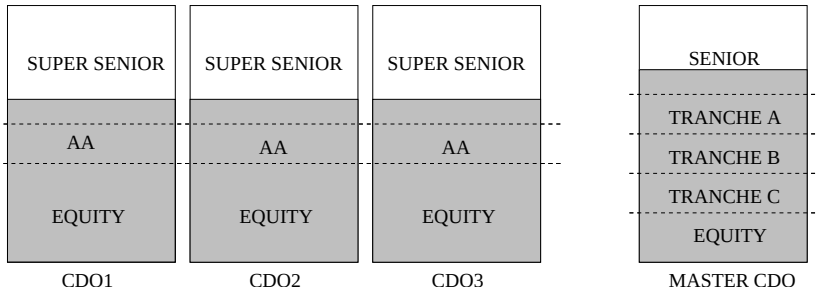


Figure 5.4: **Second scenario:** We consider the portfolio presented by figure 5.2. All the mezzanine CLOs'tranches have been fully amortized. The global loss at the level of the master CLO corresponds to the sum of each global loss occurring in the ICLOs. Some additional standalone assets are included in the portfolio of the master CLO. All the tranches but the most senior one are fully amortized. The senior tranche is partly amortized. The ABS and SN act like additional buffer against most severe loss events.

master CLO occurred (see figure 5.3-5.4). Except this additional subordination feature, the loss allocation scheme remains unchanged once we collected losses at the level of the master CLO. This means that we only need to know the amount of current loss due to a new facility, to compute the global loss at the level of the master CLO and to allocate this global loss through the tranche with respect to the nominal of each master investment tranche.

After computing for each scenario the loss amount we can determine the discrete density function β_L of the random variable L corresponding to the value of the loss (see Figure 5.5).

Algorithmic implementation

We present here the general algorithm that allows us to compute the discrete CLO^2's Loss density function, each step has been described in previous subsections:

Input: Data collected during Step 1

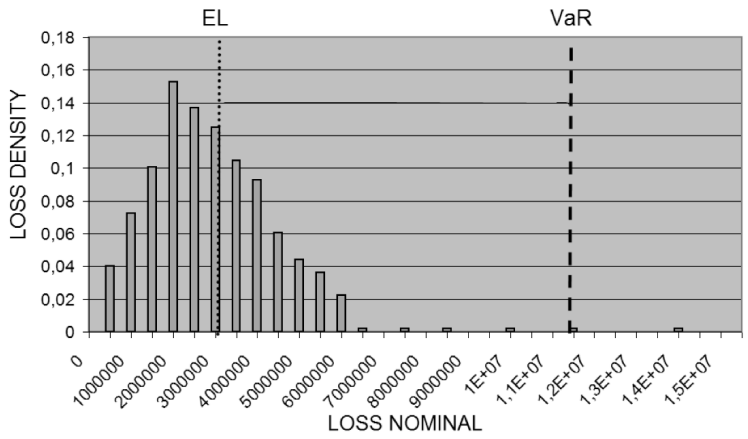


Figure 5.5: Example of generated density function of losses β_L. We have also reported on the graph the Expected Loss (**EL**) and Value at Risk (**VaR**).

For i=1:M

- **Generate a default time vector T.**
- **Compute Loss amount of the scenario i.**
- **Complete the discrete CLO^2's Loss density function β_L**

EndFor

Output: β_L

where $M \in \mathbb{R}$ corresponds to the iteration number of the Monte-Carlo scheme.

5.4.3 Step 3- Compute the desired portfolio's performance indicators

Using data stocked in $PORT$ and the discrete loss density function β_L computed in previous step, we can now compute various portfolio's performance indicators:

◦ **Portfolio's Nominal (PN):**

Money amount of the portfolio. It's given by:

$$PN(PORT) = \sum_{i=1}^{n} N_i \tag{5.6}$$

◦ **Income (IC)**

Money received by a person or organization because of **return** on investments. It's given by:

$$IC(PORT) = \sum_{i=1}^{n} Sp_i \times N_i \tag{5.7}$$

◦ **Expected Loss (EL)**

The expected value, over some specified horizon, of portfolio's losses due to default. EL can be easily computed from β_L:

$$EL(PORT, \beta_L) = \int_0^{PN} x\beta_L(x)dx \tag{5.8}$$

◦**Economic Capital (EC)**

It's a market value of assets minus fair value of liabilities. Used in practice as a risk-adjusted **capital** measure; specifically, the amount of capital required to meet an explicit solvency constraint (i.e. a the amount of cash to be retained by the financial institution to cover high losses).

We can approximate it using a regression formula. This formula is given by the financial institution calibrating coefficients with market evolution and the maturity, the PD, the LGD, the geographical zone and the rating of each facility.

For example: In 2003, the economic capital of BNP was of the form:

$$EC(PORT) = c_1 \times \sum_{i=1}^{n} EC_i(PORT) \times x_i \tag{5.9}$$

with

$$\begin{aligned} EC_i(PORT) = \quad & exp(c_2 + c_3 \log(PD_i) + (c_4 \times 1_{T\leq 1} + (c_5 - log(PD_i) \\ & \times c_6 \times 1_{T_i \geq 1}) \times log(T_i) + c_7 \times log(LGD_i)) \end{aligned} \tag{5.10}$$

where $c_1, ..., c_7$ are coefficients computed using market evolution.

◦ **Risk Adjusted Return On Capital (RAROC)**

The expected after-tax return divided by Economic Capital. RAROC is typically employed to evaluate the relative performance of business segments that have different levels of risk; the difference levels of risk are reflected in the denominator. Evaluating financial performance under RAROC calls for comparison to a benchmark return. It can be seen as a profitability measure. RAROC is given by:

$$RAROC(PORT, \beta_L) = \frac{IC(PORT) - EL(PORT, \beta_L)}{EC(PORT)} \tag{5.11}$$

◦ Risk measures: Value at Risk (VaR) and Conditional-VaR(CVaR)

Basically the risk is the chance that an investment's actual return will be different than expected. This includes the possibility of losing some or all of the original investment. In this part we give a short description of the mathematical definition of risk measures and introduce two of the most popular ones: Value at Risk (VaR) and Conditional-VaR(CVaR).

Mathematical definition of Risk measure The basic objects of our study are the random variables on the set of states of nature at a future date, interpreted as possible future values of **positions** or portfolio currently held.

We denote by Ω a finite set of states of nature. Considering Ω as the set of outcomes of an experiment, we compute the final net worth of a position for each element of Ω. It's a random variable denoted by X. We consider the probability space $L^\infty(\Omega, \Im, \mathbb{P})$, the space of all equivalence classes of bounded real valued random variables. It represent the set of all risks, that is the set of all real valued functions on Ω.

Definition 28 *Any mapping $\varpi : L^\infty(\Omega, \Im, \mathbb{P}) \to \mathbb{R}$ is called risk measure.*

In order to have a reasonable modeled risk measure, we prefer to use measures that satisfy four axioms [64, 71]:

Definition 29 *A mapping $\varpi : L^\infty(\Omega, \Im, \mathbb{P}) \to \mathbb{R}$ is called a coherent risk measure if the following properties hold:*

- *If $X \geq 0$ then $\varpi(X) \leq 0$.*
- *Sub-additivity: $\varpi(X_1 + X_2) \leq \varpi(X_1) + \varpi(X_2)$.*
- *Positive homogeneity: For $\lambda \geq 0$ we have $\varpi : (\lambda X) = \lambda \varpi(X)$.*
- *For every constant function a we have that $\varpi(a + X) = \varpi(X) - a$.*

Financial interpretation of each axiom can be found in literature [64].

VaR and CVaR VaR and CVaR (also called Expected Shortfall) approaches have been developed in various paper [63, 65]. Here we only propose a summary:

Literally, considering a given confidence level α given (i.e. a percent level), the VaR_α is defined as the smallest loss of the worst α % of losses and the $CVaR_\alpha$ is the average of the worst α % of losses.

More precisely, let $f(x, y)$ be the loss associated with a decision vector $x \in X \subset \mathbb{R}^n$ and a random vector $y \in \mathbb{R}^n$. y is governed by a probability Borel measure $\mathbb{P}$ on Y that is independent of x. We assume $f(x, y)$ is continuous in x and measurable in y, and $E[\|f(x, y)\|] < \infty$ for each $x \in X$.

For each x, we denote by $\Psi(x, .)$ on $\mathbb{R}$ the resulting distribution function for the loss $z = f(x, y)$:

$$\Psi(x, \alpha) = \mathbb{P}\{y / f(x, y) \leq \zeta\} = \int_{f(x,y)\leq\alpha} \rho(y)dy \tag{5.12}$$

with ρ the density function of y.

The VaR_α and $CVaR_\alpha$ risk measurement is defined as:

Definition 30 *The VaR_α and $CVaR_\alpha$ values for the loss random variable associated with x and any specified probability level α in $(0,1)$ are defined by:*

$$VaR_\alpha(x) = min\{\zeta \in \mathbb{R} : \Psi(x,\alpha) \geq (1-\alpha)\} \tag{5.13}$$

and

$$CVaR_\alpha(x) = \frac{1}{\alpha}\int_{f(x,y)\geq VaR_\alpha(x)} f(x,y)\rho(y)dy \tag{5.14}$$

$CVaR_\alpha$ measure is generally preferred to VaR_α because, in certain cases, $CVaR_\alpha$ is a convex and coherent measure:

Theorem 31 *If $f(x,y)$ is convex with respect to x, then $CVaR_\alpha(x)$ is convex with respect to x as well.*

Theorem 32 *$CVaR_\alpha$ is a coherent risk measure: when $f(x,y)$ is linear with respect to x, not only is $CVaR_\alpha(x)$ sub-linear with respect to x, but furthermore it satisfies:*

$$CVaR_\alpha(x) = c \ when \ f(x,y) \equiv c$$

and obeys the monotonicity rule that:

$$CVaR_\alpha(x) \leq CVaR_\alpha(x') \ when \ f(x,y) \leq f(x',y)$$

Implementation VaR_α and $CVaR_\alpha$ are given by the following formulas:

- The α-Value at Risk calculated is from the loss distribution:

$$VaR_\alpha(\beta_L) = inf[L' | \int_0^{L'} \beta_L(x)dx > (1-\alpha)] \tag{5.15}$$

- In the case of the discrete distribution β_L, the $CVaR_\alpha$ is given by:

$$CVaR_\alpha(\beta_L) = \frac{1}{\alpha}\int_0^{\alpha} inf[L' | \int_0^{L'} \beta_L(x)dx > (1-p)]dp \tag{5.16}$$

5.5 Portfolio optimization problems

In this section we are interested by optimizing the allocation structure (i.e. the nominal of each facility in a portfolio) of BNP-Paribas Portfolio-Management (PM) portfolio, respecting to given constraints, in order to improve his performances at one year horizon. By improving performances, we mean:

- P_1 **Reduce risk measure keeping the income higher than the initial value**: The risk measures considered here are VaR and ES with a confidence level set to $\alpha = 0.1\%$, this level is often used in banking system. It avoid too extreme, and thus non realistic, risk scenarios. Both measures have led to the same kind of results. However, only VaR one is presented here, because it's the more difficult and challenging to optimize [71, 63].

	$\text{VaR}_{0,1\%}$	RAROC	IC
P_1	minimize		$\geq ini$
P_2	$\leq ini$		maximize
P_3		maximize	$\geq ini$
P_4	minimize	maximize	maximize

Table 5.2: Problems to be solved using HSGA optimization method.

- P_2 **Maximize income keeping risk measure lower than the initial value**.
- P_3 **Maximize profitability keeping the income higher than the initial value:** We use the RAROC as profitability measure to be maximized.
- P_4 **Maximize income, Maximize profitability and Reduce risk measure:** This problem is only studied here in order to see interaction between each characteristics.

All optimization problems are listed in Table 5.2.

The initial PM portfolio has a CLO^2 structure and is compound by 500 facilities dispatched in 40 ICLOs'tranches and 54 Single-Names ('SN'). The portfolio nominal is close to 2.e9 Euros (E) and its income near to 2.1e7 E. Its $VaR_{0,01\%} = 1.9e8$ E and its $RAROC = 90\%$.

5.5.1 Parameterization

Parameters considered here are the nominal of each facility included in the portfolio, here ICLO's tranches and SN included, not only in the initial Portfolio, but also in the entire PM universe (1500 facilities). They are represented by a real vector of the form:

$$x = (Nominal_{CLO_1}, ..., Nominal_{SN_1}, ..., Nominal_{SN_{1500}}) \tag{5.17}$$

In order to obtain a versatile portfolio, respecting the BNP-Paribas investment guideline (i.e. investment rules), all parameters are subject to the following constraints:

- **Avoid too much concentration in one facility:** Each nominal of ICLOs'tranche or SN must be inferior to a certain value, typically: 1e8 E.
- **Minimum facility investment:** Each ICLO's tranche or SN with a nominal lower than 5e6 E is abandoned (i.e. its nominal is set to 0 E).
- **Facility quality:** ICLOs'tranches or SN which have to be modified must have a ROPE superior to 16%, a RAROC superior to 30% and a Rating better than BBB-. Otherwise depending on the facility **liquidity**:
  - *Non liquid facility*: Nominal is kept to the initial PM portfolio's value.
  - *Liquid facility*: Nominal must be inferior to the initial PM portfolio's value.

Modifiable and non-null constant facilities are presented in Tables 5.4-5.5. Facilities satisfying those quality guideline are essentially included in financial sectors (KMV code 6 and 23) (this is visible on Figures 5.8-5.10-5.12-5.14-5.16 in Industry concentration graphs).

Due to those constraints, the total number of facilities which can be modified (added, increased or decreased) inside the portfolio is about $n_p = 65$. The admissible space is $\Omega = \prod_{i=1}^{n_p}[b_i, u_i]$, where:

- $b_i = 0$ and $u_i = initial\ \ value$ in cases when the facility can only be decreased.
- $b_i = 0$ and $u_i = 1e8$ in other cases.

5.5.2 Cost function

For each problem defined in Table 5.2, the cost function $J(x)$ corresponds to the sum of the desired performances'values of the portfolio associated with parameters x in Ω. $J : \Omega \to \mathbb{R}$ is of the form:

$$J(x) = \sum_{n=1}^{nchar} \xi_i J_n(x) \tag{5.18}$$

where $\{J_n\}_{n=1,\dots,nchar}$ correspond to the $nchar$ desired performances to be optimized, computed following the model proposed in section 5.4, and ξ_i to a fixed weight coefficient.

N.B: Only P_4 deals with multi-criteria optimization. A better approach is to use a Nash type algorithm. Future works should explore this way by combining our algorithms with a Nash's one.

Furthermore J is minimized according to $ncons$ constraints on facilities and portfolio, thus minimization problems can be written as:

$$\begin{gathered} \min_{x\in\Omega} \textstyle\sum_{n=1}^{nchar} \xi_i J_n(x) \\ lc_1 \le C_1(x) \le uc_1 \\ \vdots \\ lc_{ncons} \le C_{ncons}(x) \le uc_{ncons} \end{gathered} \tag{5.19}$$

where C_j, uc_j and lc_j correspond respectively to the constraint function, the upper and lower boundary value of the $j^{Th}constraint$.

In order to solve (5.19), according to the work performed in [64], we reformulate the optimization problem considering a new functional in Ω to be minimized:

$$\widetilde{J}(x) = J(x) + \vartheta \sum_{j=1}^{ncons} (max(uc_j - C_j(x), 0) + max(C_j(x) - lc_j, 0)) \tag{5.20}$$

where $\vartheta \gg 1$. In (5.20) the constraints are directly included in the cost function using wall functions.

$\widetilde{J}(x)$ is evaluated using an algorithm based on the default time model described in previous 5.4.

5.5.3 Results and discussion

The initial portfolio's allocation structure and loss variation are depicted in Figure 5.7. For each problem, main portfolio's characteristics and their evolution between initial and optimized portfolios are presented in Table 5.3. Nominal concentrations of each modifiable SN according to Rating, Spread, Country and Industry are given by Figure 5.8.

The algorithm used here is the HSGA, applied with parameters presented in section 2. We prefer to use a non-gradient based method as we don't need too much precision on the optimized result (Due to market fluctuation it's impossible to strictly respect the optimized allocation) and as sensitivity analysis is difficult to perform. Indeed, Monte-Carlo scheme used to compute $\widetilde{J}(x)$ needs about 1e6 iterations to capture effects of short portfolio modifications, that requires about 20 minutes for one functional evaluation on a P4-3GHz/1Gb computer which is too time consuming.

In order to reduce drastically computational time, default times vector is generated once during the first evaluation and stocked into computer memory. For 1e5 Monte-Carlo simulations, evaluation time decrease from 2 minutes to 10 seconds. In counterpart, it requires about 1Gb of computer memory for 2e5 iterations and 5 Gb for 1e6 iterations. As we don't need too much precision on the approximation of the global optimum, Monte-Carlo iteration number is set to 1e5.

Using those set of parameters, the HSGA iteration number is about 2000 and the total optimization time is closed to 6 hours.

Furthermore, in order to improve algorithm accuracy and avoid unnecessary computations, computed portfolio's parameters outside the admissible space (i.e. when constraints are not satisfied) are projected inside it, using a real proportional coefficient, before performing Monte-Carlo simulation. When projection technique fail, only the part of function (5.20) associated with non-respected constraint is computed.

Convergence histories of the best element for each optimization problem are presented in Figure 5.6.

At the end of each optimization, a sensibility analysis is performed on initial and optimized portfolios IC, VaR and RAROC. Facilities'nominals are randomly increased or decreased by 1e7 E. Results are reported in Table 5.3. As we can see, depending on the characteristic to be optimized, the optimized result is more stable than the initial one.

P_1 - $\mathbf{VaR}_{0,1\%}$ reduction with income $\geq$ 2.1e7 Optimized portfolio is described by Figure 5.9. $\mathrm{VaR}_{0,1\%}$ has been reduced by 30 % of its initial value. Portfolio's income is kept to its initial value. This is foreseeable as the result must be situated on the constraint border: A portfolio having an income superior to 2.1e7 can be improved by projecting it on the constraint border, the risk is then reduced as each facility nominal is decreased.

Result obtained by HSGA suggests to choose a diversified allocation structure, with a high number of different facilities, each one having an average nominal of 2e7 E (less for higher risk ones, more for others).

Due to the high correlation and overlap (the same product is present in various ICLOs) between each ICLO's tranches, the total nominal invested on this kind of facility is reduced by 26 %. In fact, although a simple ICLO is robust to low loss scenarios, combining those ICLO with a high nominal increase the high loss scenarios probability: if a default occurs

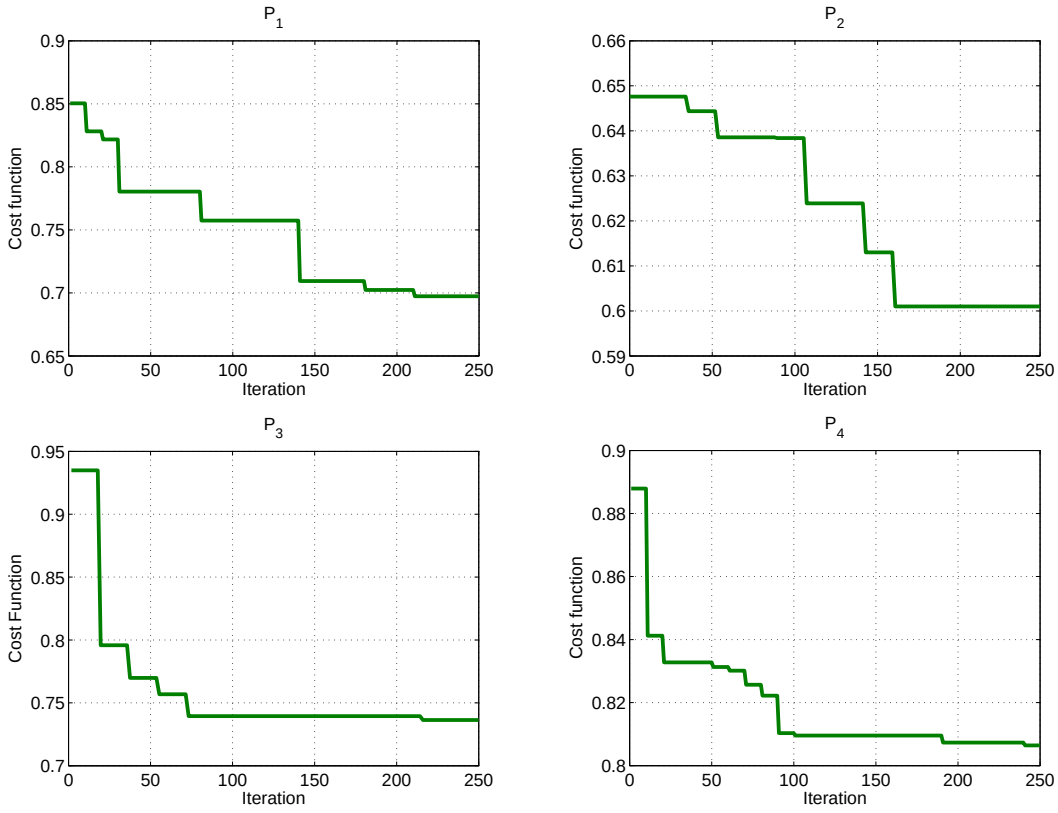


Figure 5.6: Convergence histories of the best element during optimization process for problems P_1, P_2, P_3 and P_4.

in one ICLO other ICLOs have higher chances to be also impacted.

In comparison, investing on diversified Single-Names with a reasonable nominal amount (around 2e7 E) decrease the chance to encounter high loss scenarios: Facility's defaults in various sectors (although here banking sector is privileged due to our investment guideline) and countries should occur in a same scenario to raise a critical loss amount. Thus the total nominal is increased and divided in all eligible Single-Names. This can be observed on Figure 5.10 where nominal concentration of each modifiable SN according to Rating, Spread, Country and Industry is regularly, according to investment guideline, dispatched.

P_2 **- Income maximization with $\mathbf{VaR}_{0,1\%}$ <1.9e8** Optimized portfolio is depicted in Figure 5.11. Income has been increased by 40%. As a consequence total portfolio's nominal has augmented by 40 %, but for the same reasons as previously, SN's nominal proportion has been raised in the total nominal amount (+70%). In order to improve the portfolio's income and control its risk level, essentially SN combining good spread and good rating have been privileged. Another consequence of the risk constraint is that the optimized portfolio still relatively diversified. Optimized portfolio's $\mathrm{VaR}_{0,1\%}$ is equals to the constraint boundary, as in problem P_1 this is intuitive. All those remarks are observable analyzing the SN's concentrations given by Figure 5.12.

P_3 **- RAROC maximization with income $\geq$ 2.1e7** Optimized portfolio's allocation structure and loss distribution are presented in Figure 5.13. RAROC and Nominal have been increased by 26%. The total nominal of ICLOs'tranches has been increased to its maximal value. Good rating SN's nominals have also been drastically raised. Due to their high quality rating those products consume less economic capital than other products and have a good RAROC ratio. Furthermore ICLOs'tranches have better spread than SN, thus those kind of facility should be privileged: Here the nominal limit is reached but if we raise it, the proportion of nominal of ICLOs'tranches in total portfolio's nominal will increase. Depending of their rating, other facilities are drastically decreased (medium rating) or abandoned (poor rating). The granularity of the portfolio is severely decreased. This is visible on Figure 5.14, where concentration is higher around SN with a rating near to 5 and a spread close to 27. A direct impact is the reduction of country and industry diversification.

P_4 **- $\mathbf{VaR}_{0,1\%}$ reduction, Income and RAROC maximization** This problem was proposed in order to see the interaction between RAROC maximization and $\mathrm{VaR}_{0,1\%}$ reduction as they have opposite effects on optimized solution. Weight coefficients ξ_i defined in cost function (5.19) have been set to 1. Results are depicted in Figure 5.15. All objectives have been satisfied: $\mathrm{VaR}_{0,1\%}$ has been reduced by 26%, Income and RAROC have been raised by respectively 17 % and 7 %. The portfolio is diversified and the SN's nominal has been increased instead of ICLOs'tranches'one which has decreased. However, in order to satisfy RAROC and Income requirements, some good rating or good spread SN are privileged and have a greater notional than other facilities. This is observable on SN's concentrations depicted by Figure 5.16 with a higher value for products with a good rating (and thus a low spread) and, in weak proportion, a good spread (but bad rating).

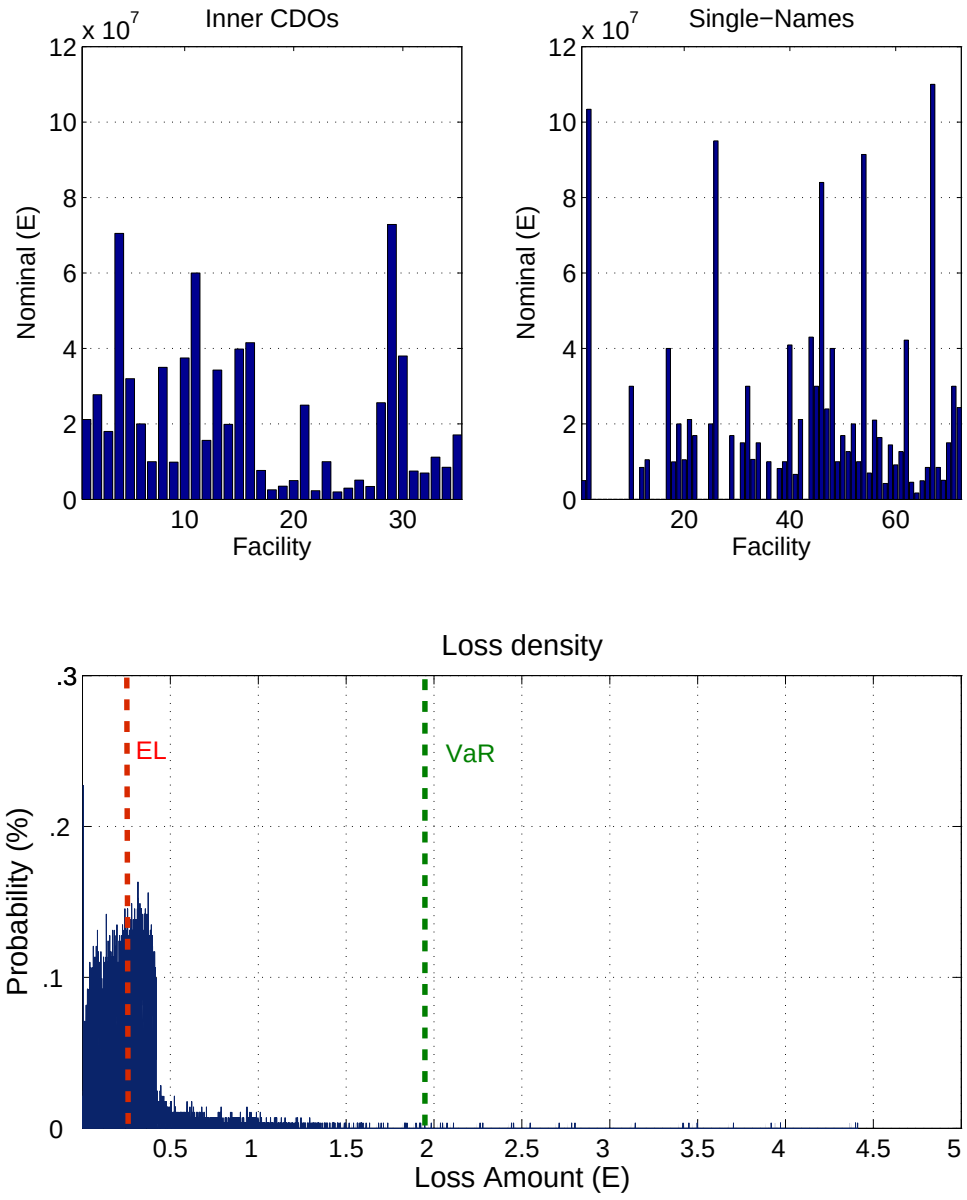


Figure 5.7: Initial PM Portfolio's Facility allocation (**Up**) and Loss Distribution (**Bottom**)

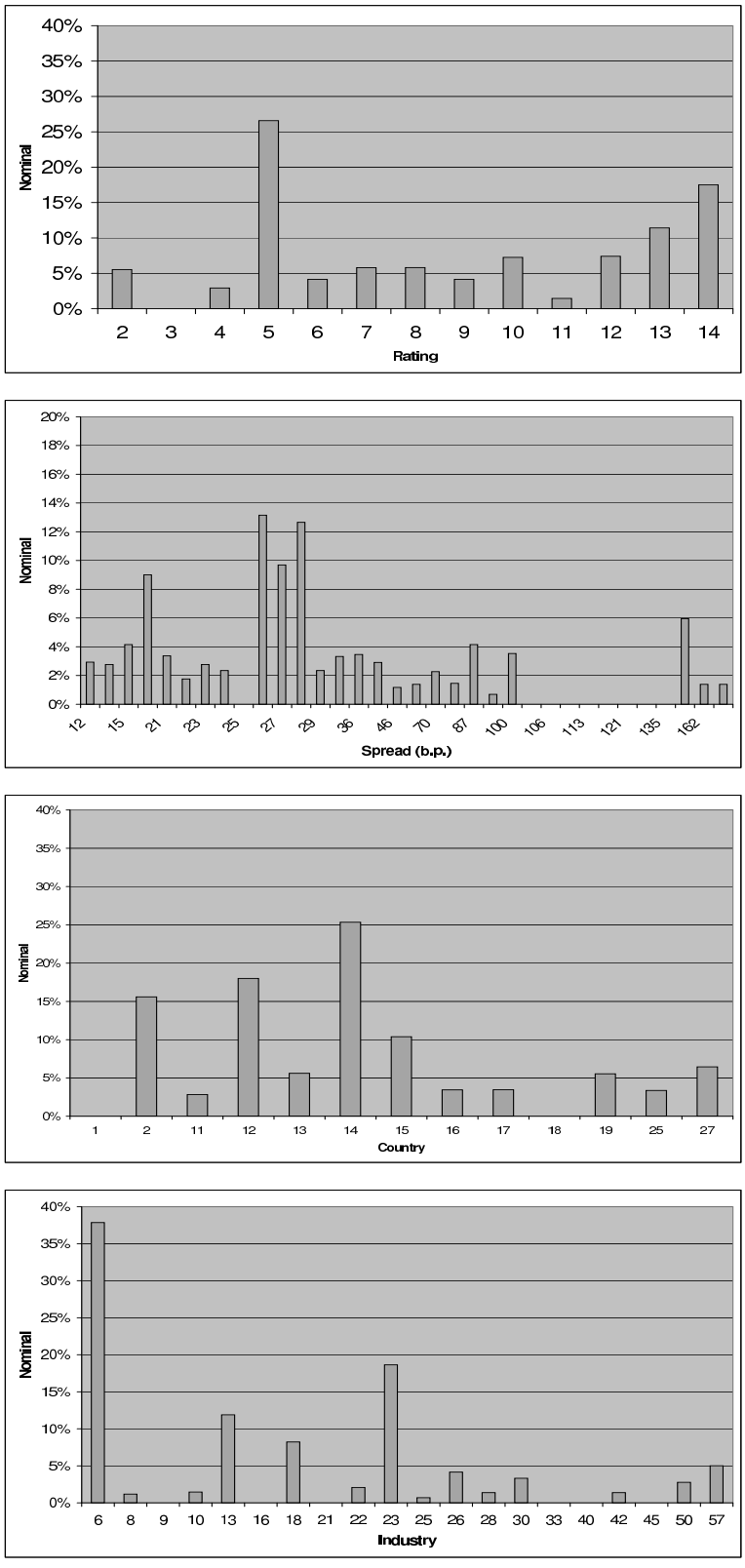


Figure 5.8: Initial PM Portfolio's modifiable SN's concentration: From (**Top**) to (**Bottom**): Rating, Spread (in b.p.), KMV Country zone and KMV Industry zone.

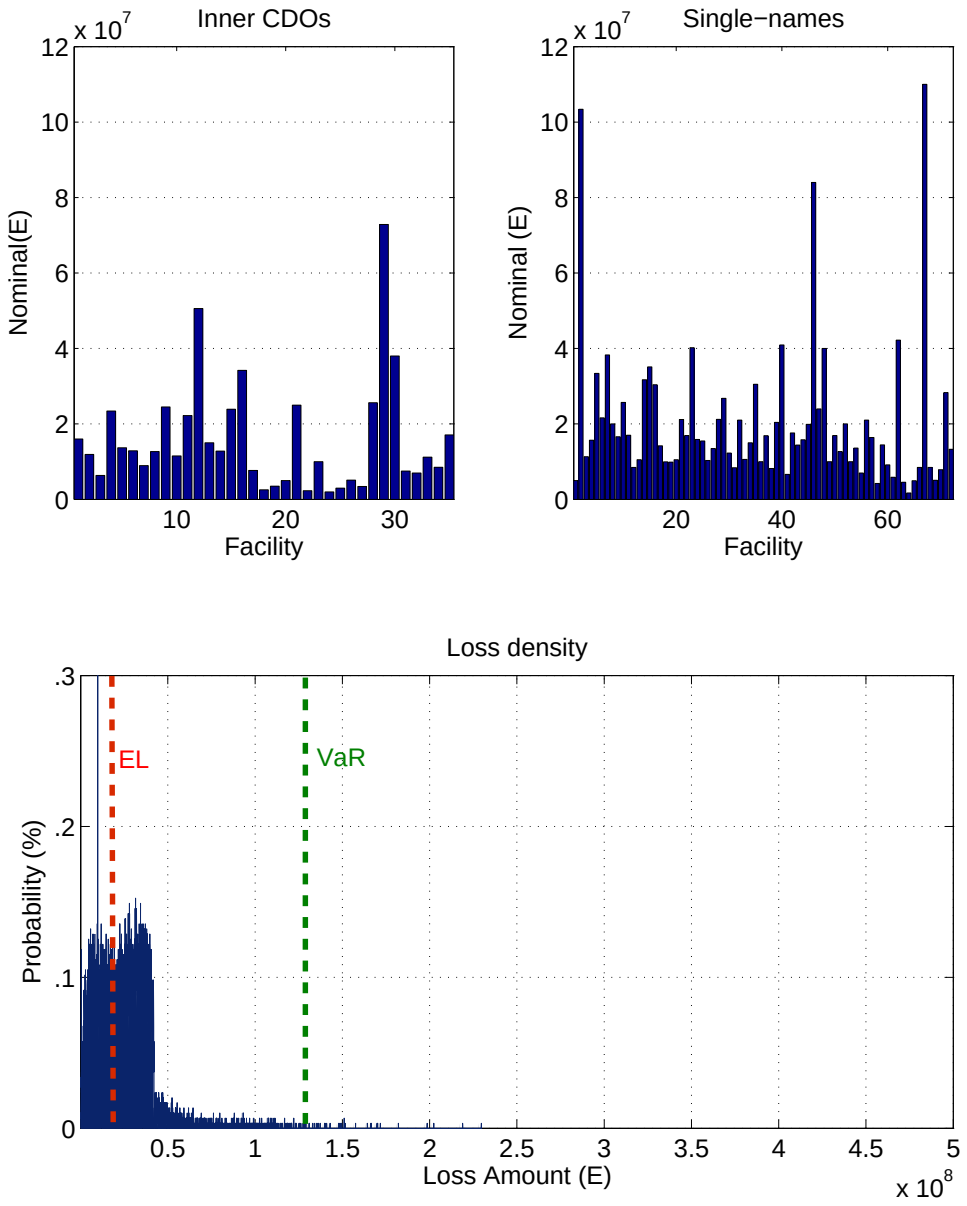


Figure 5.9: VaR optimized Portfolio's Facility allocation (**Up**) and Loss Distribution (**Bottom**)

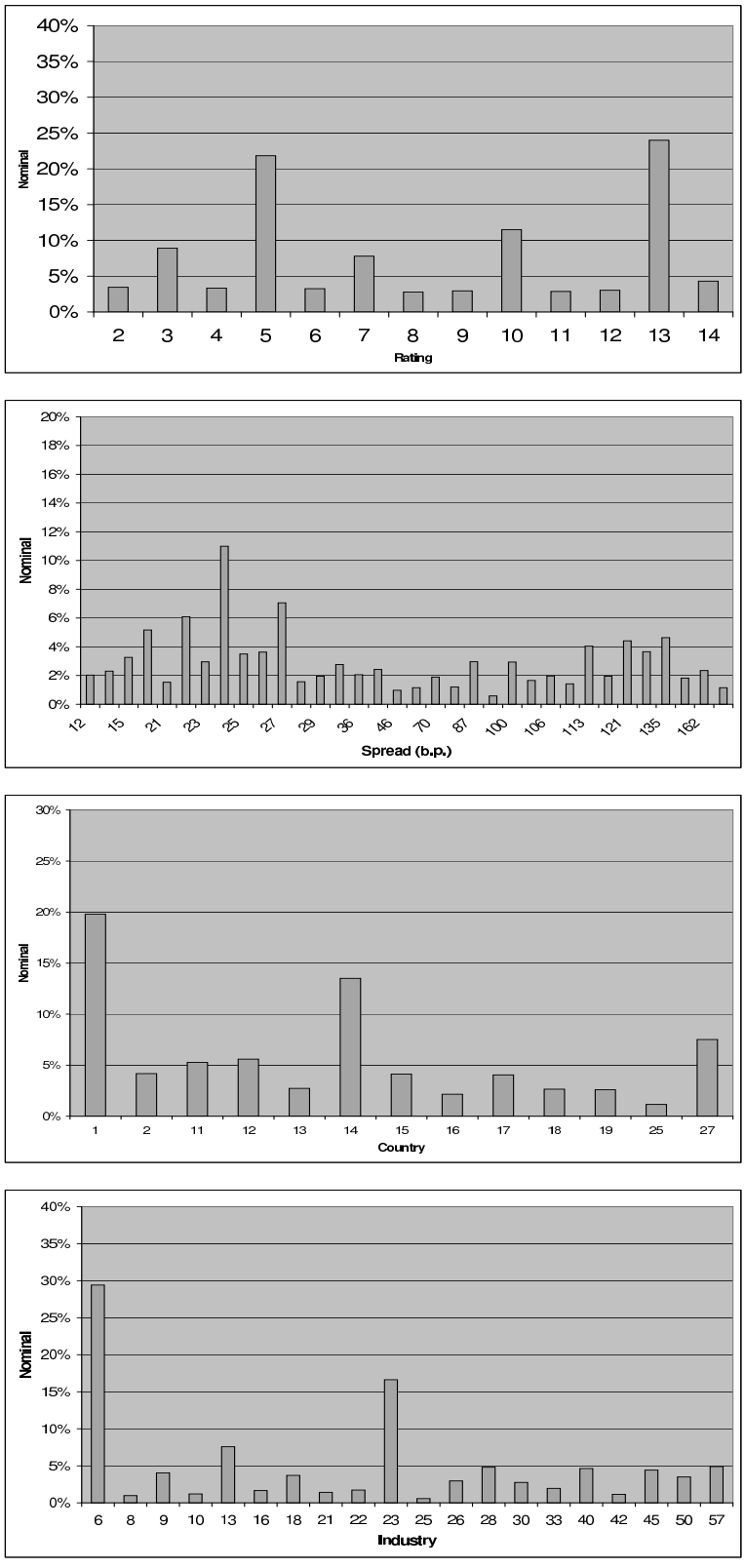


Figure 5.10: VaR optimized Portfolio's modifiable SN's concentration: From (**Top**) to (**Bottom**): Rating, Spread (in b.p.), KMV Country zone and KMV Industry zone.

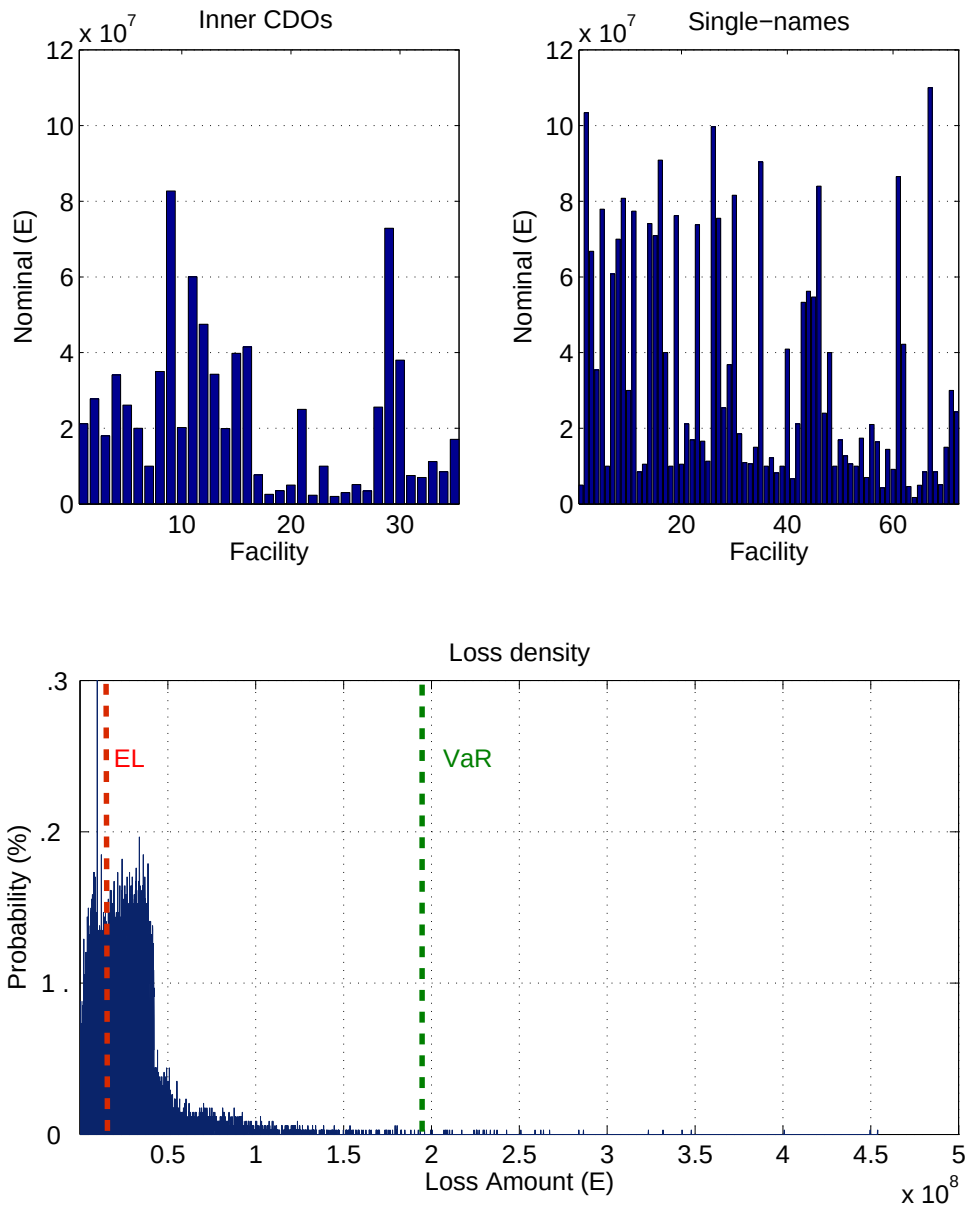


Figure 5.11: Income optimized Portfolio's Facility allocation (**Up**) and Loss Distribution (**Bottom**)

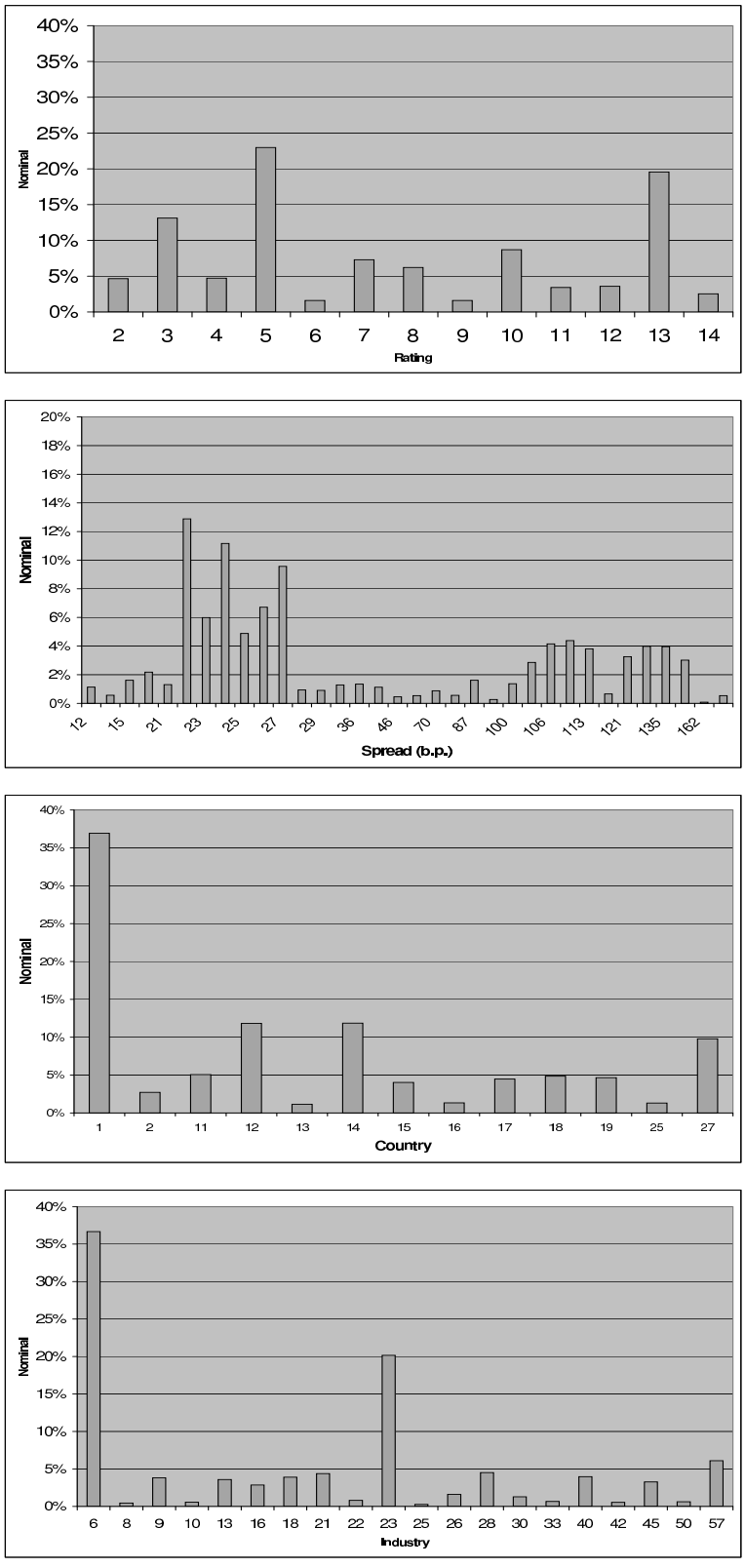


Figure 5.12: Income optimized Portfolio's modifiable SN's concentration: From (**Top**) to (**Bottom**): Rating, Spread (in b.p.), KMV Country zone and KMV Industry zone.

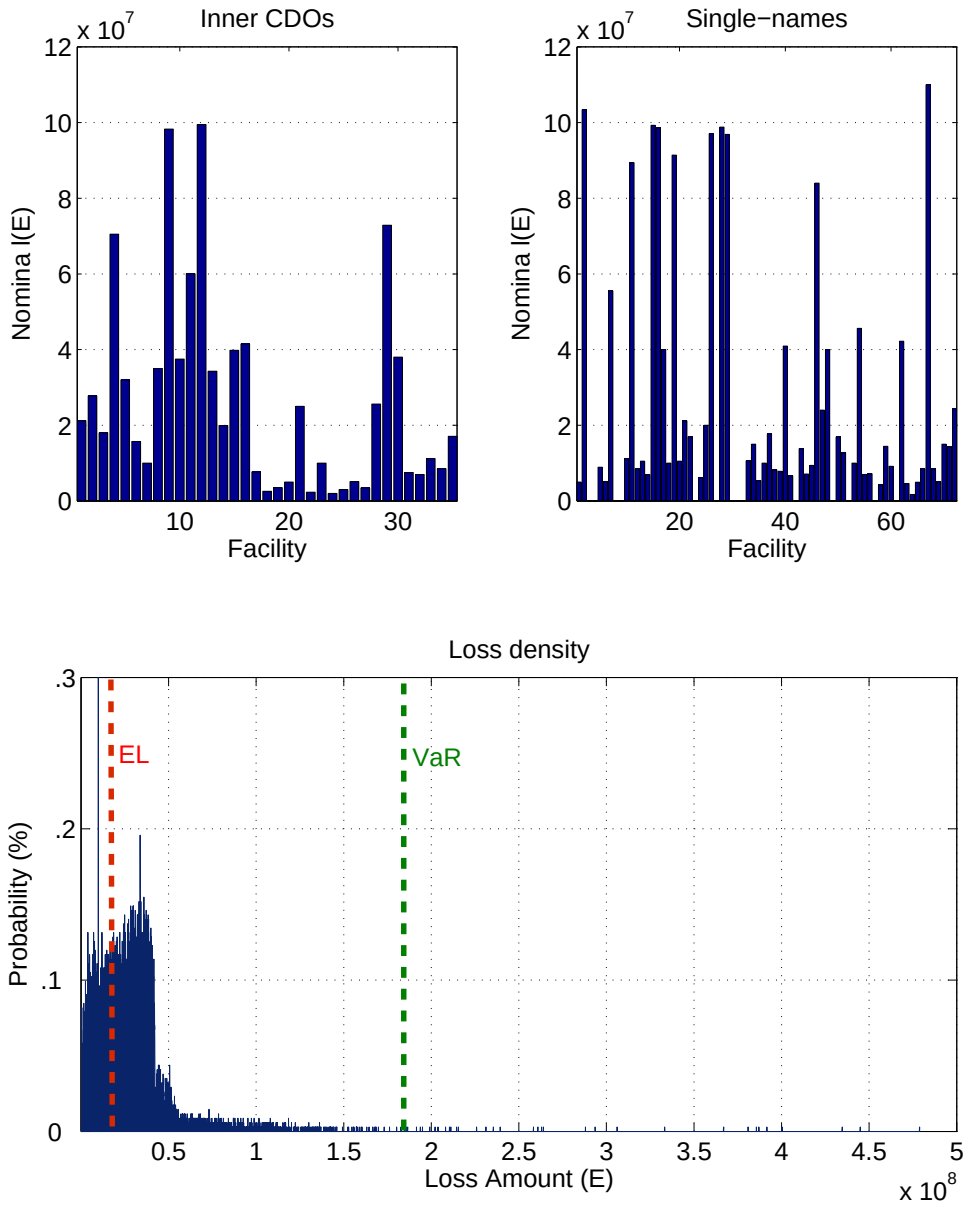


Figure 5.13: RAROC optimized Portfolio's Facility allocation (**Up**) and Loss Distribution (**Bottom**)

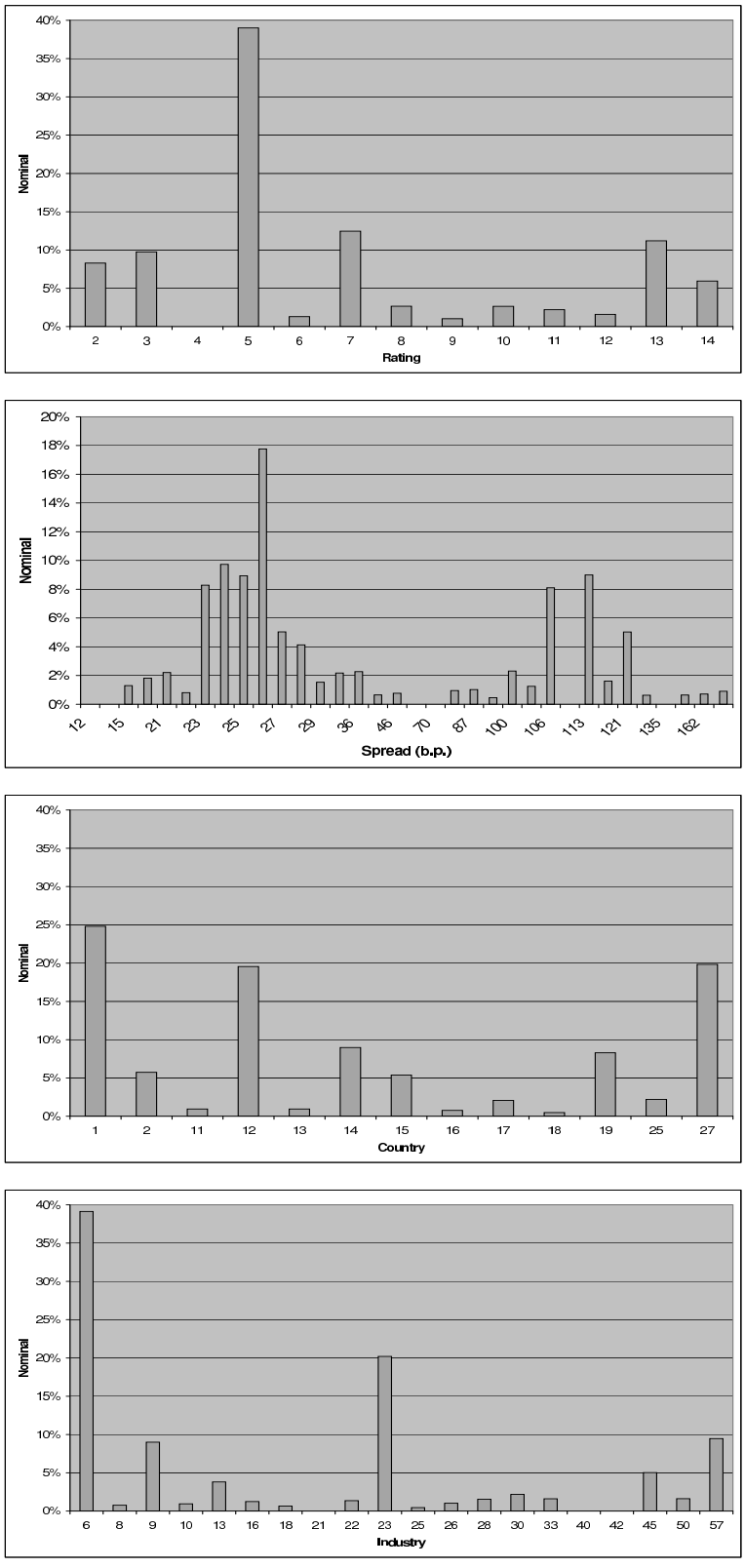


Figure 5.14: RAROC optimized Portfolio's modifiable SN's concentration: From (**Top**) to (**Bottom**): Rating, Spread (in b.p.), KMV Country zone and KMV Industry zone.

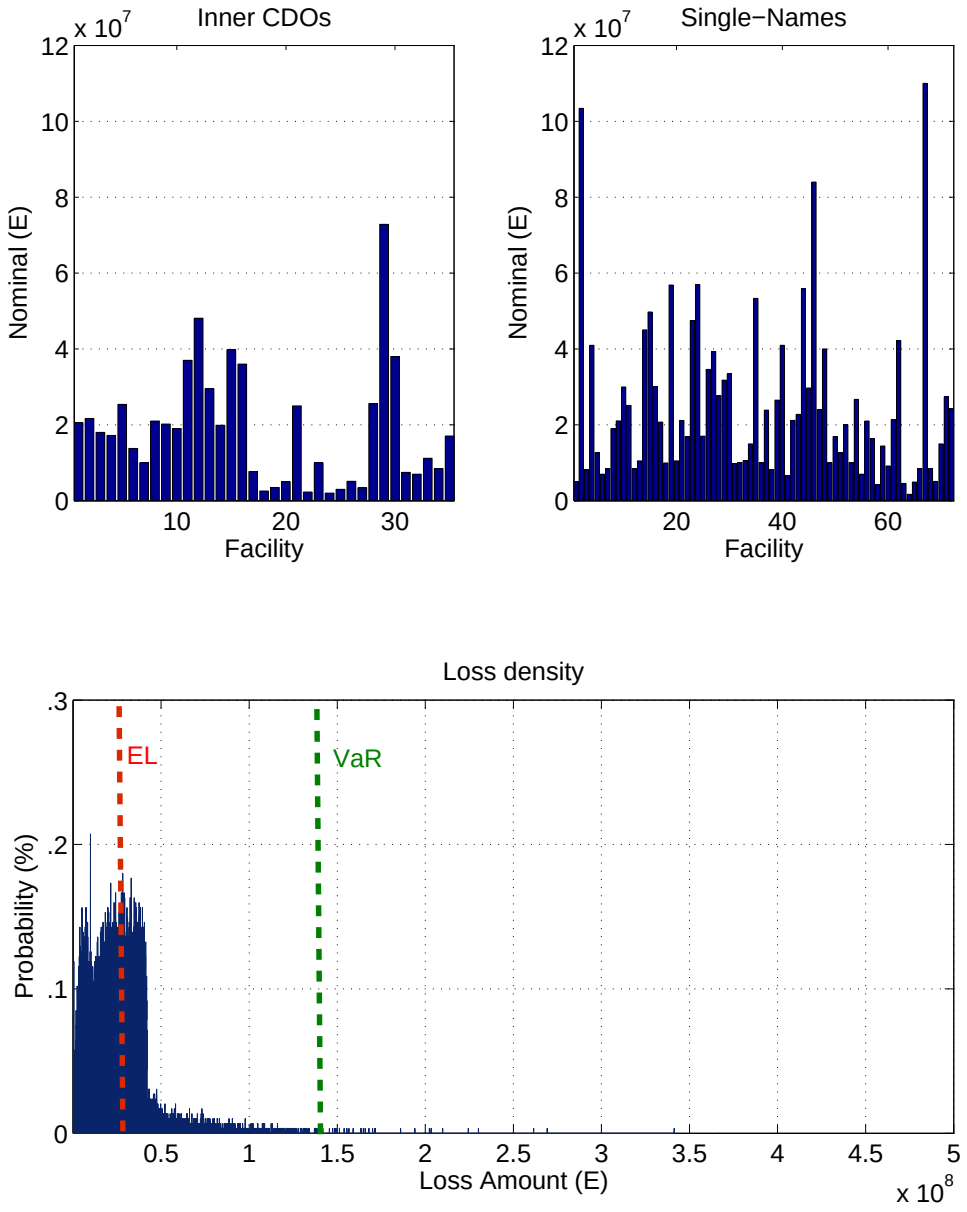


Figure 5.15: Combined characteristic optimized Portfolio's Facility allocation (**Up**) and Loss Distribution (**Bottom**).

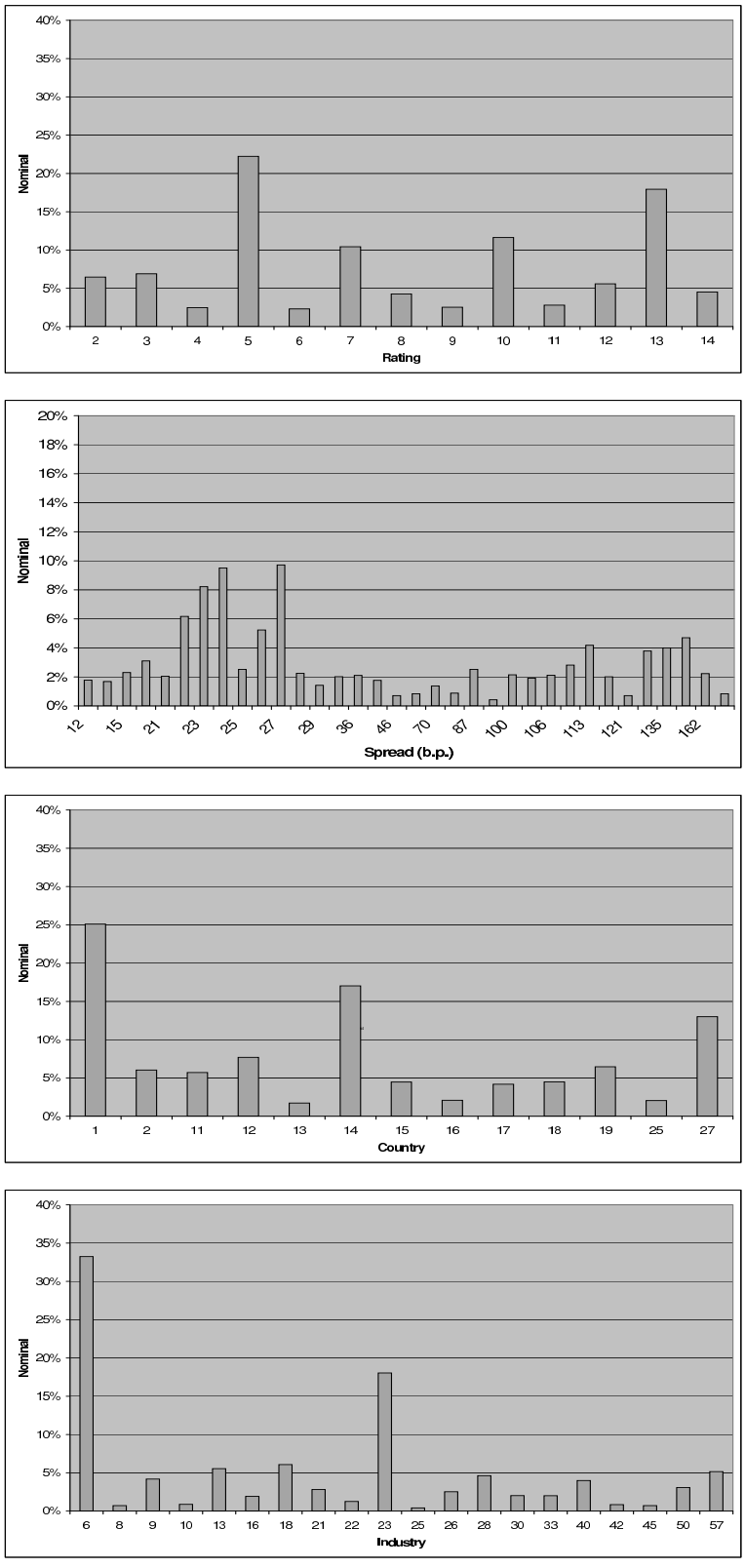


Figure 5.16: Combined optimized Portfolio's modifiable SN's concentration: From (**Top**) to (**Bottom**): Rating, Spread (in b.p.), KMV Country zone and KMV Industry zone.

	Nominal	Nom. ICLO	Nom. SN	Income	$VaR_{0,1\%}$	RAROC
Initial	2.e9	7.5e8	1.25e9	2.1e7	1.9e8	90%
Sen.	-	-	-	3%	7%	2%
P_1	2e9	5.5e8	1.45e9	2.1e7	1.3e8	87%
Evo.	0%	-26%	11%	0%	-31%	-3%
Sen.	-	-	-	1%	1%	1%
P_2	3.2e9	7.5e8	2.7e9	3.e8	1.9e8	92%
Evo.	57%	6%	88%	28%	0%	2%
Sen.	-	-	-	2%	1%	1%
P_3	2.6e9	9.e8	1.7e9	2.7e7	1.75e8	113%
Evo.	27%	23%	29%	24%	-8%	26%
Sen.	-	-	-	3%	3%	1%
P_4	2.7e9	73e8	2.e9	2.5e7	1.4e8	96%
Evo.	31%	-16%	57%	17%	-26%	7%
Sen.	-	-	-	5%	1%	2%

Table 5.3: PM portfolio's optimization results. From left to right main portfolio's characteristics: Total nominal, ICLOs'tranches'nominal, SN's nominal, portfolio's income, $VaR_{0,1\%}$ and RAROC. From (**Top**) to (**Bottom**), initial and optimized portfolios. Evolution (Evo.) between initial and optimized portfolio and a sensitivity analysis (Sen.) are also reported.

5.6 Conclusions

Results obtained with our HSGA method are in adequacy with financial intuition: depending on the cost function definition, particular facilities characteristics (here spread and rating) have been privileged.

However, even if this intuition can help us a to generate a general portfolio's structure, optimization methods furnish a precise nominal allocation for each facility and are powerful tools to help portfolio managers to improve portfolio's performances.

Identity	Country	Industry	Rating	Spread	Nom. Init.	Nom. P1	Nom. P2	Nom. P3	Nom. P4	Type
ICLOs										
1	#NA	#NA	5	113	2,12E+07	2,12E+07	1,60E+07	2,12E+07	2,06E+07	=
2	#NA	#NA	5	200	2,78E+07	2,78E+07	1,19E+07	2,78E+07	2,17E+07	=
3	#NA	#NA	3	95	1,80E+07	1,80E+07	6,37E+06	1,80E+07	1,80E+07	=
4	#NA	#NA	5	74	7,05E+07	3,41E+07	2,34E+07	7,05E+07	1,72E+07	D
5	#NA	#NA	6	140	3,20E+07	2,61E+07	1,37E+07	3,20E+07	2,54E+07	D
6	#NA	#NA	6	80	2,00E+07	2,00E+07	1,29E+07	1,57E+07	1,38E+07	=
7	#NA	#NA	6	192	1,00E+07	7,55E+06	8,94E+06	1,00E+07	1,00E+07	D
8	#NA	#NA	5	146	3,50E+07	3,50E+07	1,27E+07	3,50E+07	2,10E+07	=
9	#NA	#NA	2	173	9,88E+06	8,27E+07	2,45E+07	9,83E+07	2,02E+07	U
10	#NA	#NA	5	96	3,75E+07	2,02E+07	1,15E+07	3,75E+07	1,90E+07	D
11	#NA	#NA	3	100	6,00E+07	6,00E+07	2,22E+07	6,00E+07	3,70E+07	=
12	#NA	#NA	6	150	1,57E+07	4,75E+07	5,06E+07	9,95E+07	4,81E+07	U
13	#NA	#NA	5	115	3,43E+07	3,43E+07	1,50E+07	3,43E+07	2,95E+07	=
14	#NA	#NA	14	192	1,99E+07	1,99E+07	1,28E+07	1,99E+07	1,99E+07	=
15	#NA	#NA	15	185	3,98E+07	3,98E+07	2,39E+07	3,98E+07	3,98E+07	=
16	#NA	#NA	13	60	4,15E+07	4,15E+07	3,42E+07	4,15E+07	3,60E+07	=
17	#NA	#NA	2	60	7,68E+06	7,68E+06	7,68E+06	7,68E+06	7,68E+06	=
18	#NA	#NA	6	75	2,56E+06	2,56E+06	2,56E+06	2,56E+06	2,56E+06	=
19	#NA	#NA	2	37	3,50E+06	3,50E+06	3,50E+06	3,50E+06	3,50E+06	=
20	#NA	#NA	5	73	5,00E+06	5,00E+06	5,00E+06	5,00E+06	5,00E+06	=
21	#NA	#NA	9	158	2,50E+07	2,50E+07	2,50E+07	2,50E+07	2,50E+07	=
22	#NA	#NA	6	88	2,31E+06	2,31E+06	2,31E+06	2,31E+06	2,31E+06	=
23	#NA	#NA	3	88	1,00E+07	1,00E+07	1,00E+07	1,00E+07	1,00E+07	=
24	#NA	#NA	2	43	2,00E+06	2,00E+06	2,00E+06	2,00E+06	2,00E+06	=
25	#NA	#NA	3	68	3,00E+06	3,00E+06	3,00E+06	3,00E+06	3,00E+06	=
26	#NA	#NA	3	50	5,12E+06	5,12E+06	5,12E+06	5,12E+06	5,12E+06	=
27	#NA	#NA	6	73	3,42E+06	3,42E+06	3,42E+06	3,42E+06	3,42E+06	=
28	#NA	#NA	2	125	2,56E+07	2,56E+07	2,56E+07	2,56E+07	2,56E+07	=
29	#NA	#NA	5	60	7,29E+07	7,29E+07	7,29E+07	7,29E+07	7,29E+07	=
30	#NA	#NA	4	118	3,80E+07	3,80E+07	3,80E+07	3,80E+07	3,80E+07	=
31	#NA	#NA	6	298	7,50E+06	7,50E+06	7,50E+06	7,50E+06	7,50E+06	=
32	#NA	#NA	3	53	7,00E+06	7,00E+06	7,00E+06	7,00E+06	7,00E+06	=
33	#NA	#NA	5	78	1,12E+07	1,12E+07	1,12E+07	1,12E+07	1,12E+07	=
34	#NA	#NA	2	88	8,50E+06	8,50E+06	8,50E+06	8,50E+06	8,50E+06	=
35	#NA	#NA	6	88	1,71E+07	1,71E+07	1,71E+07	1,71E+07	1,71E+07	=
SN										
1	15	25	9	91	5,00E+06	5,00E+06	5,00E+06	5,00E+06	5,00E+06	=
2	14	29	5	223	1,03E+08	1,03E+08	1,03E+08	1,03E+08	1,03E+08	=
3	17	6	5	27	1,00E+00	6,68E+07	1,13E+07	1,00E+00	8,20E+06	U
4	1	6	11	23	1,00E+00	3,54E+07	1,57E+07	1,00E+00	4,09E+07	U
5	1	6	4	22	1,00E+00	7,79E+07	3,34E+07	8,91E+06	1,27E+07	U
6	14	23	6	24	1,00E+00	1,00E+00	2,16E+07	5,12E+06	7,00E+06	=
7	1	45	14	121	1,00E+00	6,08E+07	3,83E+07	5,55E+07	8,46E+06	U
8	17	4	12	228	2,50E+06	1,00E+00	1,00E+00	1,00E+00	1,00E+00	D
9	1	23	14	24	1,00E+00	8,08E+07	1,66E+07	1,00E+00	2,10E+07	U
10	14	26	10	87	3,00E+07	3,00E+07	2,57E+07	1,12E+07	3,00E+07	=
11	1	57	8	106	1,00E+00	7,74E+07	1,70E+07	8,94E+07	2,51E+07	U
12	16	8	14	46	8,46E+06	8,46E+06	8,46E+06	8,46E+06	8,46E+06	=
13	11	10	13	86	1,05E+07	1,05E+07	1,05E+07	1,05E+07	1,05E+07	=
14	1	28	11	126	1,00E+00	7,41E+07	3,17E+07	6,94E+06	4,50E+07	U
15	1	9	6	113	1,00E+00	7,09E+07	3,51E+07	9,93E+07	4,97E+07	U
16	12	6	4	25	1,00E+00	9,09E+07	3,04E+07	9,86E+07	3,00E+07	U
17	15	6	6	27	4,00E+07	4,00E+07	1,42E+07	4,00E+07	2,07E+07	=

Table 5.4: Detailed description of each facility (part1). The "**Type**" column indicates if a facility can be modified (**U**), only decreased (**D**) or unchanged (=).

18	42	59	6	26	9,98E+06	9,98E+06	9,98E+06	9,98E+06	9,98E+06	=
19	19	23	3	23	2,00E+07	7,62E+07	9,89E+06	9,14E+07	5,69E+07	U
20	13	13	12	0	1,05E+07	1,05E+07	1,05E+07	1,05E+07	1,05E+07	=
21	1	5	17	551	2,12E+07	2,12E+07	2,12E+07	2,12E+07	2,12E+07	=
22	18	28	8	60	1,69E+07	1,69E+07	1,69E+07	1,69E+07	1,69E+07	=
23	11	40	14	135	1,00E+00	7,38E+07	4,02E+07	1,00E+00	4,75E+07	U
24	27	6	8	27	1,00E+00	1,66E+07	1,59E+07	6,17E+06	5,70E+07	U
25	12	23	15	20	2,00E+07	1,13E+07	1,55E+07	2,00E+07	1,70E+07	D
26	12	6	6	26	9,50E+07	9,97E+07	1,03E+07	9,71E+07	3,46E+07	U
27	1	6	4	22	1,00E+00	7,55E+07	1,35E+07	1,00E+00	3,93E+07	U
28	27	6	6	26	1,00E+00	2,54E+07	2,12E+07	9,88E+07	2,77E+07	U
29	27	23	6	24	1,69E+07	3,68E+07	2,68E+07	9,69E+07	3,18E+07	U
30	1	21	14	109	1,00E+00	8,16E+07	1,23E+07	1,00E+00	3,35E+07	U
31	12	23	15	20	1,50E+07	1,85E+07	8,38E+06	1,00E+00	9,80E+06	U
32	13	13	14	20	3,00E+07	1,09E+07	2,10E+07	1,00E+00	1,01E+07	D
33	7	49	13	90	1,06E+07	1,06E+07	1,06E+07	1,06E+07	1,06E+07	=
34	17	22	11	100	1,50E+07	1,50E+07	1,50E+07	1,50E+07	1,50E+07	=
35	18	6	6	24	1,00E+00	9,04E+07	3,05E+07	5,30E+06	5,33E+07	U
36	14	50	14	240	1,00E+07	1,00E+07	1,00E+07	1,00E+07	1,00E+07	=
37	2	33	14	120	1,00E+00	1,22E+07	1,69E+07	1,77E+07	2,39E+07	U
38	18	60	6	210	8,20E+06	8,20E+06	8,20E+06	8,20E+06	8,20E+06	=
39	17	50	14	162	1,00E+07	1,60E+06	2,04E+07	7,84E+06	2,65E+07	D
40	27	50	5	45	4,09E+07	4,09E+07	4,09E+07	4,09E+07	4,09E+07	=
41	8	57	6	27	6,61E+06	6,61E+06	6,61E+06	6,61E+06	6,61E+06	=
42	2	57	5	12	2,12E+07	2,12E+07	1,76E+07	1,00E+00	2,12E+07	=
43	1	16	12	104	1,00E+00	5,33E+07	1,44E+07	1,38E+07	2,28E+07	U
44	14	18	13	160	4,30E+07	5,62E+07	1,58E+07	7,09E+06	5,59E+07	U
45	14	23	6	27	3,00E+07	5,47E+07	1,99E+07	9,38E+06	2,97E+07	U
46	14	8	8	36	8,40E+07	8,40E+07	8,40E+07	8,40E+07	8,40E+07	=
47	14	30	14	33	2,40E+07	2,40E+07	2,40E+07	2,40E+07	2,40E+07	=
48	15	25	11	44	4,00E+07	4,00E+07	4,00E+07	4,00E+07	4,00E+07	=
49	11	42	6	67	1,00E+07	1,00E+07	1,00E+07	1,00E+00	1,00E+07	=
50	27	6	8	29	1,69E+07	1,69E+07	1,69E+07	1,69E+07	1,69E+07	=
51	27	51	6	62	1,27E+07	1,27E+07	1,27E+07	1,27E+07	1,27E+07	=
52	19	23	3	13	2,00E+07	1,06E+07	2,00E+07	1,00E+00	2,00E+07	D
53	14	28	8	36	1,00E+07	1,00E+07	1,00E+07	1,00E+07	1,00E+07	=
54	2	6	15	0	9,14E+07	1,74E+07	1,36E+07	4,56E+07	2,67E+07	D
55	39	49	8	31	7,00E+06	7,00E+06	7,00E+06	7,00E+06	7,00E+06	=
56	14	13	11	43	2,10E+07	2,10E+07	2,10E+07	7,18E+06	2,10E+07	=
57	16	18	11	70	1,64E+07	1,64E+07	1,64E+07	1,00E+00	1,64E+07	=
58	43	49	30	163	4,23E+06	4,23E+06	4,23E+06	4,23E+06	4,23E+06	=
59	5	23	3	0	1,44E+07	1,44E+07	1,44E+07	1,44E+07	1,44E+07	=
60	1	19	12	45	9,17E+06	9,17E+06	9,17E+06	9,17E+06	9,17E+06	=
61	27	23	9	22	1,27E+07	8,65E+07	5,85E+06	1,00E+00	2,14E+07	U
62	8	38	30	76	4,22E+07	4,22E+07	4,22E+07	4,22E+07	4,22E+07	=
63	5	57	15	32	4,56E+06	4,56E+06	4,56E+06	4,56E+06	4,56E+06	=
64	43	11	12	65	1,69E+06	1,69E+06	1,69E+06	1,69E+06	1,69E+06	=
65	35	57	15	150	4,93E+06	4,93E+06	4,93E+06	4,93E+06	4,93E+06	=
66	43	12	15	150	8,46E+06	8,46E+06	8,46E+06	8,46E+06	8,46E+06	=
67	14	9	15	420	1,10E+08	1,10E+08	1,10E+08	1,10E+08	1,10E+08	=
68	27	6	8	36	8,46E+06	8,46E+06	8,46E+06	8,46E+06	8,46E+06	=
69	39	10	13	59	5,08E+06	5,08E+06	5,08E+06	5,08E+06	5,08E+06	=
70	14	57	8	36	1,50E+07	1,50E+07	7,86E+06	1,50E+07	1,50E+07	=
71	15	6	7	15	3,00E+07	3,00E+07	2,83E+07	1,43E+07	2,74E+07	=
72	25	13	9	21	2,43E+07	2,43E+07	1,33E+07	2,43E+07	2,43E+07	=

Table 5.5: Detailed description of each facility (part2). The "**Type**" column indicates if a facility can be modified (**U**), only decreased (**D**) or unchanged (=).

Part III

Optimization software userguide

Chapter 6

BMO Userguide (V26.01.06)

6.1 Install

1- Decompression of the software

Windows: Launch the executable "*install.exe*".

The default install directory is "*C:/BMO*" and consider that the Matlab root path is "*C:/Matlab*".

In other cases the link installed in the windows desktop "**BMO.lnk**" must be modified (**Right mouse button/Property/Target**).

Example:

Default Link Target is:

"*C:/MATLAB/bin/win32/MATLAB.exe -r "addpath('C:/BMO');cd c:/BMO;*"

If Matlab root path is "*C:/MesOutils/Matlab*" an BMO install path "*C:/Prog*" Link target must be:

"*C:/MesOutils/Matlab/bin/win32/MATLAB.exe -r "addpath('C:/Prog/BMO'); cd C:/Prog/BMO;*"

Other systems: Decompress the program manually inside the desired repertory.

2- Modification of Matlab PATH :

Method 1 (permanent modification): In Matlab menu "FILE/Set Path.../" select "Add Folder" and add the directory where you have installed BMO.

Method 2 (temporary modification): Execute he following Matlab command: "*addpath(* BMO install directory *);*"

Example:

If BMO install directory is "*C:/Prog/BMO*", execute:

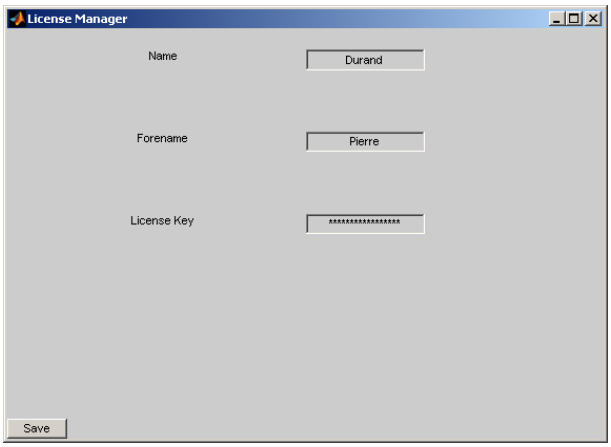
"*addpath(C:/Prog/BMO);*"

3- Launch the program: Click on the desktop icon (Windows only) or execute the Matlab command: "*bmo*".

WARNING: Workspace is cleared when bmo is executed!

4- Product registration: When BMO is launched for the first time a window "*License Manager*" appears and asks for your (*Name*), (*Forename*) and (*License Key*). Please fill

in those boxes and then press Save.



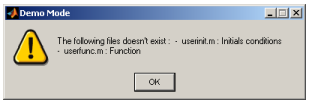
Normally, bmo is now working correctly.

6.2 Basic use

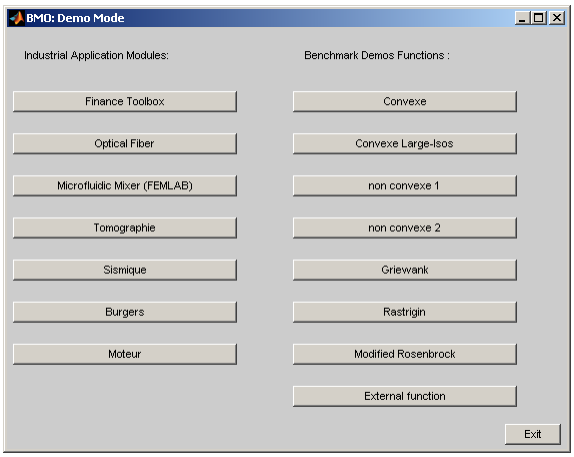
Step by step, we will describe the basic functionalities of the software presented through the use of a benchmark function.

6.2.1 Demonstration mode

This mode is started automatically when the software doesn't detect user files corresponding to a personal optimization problem (See section 6.4). A window will notify the mode activation and the missing files.

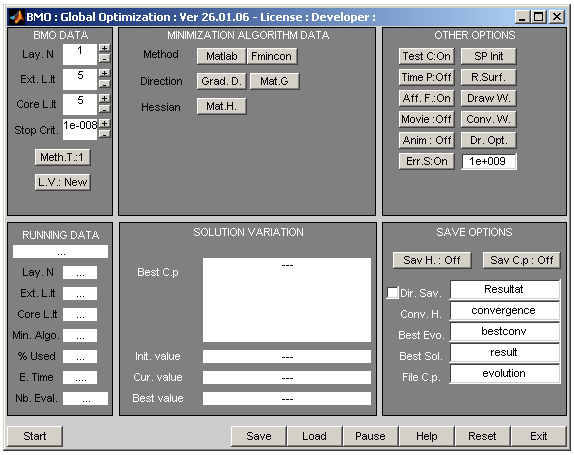


Launch bmo in the install directory. A window "*BMO: Demo Mode*" should appear. It present various benchmark functions and toolboxes developed during this work. Choose the Rastringin function.



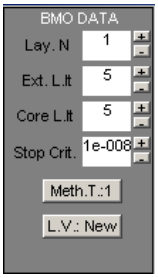
The main control panel should appear.

6.2.2 Main control panel



The main control panel is compound by six main zones:

Zone 1- BMO data: Correspond to the external layer algorithms (A_i or B_i introduced in chapter 2) data.



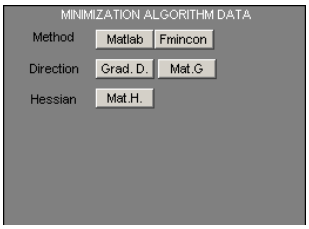
- **Level2**: Number of external layers (recommended 1).
- **Level1**: Iteration number of the external layer (recommended 5).
- **Level0**: Iteration number of the internal layer (A_1 or B_1) (recommended 5).
- **Stop Crit.**: Stopping criterion. It's based on the cost function reduction relatively to its initial value.
- **Meth.T**: Technique for choosing second initial condition of layers:

  -1: Point is chosen in a ball of center first initial point and with a radius proportional to the search space length.

  -2: Point is chosen randomly in the search space.

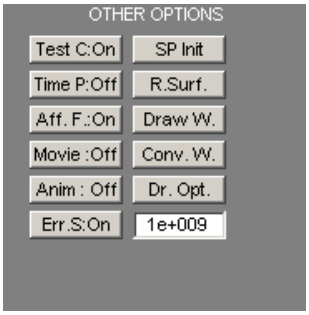
  -3: Only used when a genetic method is selected as core optimization method: The second population is created using a shooting method starting from each individual and looking in the direction of the best current point.

Zone 2- Minimization algorithm data: correspond to the core optimization algorithm data. The use of this box is detailed in section 6.2.3.



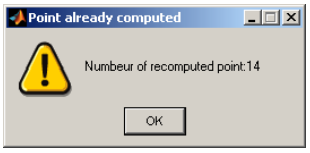
For the moment, we keep the default options. The "*fmincon*" Matlab function is used as core optimization method.

Zone 3- Other options: presents various options used during optimization process:



- **Test C.**: (recommended on)

  ◦ If "*on*": All the computed points are saved in the workspace in the matrix '*savalu*' and their value in the vector '*valu*'. At the end of the optimization they are saved into "*Computedpoints.csv*" in the save directory "*Dir.sav.*" defined in **zone 5**. The structure of those variables is described in *Advanced use* paragraph. Furthermore, if a point is recomputed the software take its previous value. A warning box will display the number of time a computed pointed has been considered.



  ◦ If "*Sv*": As previously computed points are saved but their functional value are reevaluated if they are considered various time.

  ◦ If "*off*": Computed points are not saved.

- **SP**: Starting point:

  ◦ If "*Init*": Optimization starts from the initial point defined in initial condition file (described in section 6.4).

  ◦ If "*Rand*": Optimization starts from an initial random point.

  ◦ If "*Opt*": If exist, optimization starts from the previous optimized point "*xoptg*" (described in section 6.4).

- **Time P.**:

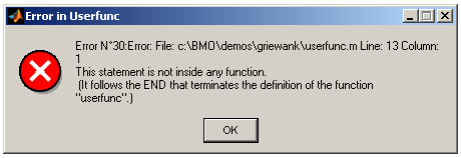
  ◦ If "*on*": After each computation a 0.1 sec pause is performed. This allows Matlab to refresh the graphical interface.

  ◦ If "*off*": Pause is disabled.

- **Err.S**: (recommended off)

  ◦ If "*on*": Software stop in case of functional evaluation error.

  ◦ If "*off*": During the functional evaluation if an error occurs the software doesn't stop. The functional value is set to the value defined in the box at the right of Err.S button (default 1e9). A error box will display the number and type of errors.



- **Aff F.**: Disabled if Matlab figure "*FIGUSER*" is not defined (See section 6.4).

  ◦ If "*on*": Graph, defined by user, is drawn at each iteration in "*FIGUSER*", except during gradient evaluation.

  ◦ If "*off*": Graph is not drawn and figure "*FIGUSER*" is hidden.

- **Movie**: Disabled if Matlab figure "*FIGUSER*" is hidden.

  ◦ If "*on*": A movie is saved (*avi* format) showing graph evolution at each iteration of the external layer algorithms. The movie is saved in the file and directory defined in **zone 5**.

  ◦ If "*off*": Movie is not saved..

- **Conv.W.**: Show/hide the "*Convergence Graph*" window. This tool is described in section 6.2.4.

- **Draw**: Show/hide the functional "*Drawing window*". This tool is described in section 6.2.4.

- **R. Surf**: Show/hide the response "*Surface drawing window*". This tool is described in section 6.2.4.

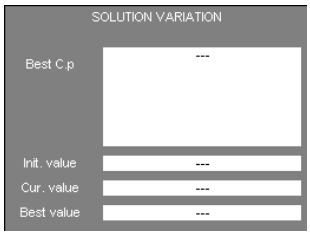
- **Draw Opt**: Show/hide drawing option toolbar in the "*Drawing window*".

- **Anim**:

  ◦ If "*on*": At each functional evaluation each graph is updated.

  ◦ Si "*off*": Some graphs are not updated until the end of the optimization.

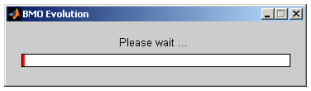
Zone 4- Solution variation: Display functional data during optimization:



- **Best C.P**: Display current best point.
- **Init. Value**: Display initial value of the function.
- **Cur. Value**: Display current value of the function.
- **Best value**: Display current best value of the function.

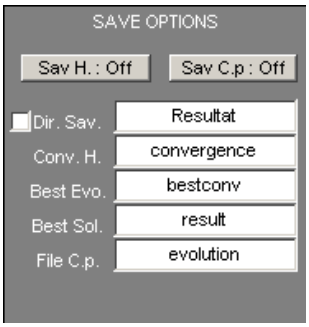
Zone 5- Running data: Display optimization evolution data:

- **Level 2**: Total external layer Iteration.
- **Level 1**: Current Layer Iteration.
- **Level 0**: First Layer Iteration.
- **Min Algo**: Core optimization Iteration.
- **% Used**: Total progress of the optimization. A progress barr "*BMO Evolution*" is also displayed.



- **E.Time**: Elapsed time since the beginning of the optimization.
- **NB. Eval.**: Number of functional evaluations.

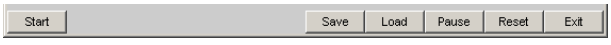
Zone 6- Save option: Output file options:



- **Sav H.**: (recommended on)
  - If "*on*": Files are saved after each functional evaluation.
  - If "*off*": Files are saved only at the end of the optimization
- **Sav C.P.**: (recommended All)
  - If "*All*": All computed points are saved in the file defined by "*File C.p.*" box.
  - If "*Best*": Best points are saved in the file defined by "*File C.p.*" box.
  - Si "*off*": No point is saved.
- **Dir.Sav.**: Name of the directory where file are saved Check the left box in order to set this name to the current date:Time. When you start optimization, if the directory already exist a dialog box will ask you to continue. If you choose "*Yes*" all files in this directory will be erased.

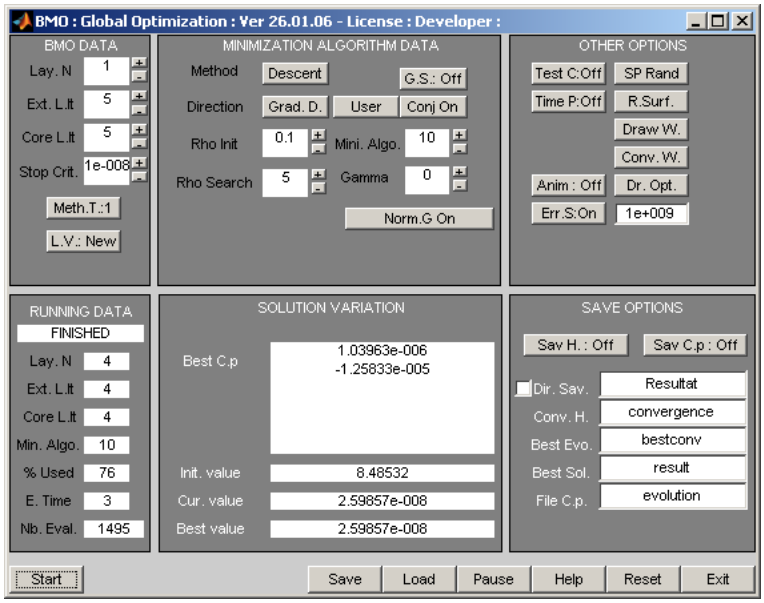
- **Conv.H.**: Name of the file (in txt format) where the evolution of the functional value is saved.
- **Best Evo.**: Name of the file (in txt format) where the evolution of the best functional value is saved.
- *Best Sol.*: Name of the files (in txt and csv formats) where the best point and its functional value is saved.
- *File C.p.*: Name of the files (in txt and csv formats) where computed points and their functional values are saved.

Zone 7- Menu Bar: Menu with the principal control commands:



- **Start**: Launch optimization algorithm. If an optimization file miss (for example you are not in the good directory) the button becomes "*Scan*", go to the correct directory and press it. If missing files are presents button becomes "*Start*".
- **Save**: Most optimization options are saved in the current directory.
- **Load**: Saved options are loaded.
- **Pause**: Software is paused. To restart it press enter.
- **Reset**: Reset the software.
  WARNING: Workspace is cleared !
- **Exit**: Software is closed.

Application: Now as we have presented the main control panel commands, we can launch our first optimization using the default software options. Press "**Start**". Normally, at the end of the optimization process zone 4 and 6 should be of the form:



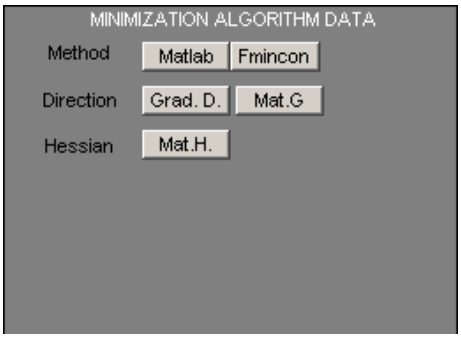
In this example the minimum found is $2e^{-8}$ at point $[1e-6, 1e-5]$ after 1500 functional evaluations and 7 seconds of computation. 76 % of the total iteration number has been performed.

6.2.3 Implemented core optimization algorithms

We are interested in this part by the different core algorithms and their options implemented in the software. This is modifiable in the "**zone 2**" of the main control panel.

There are four main classes of methods which can be selected pressing the method button.

• **Matlab** methods:



In this class we have implemented four optimization methods included in Matlab. You can read Matlab help file to have more explanations.

- **fmincon**:

  Direction: Choose the gradient direction. The "**Grad D.**" button is blocked.

  - **Mat G**: Matlab gradient
  - **Order 1**: First order gradient
  - **Order 2**: Second order gradient
  - **User**: User defined gradient (if it has been defined, see section 6.4 for more details).

  Furthermore depending on the selected gradient direction, you can choose to activate "**G.S.**" a simplified version of the gradient turning this button "*on*"|"*off*".

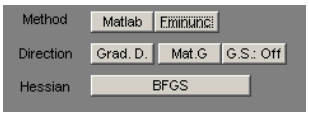
  Hessian: choose the Hessian definition:

  - **Mat GH.**: Matlab Hessian
  - **User**: User defined Hessian (if it has been defined).

- **Lsqnlin**:

  They are two main settings for this algorithm "**Gauss-Newton**" or "**Levenberg-Marquart**".

- **Fminunc**:



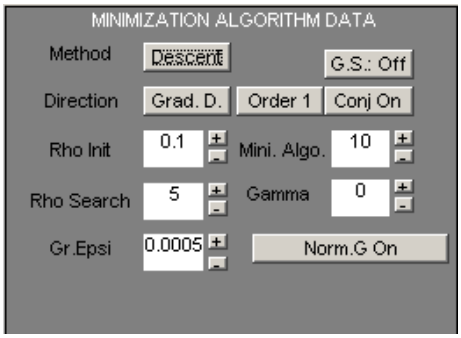
Direction: Same options as in "*fmincon*" case.

In addition to the choose of gradient direction there are three sub-methods proposed: "**BFGS**", "**Davidon-Fletcher-Powell**" and "**Steepest Descent**".

◦ **Fminsch**:

No option for this algorithm.

• **Descent** methods:



◦ **Direction**: Choose the descent direction:

Grad.D.: Use a gradient descent direction:

· **Mat G**: Matlab gradient

· **Order 1**: First order gradient

· **Order 2**: Second order gradient

· **User**: User defined gradient (if it has been defined).

Furthermore depending on the selected gradient direction, you can choose to activate:

· "**Conj**": use a conjugate direction: "*on*"|"*off*".

· "**Norm**": Normalize the direction of the gradient: "*on*"|"*off*".

· "**G.S.**": a simplified version of the gradient: "*on*"|"*off*".

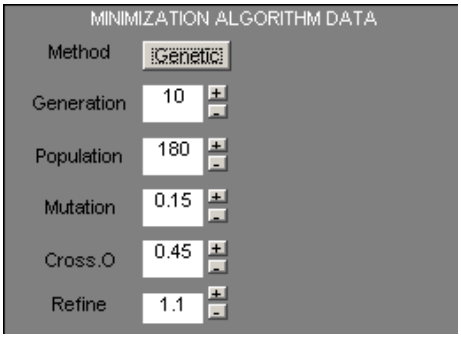
· "**Gr.Epsi**": In case of order 1 or 2 gradient, choose the epsilon step for finite difference (normalized with the search space length).

Rand.D.: Use a random descent direction:

Dir. Search option allows to choose the maximum number of directions tested in order to have a suitable descent direction.

- **Rho init**: Initial value of ρ in the method.
- **Rho search**: Iteration number of the dichotomy search method used in order to determine an appropriate ρ.
- **Mini Algo.**: Iteration number of the descent algorithm.
- **Gamma**: η coefficient. If $\eta = 0$ the considered system is a typically first order system. Else it's a second order dynamical system coming from discretization of the "**Heavy ball equation**".

• **Genetic** methods:



- **Generation**: Generation number.
- **Population**: Population size.
- **Mutation**: Probability of mutation. it's a number between $[0, 1]$.
- **Cross.O**: Probability of Cross Over. it's a number between $[0, 1]$.
- **Refine**: Refinement parameter included in the distance of hamming (See chapter 1).

• **User** method:

This is an option detailed in **Advanced Use**. This option appears only if useropt.m has been created.

Application: Increase "**level1**" and "**level0**" to 10 and select "**Meth.T.:2**" in **zone 1**. You can play with those options and change the method in order to see the performances of each one on Rastringin function.

6.2.4 Default graphical tools

We will now see how to use the various graphical tools implemented in BMO. All those windows contains the classical Matlab menu in order to modify and save them.

Convergence Windows

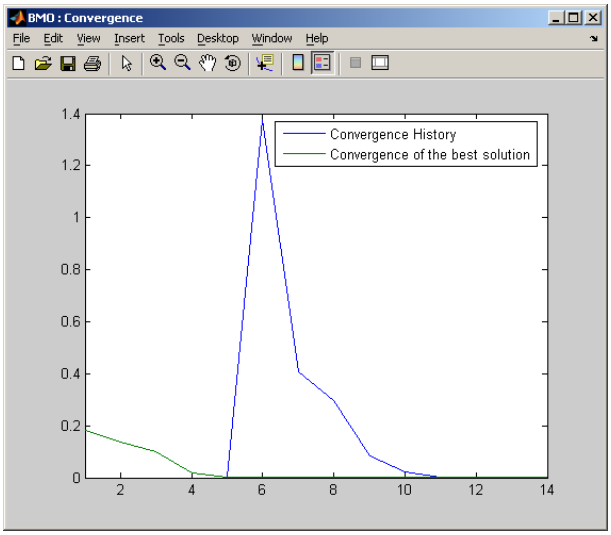
Click on the "**Conv W.**" present in **zone 3**. A new window entitled "**BMO: Convergence**" should appear.

This windows display two graphs:

- The "**Convergence History**": Blue line corresponds to the convergence history of the cost function during optimization.

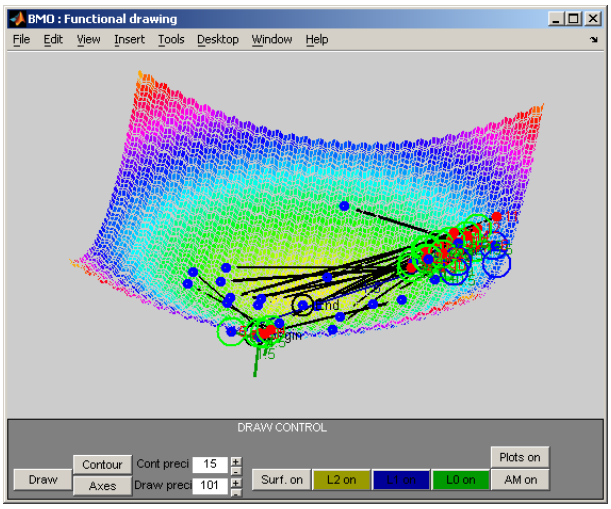
- The "**Convergence of the best element**": Green line corresponds to the evolution of the cost function value of the best element found during optimization.

The option "**Anim**" is active for this window.



Functional drawing

Click on the "**Draw W.**" present in **zone 3**. A new window entitled "**BMO: Functional drawing**" must appear.



Depending on the dimension of the problem, various options are displayed:

- **Draw**: Drawing the cost function.
- **Axes**: Show/Hide figure axes.
- **Contour**: Show/Hide isocontour of the function.
- **Draw preci**: Selected the mesh precision for the cost function graph.
- **Cont preci**: Number of isocontour lines.

Other options are applicable in order to show the graphical evolution of the optimization algorithm:

- **Surf**: Show/Hide the functional graph.
- **L2**: Show/Hidea the yellow line showing the evolution of the starting condition of external layer.
- **L1**: Show/Hide the blue line showing the evolution of the starting condition of internal layer.
- **L0**: Show/Hide the green line showing the evolution of the starting condition of the core algorithm.
- **AM**: Show/Hide the core algorithm trajectory evolution with a black line.
- **Plots**: Show/Hide the plots showing the various initial and final points for each core algorithm.

For each previous element a colored number indicate its order of apparition.

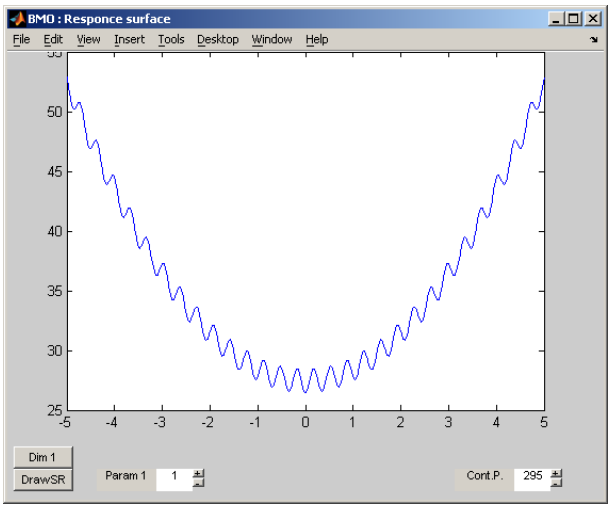
The option "**Dr. Opt.**" in **zone 3** is active for this figure and allows to Show/Hide the draw control menu.

The option "**Anim**" in **zone 3** is active for this windows.

Application: Open convergence and functional drawing Windows. Turn "**Anim**" on. Start an optimization.

Response surface drawing

Click on the "**R. Surf.**" present in **zone 3**. A window entitled "**BMO: Response surface**" appears.



Depending on the dimension of the problem, options are displayed:

- **Dim**: Choose the dimension of the drawing
- **Draw SR**: Draw the surface of response of the cost function.
- **Param 1**: Choose the first dimension to be studied.
- **Param 2**: Choose the second dimension to be studied.
- **Param 3**: Choose the third dimension to be studied.

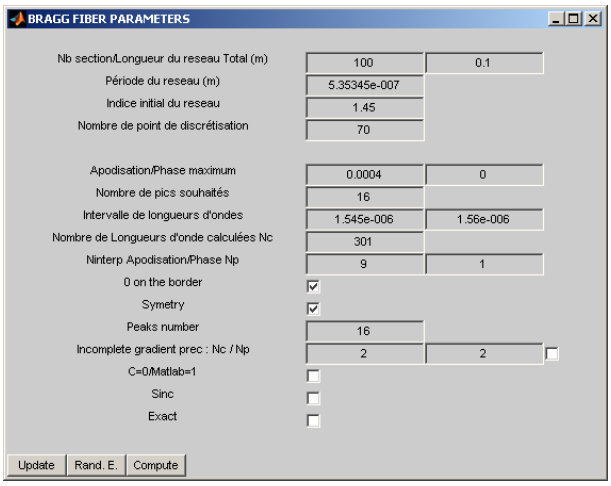
The default value for fixed dimension is set to the minimal value of the boundary. This can me modified into the "*drawsr.m*" file.

6.3 Implemented Toolboxes

Here we will describe the toolboxes developed during this work. First you need to launches BMO in **demonstration mode** (see section 6.2.1).

6.3.1 Optical Multichannel design

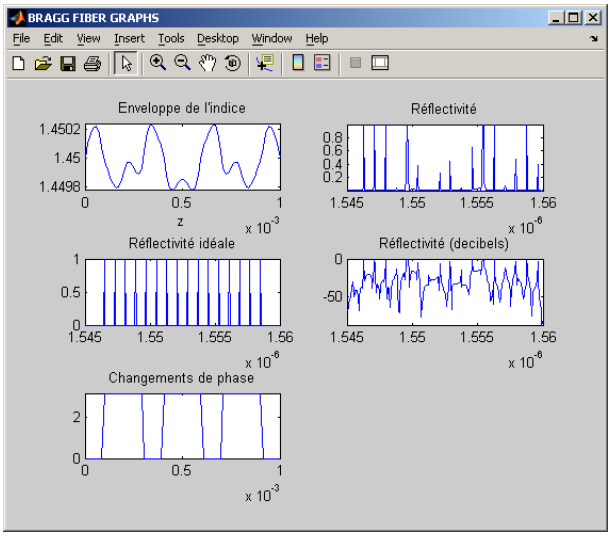
This problem is presented in Chapter 3. When the toolbox is launched the following control panel appears:



- **Nb Section**: Sample number in the filter.
- **Longueur du reseau**: Length of the total fiber in meter.
- **Période du réseau**: Grating period in meter.
- **Indice initial du réseau**: Initial core fiber refractive index.
- **Nombre de points de discrétisation**: Number of discretization points of one sample.
- **Apodisation/phase maximum**: Maximum variation of apodisation (left) and phase (right) in the grating.
- **Nombre de pics souhaités**: Number of desired peaks.
- **Intervalle de longueurs d'onde**: Minimal (left) and maximal(right) wavelength where the spectrum is computed.
- **Nombre de longueurs d'onde calculées**: Number of discretization points of the wavelength interval.
- **Ninterp Apodisation/Phase Np**: Number of interpolation points for apodisation (left) and phase (right) generation.
- **0 on the border**: If checked fix the apodisation to 0 on the border of each sample.
- **Symmetry**: If checked generated profiles are symmetric.
- **Incomplete gradient prec: NC/NP**: If checked divide the number of discretization points for wavelength (left) and apodisation (right) by the input value during sensitivity evaluation in order to compute a simplified gradient.
- **C=0/Matlab=1**: If checked the Matlab solver for transfer matrix method is used (slow). Else a C++ solver is considered (fast).

- **Sinc**: If checked the considered apodization has a Sinc-profile.
- **Exact**: If checked the considered apodization has the SD2A optimized profile .

At each evaluation of the functional the following graphs are drawn in window "**BRAGG FIBER GRAPHS**":



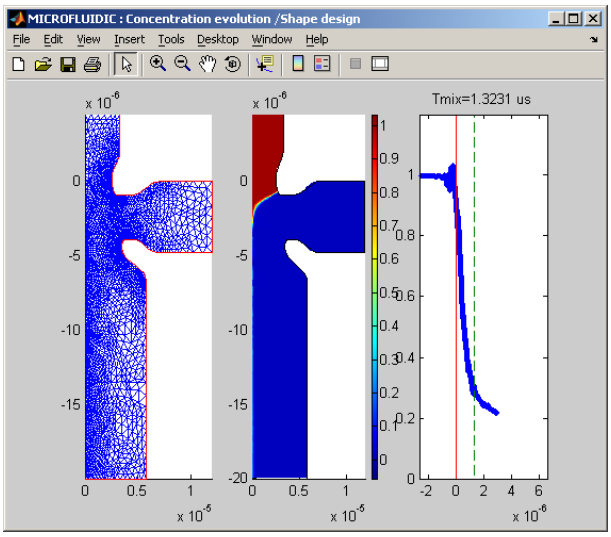
From left to right and top to bottom:

- Apodisation profile
- Associated spectrum reflexivity
- Perfect desired spectrum reflexivity
- Associated spectrum in Decibel
- Phase Shifts

6.3.2 Knight Micro-Mixer design

This toolbox is associated to the problem defined in Chapter 4. First, you need to have installed *Femlab 3.1* on your computer. Launch the "**Femlab with Matlab**" mode and then execute bmo.

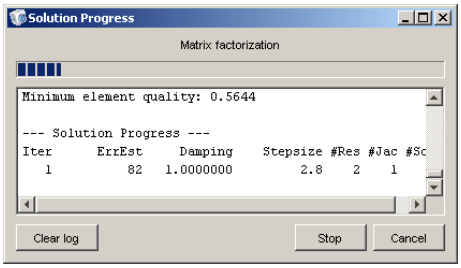
A Matlab figure appears:



Three sub-figures are displayed at each evaluation of a specific shape:

- The **left** sub-figure corresponds to the mesh grid.
- The **center** sub-figure corresponds to the surface of concentration.
- The **right** sub-figure corresponds to the concentration evolution through the symmetry line.

At each evaluation a Femlab solver progress bar is also visible:



Geometry modification: The parameterization of the shape is defined in "*usersolver.m*" file from line 22 to 94. This part should be modified only by and *experimented user* in order to modify the admissible shapes. This could be also done using Femlab software by exporting the new geometry into a M file. This technique is described in section 6.4.

6.3.3 Portfolio Manager

This problem is described in Chapter 5. Two version have been developed. A stand alone version in C++ language and a Matlab version. The stand alone version contains only the HSGA algorithm and has been developed for the BNP-Paribas Asset Management Team.

C++ Version

Installation

You should found inside the install directory the following folders and files:

- **Root Directory:** Contains main program files:
  - CLO2optimizer.exe: Launch the program.
  - nagc.dll: Nag library.
  - dataago.csv: Optimization algorithm's data:

Number of ICLOs +1 **!!!MUST BE SPECIFIED MANUALLY!!!**
Monte-Carlo iteration number
Is ICLOs and Master CLO have the same Maturity ? 1=Yes/0=No
Is ICLOs have the same Maturity ? 1=Yes/0=No
Bucket number for loss density function evaluation
Are ABS activated ? (disabled)
disabled
disabled
Use CDS Spread ? (disabled)
disabled
disabled
disabled
disabled
disabled
Confidence number
Population size
Generation number
Shooting method number

  - probdef.csv: Problem definition data:

Nominal function option (1=Max,0=Inact,-1=Min)
Income function option (1=Max,0=Inact,-1=Min)
Risk measure function option (1=Max,0=Inact,-1=Min)
RAROC Loss function option (1=Max,0=Inact,-1=Min)
CEVA function option (1=Max,0=Inact,-1=Min)
Unexpected Loss function option (1=Max,0=Inact,-1=Min)
Nominal constraint option
Income constraint option
Minimal Nominal constraint option
Risk measure constraint option
RAROC Loss constraint option
CEVA constraint option
Unexpected Loss constraint option
Coefficient in CEVA evaluation
Risk type (1=VaR/2=ES)

- opav.csv: Advanced user optimization data:

  !!! Must be modified MANUALLY !!!

Parent selection type (0=Cost independent/1=Cost dependent)
Mutation type (0=Vector type/1=case-by-case type)
Avoid incest (0=off/1=on)
Cross-Over probability (over 1)
Mutation probability (over 1)
case-by-case Mutation probability (over 1)
Vector type Mutation refinement ([1,2])

- **CorrData**: Contains correlation coefficient data:
  - *Betaindustry.csv*: This file stores the betas associated with the 65 industries. The 15th column contains the variance of the i^{th} industry:

	1 st Factor beta	...	14 th Factor beta	Variance
i th*industry* double	double	...	double	double

  - *BetaPays.csv*: This file stores the betas associated with the 45 countries. The 15th column contains the variance of the i^{th} country:

	1 st Factor beta	...	14 th Factor beta	Variance
i th*country* double	double	...	double	double

  - *TableRatingCEDF10ans.csv*: This file stores the internal cumulated default probabilities associated with a given credit class:

	PD ith*year*	...
i th*creditclass*	double	...

  - *RiskFreeRate.csv*: This file stores the risk free rate term structure:

	Risk free rate
i th*maturity*	double

  - *MarketCEDF.csv*: Contains all the information concerning the market cumulated default probabilities associated with each facility. Each row contains the information for a given facility. These CEDF are stripped from the market CDS quotations when they are available. A flat term structure is assumed if quotes are not available after 5 years:

Market CEDF 1 year	...	Market CEDF 10 years
double	...	double

  - *MarketCEDFABS.csv*: As previously we store all the information concerning the cumulated default probabilities associated with each stand alone asset:

Market CEDF 1 year	...	Market CEDF 10 years
double	...	double

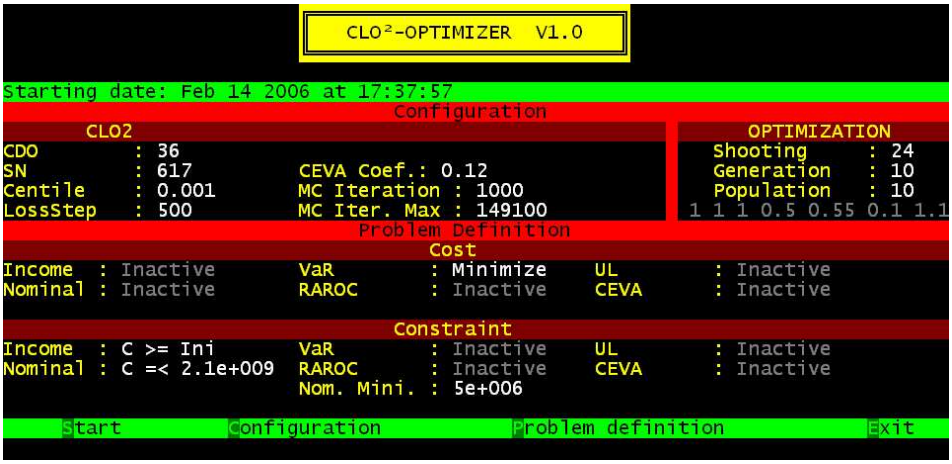
- **UserInput** folder: Contains initial portfolio data.

- **Output** folder: Contains software output files.
- **Intput** folder: Contains temporary optimization files (suppressed when exiting software).

A detailed structure description of Input/Output files is given in section 6.3.3.

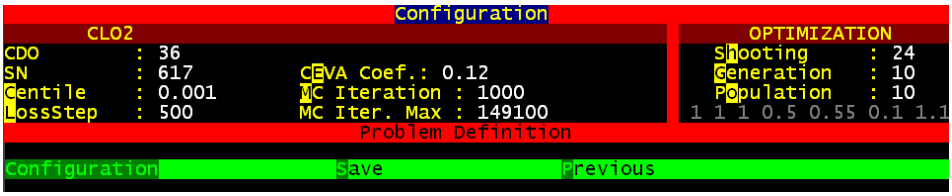
User Interface

When launching CLO2Optimizer.exe, the software displays the following control interface:



Using the mouse or Keyboard shortcuts (Highlight-letters), you can Navigate into various sub-menus in order to configure the software or launch the optimization process:

1- 'C'-Configuration: Allows to modify the CLO^2 internal analyzer and optimization configuration:



CLO^2 Options:

- **'C'-Centile**: Confidence used for considered risk measure computation.
- **'E'-CEVA Coef.**: Coefficient α used for CEVA computation.
- **'L'-LossStep**: Number of buckets used to compute Loss density.
- **'M'-MC Iteration**: Scenario number for Monte-Carlo Simulation. This number is bounded by 'MC Iter. Max' which depends on available computer memory.

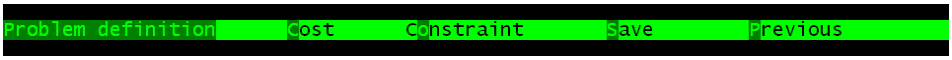
Optimization Options:

- **'H'-Shooting**: Iteration number for GA initial population improvement.
- **'G'-Generation**: Generation number for internal GA.
- **'O'-Population**: Population size for internal GA.

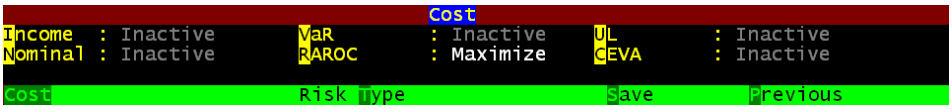
General Menu:

- **'S'-Save**: Save options into '*dataago.csv*' file.
- **'P'-Previous**: Return to main menu.

2- 'P'-Problem Definition: Configure optimization problem:



'C'-Cost: Define cost function to be minimized:



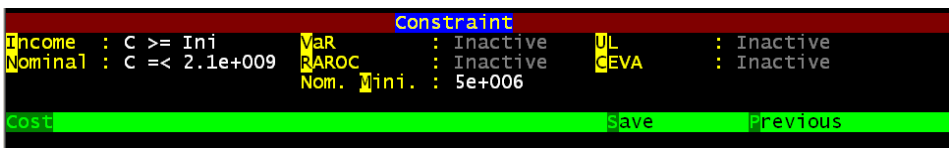
For each portfolio characteristic: **'I'-Income**, **'N'-Nominal** (Total nominal) , **'V'-Var**/ **Var-ES** (Value at risk/ Expected Shortfall) , **'R'-RAROC** , **'U'-UL** (Unexpected Loss) , **'C'-CEVA** there are three choices:

- **Inactive**: characteristic inactive in cost function.
- **Maximize**: characteristic added to cost function in order to be maximized.
- **Minimize**: characteristic added to cost function in order to be minimized.

General Menu:

- **'T'-Risk Type**: Select VaR or Expected-Shortfall (VAR-ES) as Risk measure.
- **'S'-Save**: Save options into '*probdef.csv*' file.
- **'P'-Previous**: Return to Problem Definition menu.

'O'-Consraint: Define constraints to be respected during optimization problem:



For each portfolio characteristic: **'I'-Income**, **'N'-Nominal**, **'V'-Var**/ **Var-ES**, **'R'-RAROC**, **'U'-UL** (Unexpected Loss), **'C'-CEVA** you can enter one of those five values:

- **0**: characteristic is not a constraint.

- **1**: characteristic has to be superior to its initial value.
- **-1**: characteristic has to be inferior to its initial value.
- **>1**: characteristic has to be superior to specified number.
- **<-1**: characteristic has to be inferior to absolute value of specified number.

'M'-Nom. Mini.: If a product has a nominal inferior to this value and the product nominal can decrease, its nominal is set to 0.

General Menu:

- **'S'-Save**: Save options into '*probdef.csv*' file.
- **'P'-Previous**: Return to Problem Definition menu.

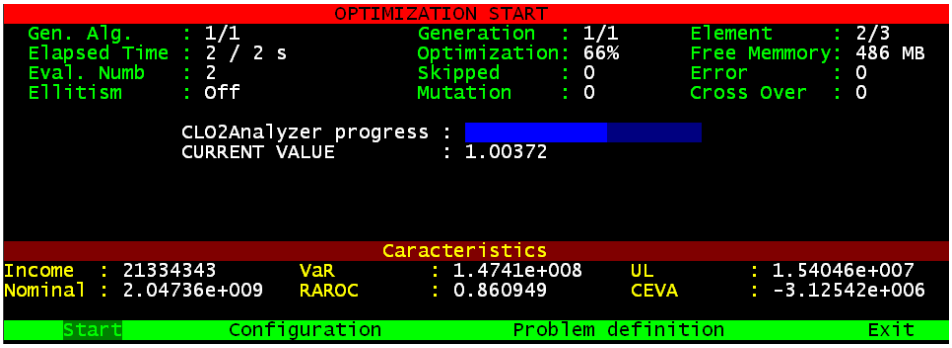
'S'-Save: Save options into '*dataago.csv*' file.

'P'-Previous: Return to main menu.

3- 'S'-Start: Launch optimization:

The algorithm first computes the initial portfolio characteristics and generate default times:

Then main optimization loop starts:



The software display the following information:

- **Gen. Alg.**: Shooting iteration / Total iteration number.
- **Generation**: Generation iteration / Total iteration number.
- **Elapsed time**: Elapsed time / evaluated total time.
- **Optimization**: Optimization progress in percent.
- **Free Memory**: Available physical memory.

- **Eval. Numb.**: Number of cost function evaluations.
- **Skipped**: Number of skipped evaluations due to recomputed points.
- **Error**: Number of points out of the constraint boundary.
- **Characteristics**: Characteristics of the current computed point.

At the end of the optimization process, OUTPUT files are saved (see section 6.3.3 for more details) and you can return on the main menu (**'Y'**) or close the software (**'N'**).

Input/Output Files

Input In order to perform an optimization:
-**YOU MUST SPECIFY MANUALLY** into '*dataago.csv*' in first row the number of ICLOs +1.

you must provide the following files inside the **UserInput** directory:
- '*Datafacility.csv*': Contains all the information concerning the assets included in the ICLOs and Single-names. The information are given at the client level to be in line with the asset correlation model of KMV. Each facility can possibly admits a maturity which does not coincide with the maturity of the transaction. You need to put '1' if the facility k does not belong to ICLO j. **Warning**: Don't let empty cells:

Facility Index	Client Index	LGD	R^2	Country Index	Industry Index
integer	integer	double	double	int	int
Rating Index	Nominal CLO^1	...	Nominal CLO^n	SN Nominal	Maturity
int	double	double	double	double	double

- '*DataABS.csv*': This file stores all the information concerning the standalone assets included in the portfolio of the master CLO:

Facility Index	Client Index	Nominal	LGD	R^2
integer	integer	double	double	double
Country Index	Industry Index	Rating Index	Maturity	
int	int	int	double	

- '*cododata.csv*': Contains ICLOs rating and LGD necessary for EC computation:

CLO Index	Rating Index	LGD
int	int	double

- '*sndata.csv*': Contains SN region index for EC computation:

SN 1 Region Index
⋮

- '*TranchingInnerDOs.csv*': This file stores all the information concerning the cumulated nominal of each tranche included in each ICLO. Each row contains the information for a given tranche. The first row contains the information related to the equity piece. There is a maximum of 10 tranches per CLO. All ten tranches row should be filled. Unused tranches row should be equal to the ICLO nominal. The last row contains the information of the most senior tranche:

Tranche Nominal CLO 1	...	Tranche Nominal CLO n
tranches 1	...	double
⋮	⋱	⋮
tranche 10	...	double

- '*TranchingMasterCDO.csv*': This file stores all the information concerning the cumulated nominal of each tranche included in the Master CLO. Each row contains the information for a given tranche. The first row contains the information related to the equity piece. The last row contains the information of the most senior tranche:

Tranche Nominal CLO
double
⋮
double

- '*maturity.csv*': Contains data relative to maturity and desired tranche of each Master/ICLOs. First row contains the starting date. Second to last row contains the Master/ICLOs maturity date and desired tranche number:

Starting year	Starting month	Starting day	
Master CLO year	Master CLO month	Master CLO day	Master CLO tranche
ICLO 1 year	ICLO 1 month	ICLO 1 day	ICLO 1 tranche
int	int	int	int
⋮	⋮	⋮	⋮

- '*SpreadCDO.csv*'/'*Spread ABS.csv*'/'*Spread SN.csv*': Contains old and current spreads of ICLOs and Single-Names (included in '*DataFaclity.csv*' / '*DataABS.csv*') and their optimization evolution (0=can't move, 1=can decrease/increase, 2=can only decrease):

old spread	parameter type	new spread
int	int	int
⋮	⋮	⋮

Output During the optimization process the following files are created into the **Output** directory:
- '*vinit.csv*' / '*xmin.csv*' / '*xmax.csv*' / '*x.csv*' / '*solution.csv*': In those files we store the initial / lower bound / upper bound / current / optimized nominal for each portfolio parameter:

Nominal CLO 1
double
⋮
Nominal SN 1
⋮

- '*allcalc.csv*'/'*allcalc-transpose.csv*'/'*logout.csv*': Contain all the computed portfolio associated to their characteristic value :

Initial portfolio	computed portfolio 1	...
cost function value	double	...
Risk measure	double	...
RAROC	double	...
UL	double	...
CEVA	double	...
Income	double	...
Total Nominal	double	...
CLO 1 Nominal	double	...
⋮	⋮	⋮
SN 1 Nominal	double	...
⋮	⋮	⋮

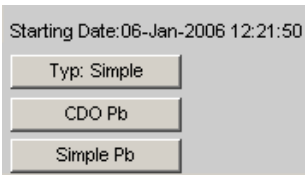
'*allcalc-transpose.csv*' and '*logout.csv*'(Real-Time) is the transposed of previous file.
- '*Solution-detailed.csv*' and Excel file: Show the evolution of parameters and characteristics between initial and optimized portfolios.
- '*convergence.csv*': Convergence history of the optimization process.

Matlab version

This Toolbox is compound by three control panels:

Problem Selection panel

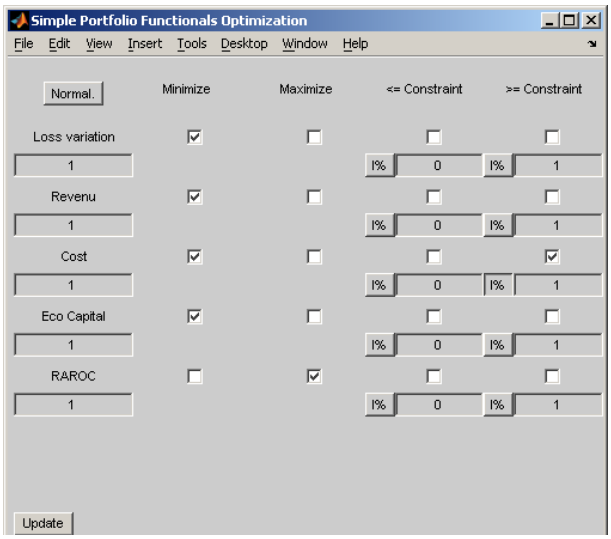
This windows allows to choose the portfolio modeling:



- **CDO Pb**: CDO^2 portfolio modeling problem.
- **Simple Pb**: Simple portfolio modeling problem.

Constraints and Target panel

Select the portfolio characteristics and constraints



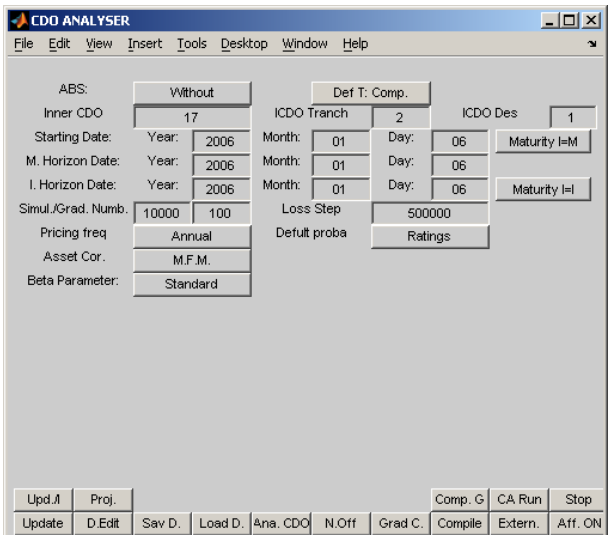
- **Normal**: If checked, portfolio characteristics are normalized in the cost function.

For each characteristic (**Loss Variation**, **Income**, **Nominal**, **Economic Capital**, **RAROC**) we have the following options:

- **First box**: Enter here the initial value. It's used for normalization.
- **Minimize**: Check this box if you want to minimize this characteristic.
- **Maximize**: Check this box if you want to maximize this characteristic.
- **<= Constraint**: Check this box if the characteristic is considered as an upper boundary constraint. Enter the boundary value in the **bottom box**.
- **>= Constraint**: Check this box if the characteristic is considered as an lower boundary constraint. Enter the boundary value in the **bottom box**.
- **%I**: Check this box if you want to use the initial portfolio's characteristic value used as boundary.
- **Update**: Update all previous options.

CDO^2 control panel

This windows permits to define problems the CDO^2 structure:



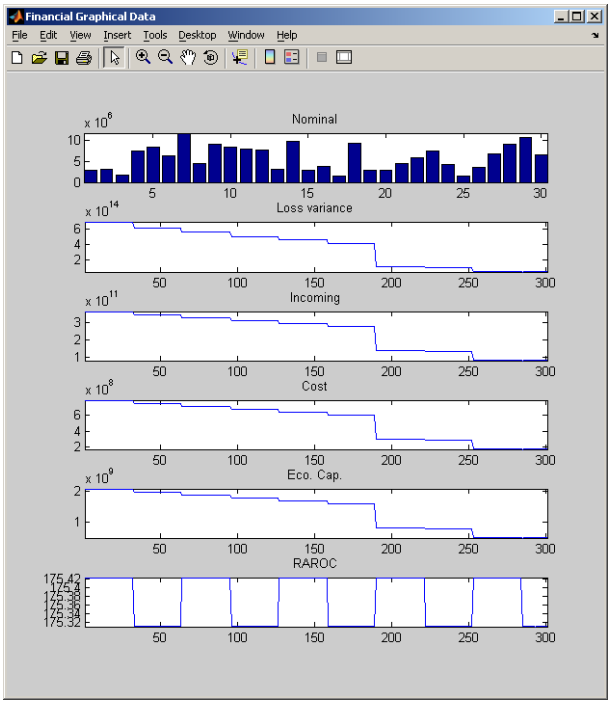
- **SN**: Single-Names options:
  - **Without**: No Single-Name considered.
  - **With**: Single-names in "*dataABS.csv*" are considered as parameters.
  - **With/No P**: Single-names in "*dataABS.csv*" are constants.
  - **Included**: Single-Names in the last row of "*dataFacility.csv*" are considered as parameters, "*dataABS.csv*" file is ignored.
  - **Included/W**: Single-names in the last row of "*dataFacility.csv*" are considered as parameters, "*dataABS.csv*" stocks independent CDOs.
- **Def. T**: Default Times options:
  - **Comp**: Computed at each functional evaluation.
  - **C&S**: Computed and then Saved in "*Defaultimes.csv*".
  - **Load**: Are Loaded from "*Defaultimes.csv*".

- **Inner CDO**: ICDOs number.
- **ICDO Tranch**: ICDOs'default tranche.
- **MCDO Des**: Master CDO's desired tranche.
- **Date**: Dates used to compute Inner CDOs and master CDO's maturity.
- **Maturity I=M**: Inner CDOs and Master CDO have the same maturity.
- **Maturity I=I**: Inner CDOs have the same maturity.
- **D. Edit**: Generate and edit manually the maturity matrix.
- **Sav D.**: Save the maturity matrix into "*maturity.csv*".

• **Load D.**: Load the maturity matrix from "*maturity.csv*".
• **Simul./Grad. Numb.**: Iteration number for the Monte-Carlo algorithm (*left*) and gradient evaluation (*right*).
• **Simul./Grad. LossStep**: Number of Global Loss sections for the Monte-Carlo algorithm ((*left*) and gradient evaluation (*right*).
• **Centile**: If "*Default*" option is selected, the default centile value associated to maturity is considered for VaR evaluation. If it's set to "*Custom*" the centile value is set to the user's value defined in right box.
• **Aff.**: If "on" display informations during each portfolio evaluation.
• **Proj./Barr.**: In first case the portfolio is projected in admissible space else a barrier function is added to the cost function.
• **CA Run**: Launch the CLOAnalyzer as a background task.
• **Stop**: Stop the CLOAnalyzer.
• **Update**: Update Data and evaluate initial portfolio.
• **Upd./l**: Update Data.

Graphical interface

- **Simple modeling problem**:

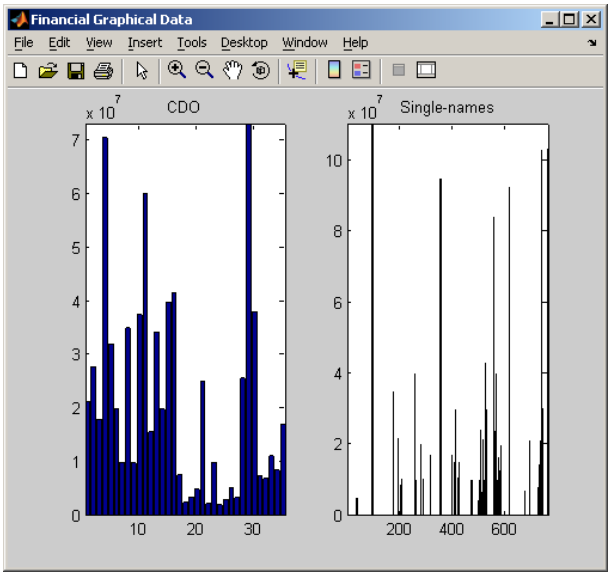


From top to bottom:

- **Nominal**: Nominal of the facilities.
- **Loss Volatility**: Evolution of the loss volatility of the portfolio.
- **Incomes**: Evolution of the portfolio's income.

- **Cost**: Evolution of the total nominal of the portfolio.
- **EC**: Evolution of the economic Capital of the portfolio.

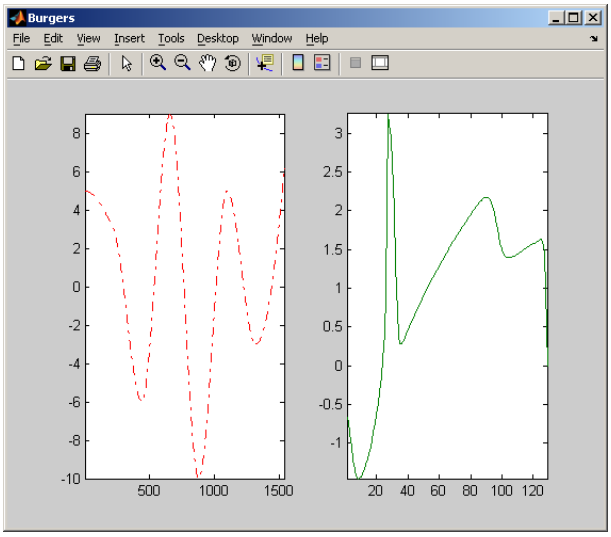
• CLO^2 **modeling problem**:



- **CDO**: Nominal of the ICDOs. First: CDOs in "*Datafacility.csv*", Last CDOs in "*DataABS.csv*".
- **Single-Names**: Nominals corresponding to the Singles Names included in the last row of the CDO in "*Datafacility.csv*".

6.3.4 Burgers inverse problem

This problem is presented in Annex A. There is a main figure:



In this figure the left-graph represents the time-control and the right-graph the associated Burgers solution.

Parameter α, corresponding to the weight coefficient of the control term of the cost function, can be modified into "*donnee.m*" by changing the value of "*y(36)*".

The target solution for burgers equation is defined in "*userinit.m*" by variable "*exact*". It's a vector of dimension defined by "*y(34)*" in "*donnee.m*". You can construct it using function "*j4(xminit,ym)*" where "*ym=zeros(pt)*" and "*xminit*" correspond to a given time-control represented by a "*xp*"-vector.

6.4 Advanced Use

6.4.1 Program Architecture

The software code could be divided in two parts:

General optimization files

- Launch the software: **bmo.m**
- Software initialization: **init.m**, **bmor.m**
- License management: **Liver.m**, **License.m**, **Licensec.m**, **Licenscecreator.m**, **Licgo.m**.

  WARNING: THIS PART MUSN'T BE MODIFIED BY USER.
- Graphical User Interface:
  - **guim.m**: Build general Graphical User Interface.
  - **draw.m**: Build Functional Draw control panel.
  - **drawsr.m**: Build Response Surface control panel.
  - **action.m**: Define button's effect.
  - **exiting.m**: Define exiting sub-script.
  - **pausing.m**: Launch pause.
  - **saveo.m**: Launch option-save sub-script.
  - **Loado.m**: Launch option-load sub-script.
- Multi-layer optimization code: **bmot.m**, **couche0.m**, .**couche1.m**, **couche2.m**, **affresult.m**, **timec.m**, **pour.m**.
- descent core optimization algorithm: **gradopt.m**,**gradient.m**, **rhodicho.m**, **rdmd.m**, **miniopt.m**.
- genetic core optimization algorithm: **initag.m**, **AG.m**, **select2.m**, **funcbraggech.m**, **randselect.m**.

User files

- **Userinit.m**: Define initial conditions of the problem.
- **Userfunc.m**: Function to be optimized.
- **Usergrad.m** (*optional*): User gradient method to be used instead of implemented gradient methods.
- **Useropt.m** (*optional*): User core optimization method.
- **Userinterface.m** (*optional*): Additional graphical interface launched at the beginning.
- **Useraffich.m** (*optional*): Output to be displayed at the end of the optimization algorithm.
- **Userexiting.m** (*optional*): Script executed when exiting the software.

A general scheme, showing how those sub-scripts are linked, is presented in Figure 6.1.

Main variables definition

When the program is running various variables are created into the Matlab Workspace. Here we describe the principal ones:

- **ndim:** The dimension of the problem. It's an Integer number.
- **vinit:** The given initial condition. If it's not given, only Random Initial condition option described in section 6.2.1 will be activated. It's a **ndim**-dimension vector of the form:

$$\begin{vmatrix} \text{Initial value of parameter 1} \\ \vdots \\ \text{Initial value of parameter } \mathbf{ndim} \end{vmatrix}$$

- **xmax:** The upper boundary of the problem. It's a **ndim**-dimension vector of the form:

$$\begin{vmatrix} \text{maximum value of parameter 1} \\ \vdots \\ \text{maximum value of parameter } \mathbf{ndim} \end{vmatrix}$$

- **xmin:** The lower boundary of the problem. It's a **ndim**-dimension vector of the form:

$$\begin{vmatrix} \text{minimum value of parameter 1} \\ \vdots \\ \text{minimum value of parameter } \mathbf{ndim} \end{vmatrix}$$

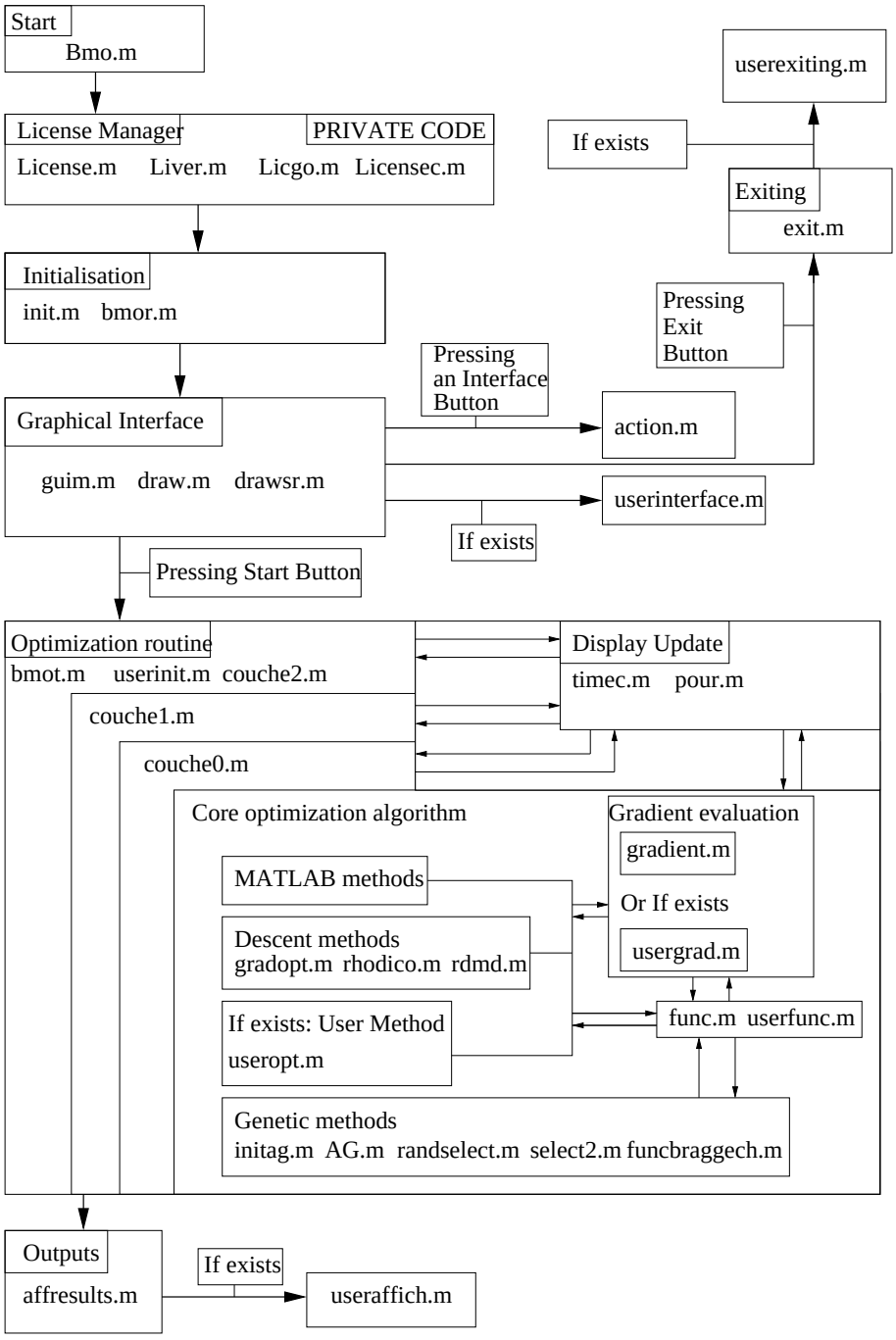


Figure 6.1: General software scheme

- **xoptg:** Best solution found during optimization process. It's a **ndim**-dimension vector of the form:

$$\left|\begin{array}{c} \text{Optimal value of parameter 1} \\ \vdots \\ \text{Optimal value of parameter } \mathbf{ndim} \end{array}\right|$$

- **costsaveming:** Functional value of **xoptg**.
- **evalu**: Number of functional evaluation. It's an Integer number.
- **timegui:** Computational time during optimization. It's a Real number.
- **savalu:** Elements computed during optimization process. It's a **ndim**×**evalu**-matrix of the form:

$$\left|\begin{array}{ccc} \text{First value of parameter 1} & \cdots & \text{First value of parameter } \mathbf{ndim} \\ \vdots & \ddots & \vdots \\ \mathbf{evalu}^{Th} \text{ value of parameter 1} & \cdots & \mathbf{evalu}^{Th} \text{ value of parameter } \mathbf{ndim} \end{array}\right|$$

- **valu:** Functional value of all computed points during optimization process. It's a **evalu**-dimension vector of the form:

$$\left|\begin{array}{c} \text{Value of first point.} \\ \vdots \\ \text{Value of } \mathbf{evalu}^{Th} \text{ point} \end{array}\right|$$

- **conv:** Functional value history of the local minimum found at each iteration of core-optimization method. It's a **iterc**-dimension vector of the form:

$$\left|\begin{array}{c} \text{Value of first local minimum} \\ \vdots \\ \text{Value of } \mathbf{iterc}^{Th} \text{ local minimum} \end{array}\right|$$

  where **iterc** to core-optimization total iteration number.

- **bconv:** Functional value evolution of the best element found during optimization. It's a **iterc**-dimension vector of the form:

$$\left|\begin{array}{c} \text{Value of first best element} \\ \vdots \\ \text{Value of } \mathbf{iterc}^{Th} \text{ best element} \end{array}\right|$$

- **MOV:** If film option is enabled, this variable stock user graph evolution picture at the end of each core-optimization method. It has a Matlab movie structure.
- **FIGUSER:** Matlab figure name reserved for user graphs. This figure should be defined in **Userinterface.m**. Note that you can add other Matlab figures for your personal graphs (See next section for more details).

Output files description

During optimization, the software stock into the selected folder (See section 6.2.1) various output files:

- **Diary.txt:** Matlab Command Windows outputs are stocked during optimization process, including error messages. This is a log file using a classical "*text*" format.
- **datbmo.txt:** This file contains all the information about bmo options. It's stocked before running optimization. It's a "*text*" format file.
- **result.txt:** Stores at the end of optimization, the best element found and it's functional value. It's a "*text*" format file of the form:

  `Final point:`
  Value of parameter 1
  ⋮
  Value of parameter **ndim**
  `Functional value :`
  Functional value

- **result.csv:** We store at the end of optimization, the best element found and it's functional value. It's a (**ndim**+1)-"*csv*" format file of the form:

  [Value of parameter 1 ,..., Value of parameter **ndim**, Functional value]

- **convergence.pts:** In this file we store convergence history of the function. It's a "*pts*" format file of the form:

1	functional value 1
⋮	⋮
iter1*iter2*iterc	functional value **iter1*iter2*iterc**

- **bestconv.pts:** In this file we store history of the best functional value. It's a "*pts*" format file of the form:

1	best functional value 1
⋮	⋮
iter1*iter2*iterc	best functional value **iter1*iter2*iterc**

- **computedpoints.csv:** This file contains all computed points and their functional value. It's a (**ndim**+2)×**evalu**-"*csv*" format file of the form:

1	Parameter 1	⋯	Parameter **ndim**	functional value 1
⋮	⋱	⋮	⋮	⋮
evalu	Parameter 1	⋯	Parameter **ndim**	functional value **evalu**

- **evolution.txt:** We store all the minimum found during core optimization process and their functional value. It's a "*text*" format file of the form:

`Functional value 1:`
Functional value of local minimum 1
`point 1:`
Parameter values of local minimum 1
⋮
`Functional value iterc:`
Functional value of local minimum **iterc**
`point iterc:`
Parameter values of local minimum **iterc**

- **Workspace.mat:** Save the workspace with all variables. It's a Matlab "*mat*"-format file.

6.4.2 Optimization problem code generation

In order to use BMO with your own optimization problem you need to generate various files described in previous section.

First, we recommend to work in an folder different than the BMO root. Once you have created this folder you must build at least two Matlab files:

- **Userinit.m**: This file must have a M-script structure. It should contain the following variables: **ndim**, **xmin**, **xmax**, **vinit**.

  Other variables or initialization scripts could be added to this file.

  Example:

  We can consider a code corresponding to the minimization of the 2-dimensional Rastringin function into the subset $[-5, 5]^2$ starting from $[4, 4]$:

```
ndim=2;
for i=1:ndim
 xmin(i)=-5;
 xmax(i)=5;
 vinit(i)=4;
end
```

- **Userfunc.m**: This file must have a function-script structure. It should return the value of the cost function. There are three main kind of functions.

  -A function using Matlab routines.

  Example: Considering the previous Rastringin optimization problem:

```
function J=userfunc(x)
temp=0.;
for ii=1:length(x)
```

```
 temp=temp+x(ii).^2-cos(18*x(ii));
end
J=temp+length(x);
end
```

-A function using an external code (C++, Fortran) routines. You have two solutions:

You compile your code in an external program that reads an input file written by a Matlab function and writes an output file read by the same Matlab function.

Example: This example is associated to "External function" button in demo mode. The compiled program, called "*external.exe*", reads its inputs in "*Input.dat*" and writes its outputs into "*Output.dat*". Thus the "*userfunc.m*" file is of the form:

```
function J=userfunc(x)
fid=fopen('Input.dat','w');
fprintf(fid,'%0.12f \n ',x);
fclose(fid);
!external.exe
fid=fopen('Output.dat','r');
J=fscanf(fid,'%g')+1;
fclose(fid);
!del *.dat
end
```

Another way consist to compile directly your code using Matlab Mex library (see Matlab help file for more detail) in order to call it directly using a Matlab function. This method is generally less time-consuming.

Example: See Financial Toolbox "*userfunc.m*" for an example.

-A function using an external blackbox (Femlab, Fluent) that can be interfaced with Matlab. You can use this interfacing propriety in order to generate an "*userfunc.m*".

Example: Femlab can directly generate a m-file from its models. The geometry and equations are generated into this file and then the Femlab solver is called to give a numerical solution. You can incorporate directly this generated file into your "*userfunc.m*".

For example considering the Microfluidic Toolbox, "*userfunc.m*" reads:

```
function [cost,fem0,tcorrected,c] = directprotein(x)
global FIGUSER fem0 efem0
⋮
```

```
cl1=x(1); cx1=-x(2); (line 67)
⋮
usersolveur (line 85)
⋮
end
```

"usersolveur" is a m-file generated by Femlab and modified in order to adapt geometry to parameters defined by "x":

```
% FEMLAB Model M-file
% Generated 06-Jul-2004 10:27:40 by FEMLAB 2.3.0.153.
rhom=1013;
⋮
cx1=cx1+cw-l1; xx(26)=xx(28);yy(26)=0; (line 58)
⋮
p=[0 0 0 0 xx(5) xx(6) xx(7) cw xx(9) ... (line 87)
⋮
CO2=solid2(p,rb,wt,lr); (line 112)
⋮
[fem.sol,fem.stop]=femnlin(fem,... (line 231)
⋮
end;
```

In addition to **Userinit.m** and **Userfunc.m** user can generate other files:

- **Usergrad.m**: This file must have a m-script structure. It should build the gradient direction using a **ndim**-vector variable **fpx**. Once you have created this file you can selected User gradient option (See previous section).

  Example: For Rastringin optimization problem:

  ```
  for ii=1:ndim
  fpx(ii)=2*x(ii)+18*sin(18*x(ii));
  end
  ```

- **Useraffich.m** (*optional*): This file must have a m-script structure. This script is launched ate the end of optimization, in order to perform a post-processing treatment of the optimized solution.

  Example: At the end of optimization when using financial toolbox, the program draw on the same graph initial and optimized portfolios'allocation structure using the following code:

```
if (probtyp==1)
userinit;
traite(xoptg,vinit);
end
if (probtyp==3)
traitecdo(xoptg,vinit)
end
```

where "traite.m" and "traitecdo.m" are function-script that draw on the same graph the two portfolios.

- **Userinterface.m** (*optional*): This is a m-script file that build additional interface windows to be interfaced with bmo. The figure name "**FIGUSER**" is already reserved in Matlab workspace to build your own figure. But you can add other figures at glance.

  Example: When launching the Finance toolbox, the m-script reads:

```
FIGUSER=figure( 'Name',['Financial Graphical Data'], 'NumberTitle'
, 'off' , 'Visible', 'on');
REMOTEC=figure('Name',['Financial Tool Optimization Remote Control'],
'NumberTitle','off','Visible','on');
FINC1=figure('Name',['Simple Portfolio Functionals Optimization'],
'NumberTitle','off', 'Visible','on');
FINC2=figure('Name',['CDO ANALYSER'], 'NumberTitle', 'off', 'Visible',
'on');
```

  Figure "**FIGUSER**" is used to draw portfolio graphical data (see Toolboxes description section). Other windows display **REMOTEC**, **FINC1** and **FINC2** control panels of simple an CDO modeling problems.

  For each figure the m-script build buttons and command control.

  Details to create your own buttons, input boxes ... are available into Matlab Family Help menu (see **uicontrol** help page).

- **Userexiting.m** (*optional*): Script executed when exiting the

  Example: When exiting bmo after using Financial toolboxes we need to close added user-interface windows using the following code:

```
set(FINC1,'Visible','off');
set(FINC2,'Visible','off');
set(REMOTEC,'Visible','off');
```

- **Useropt.m**: In order to solve an optimization problem, maybe you want to integrate your own optimization routine as core-optimization method, more adapted to the considered problem. To do so, you need to create an m-script "Useropt.m" and activate "User" method in **zone 2** of "*main control panel*"

  The main structure of this m-script should read:

  - Consider variable x_0, a **ndim**-vector as initial condition of your optimization algorithm.
  - Perform your algorithm
  - Return found solution into x_0 and its associated functional value into the real variable f.

Example: You can find an application regarding the file "gradopt.m" that corresponds to the descent method included into bmo.

6.5 Credits

Mathematical research group:

Benjamin Ivorra (ivorra@math.univ-montp2.fr)

Mohammadi Bijan (mohamadi@math.univ-montp2.fr)

Patrick Redont (redont@math.univ-montp2.fr)

Jean-Paul Dufour

Code Developers:

- **Main Optimization Software:** *Benjamin Ivorra, Mohammadi Bijan*
- **Coastal structure Toolbox:** *Damien Isebe (isebe@math.univ-montp2.fr), Pascal Azerad, Frederic Bouchette*
- **Spray dispersion Toolbox:** *Jean-Marc Brun (brun@math.univ-montp2.fr), Cemagref team*
- **Seismic Toolbox:** *Michel Cuer (cuer@math.univ-montp2.fr), Carole Duffet, Benjamin Ivorra*
- **Genetic Algorithm:** *Laurent Dumas, Benjamin Ivorra*
- **Optical Toolbox:** *Benjamin Ivorra, Laurent Dumas, Olivier Durand*
- **Microfluidic Toolbox:** *Benjamin Ivorra, Bijan Mohammadi, David Hertzog, Juan Santiago*
- **Financial Toolbox:** *Benjamin Ivorra, BNP Asset Management team*
- **Burgers Toolbox:** *Angel Ramos Del-Olmo, Benjamin Ivorra*

Conclusions

Nous venons de présenter une nouvelle classe de méthodes d'optimisation globale. La méthodologie présentée ici nous a permis d'améliorer, d'un point de vue précision et/ou complexité, aussi bien des algorithmes déterministes que stochastiques.

Deux algorithmes inclus dans cette classe ont ensuite été appliqués à divers problèmes d'optimisation industriels. Lorsque cela a été possible, les résultats obtenus ont été comparés avec ceux provenant d'autres méthodes, notamment d'algorithmes génétiques. Dans tous les cas, nos algorithmes ont donné des résultats équivalents ou meilleurs, pour un coût de calcul moindre.

Pour chacun de ces problèmes, de nouvelles voies de recherche se sont ouvertes. Par manque de temps nous n'avons pu aller plus loin dans leur étude, mais il serait intéressant de suivre ces idées pour de futurs travaux.

Durant cette thèse d'autres applications, non présentées dans ce manuscript, ont été étudiées à l'aide de nos méthodes:

- Contrôle de température et d'émission de polluants à l'intérieur d'une flamme de bec bunsen. Ces travaux ont donné lieu à la publication suivante:

  • Debiane, L. and Ivorra, B. and Mohammadi, B. and Nicoud, F. and Ern, A. and Poinsot, T. and Pitsch, H. "*A low-complexity global optimization algorithm for temperature and pollution control of a complex chemistry flame*". International Journal of Computational Fluid Dynamics, 2006, in revision process [72, 73].

- Optimisation de forme de structures côtières. Les résultats ont été publiés dans le papier suivant:

  • Azerad, P. and Isebe, D. and Ivorra, B. and Mohammadi, B. and Bouchette, F. "Optimal shape design of coastal structures minimizing coastal erosion". Proceedings of workshop on inverse problems, CIRM (2006) [74].

- Reconstruction de sous-sol à l'aide de données tomographiques. Cette problématique a été présentée avec les premiers résultats lors de la conférence suivante:

  • Ivorra, B. and Cuer, M. and Duffet, C. and Mohammadi, B. "*Quantifying Uncertainties in Seismic Tomography*". 2005 SIAM Conference on Mathematical and Computational Issues in the Geosciences (Avignon).

Part IV

Appendixes

Appendix A

Resolution of Pointwise control problems of the viscous Burgers equation by local and global optimization algorithms

Nous présentons ici des travaux commencés durant le stage de DEA, effectué à l'Université Complutense de Madrid sous la direction du professeur Angel Manuel Ramos del Olmo. Ces travaux ont été ensuite repris et enrichis durant la thèse. L'objectif ici est double:

- *Comparer, comme cela a été fait dans les précédents chapitres, les méthodes SD2A et GA sur un problème de contrôle par point de l'équation visqueuse de Burgers.*
- *Démontrer, en fonction du coefficient de pondération fixé devant le terme de contrôle de la fonction coût, l'intérêt ou non de l'emploi d'une méthode globale. Pour se faire, les résultats précédents sont aussi comparés avec ceux obtenus par un algorithme de type L-BFGS.*

A.1 Problem Formulation

We consider the following viscous Burgers equation:

$$y_t - \nu y_{xx} + y y_x = f \qquad \text{in } Q = (0,1) \times (0,T) \tag{A.1}$$

where $T \in]0, \infty)$ is an horizon time, $\nu > 0$ is a viscosity parameter and f is a density of external forces.

We look for a control $v(t)$ forcing the solution at a point $a \in [0,1]$, completing the equation with the initial and boundary conditions used in [75] (and in other references cited here):

$$\begin{cases} y_t - \upsilon y_{xx} + y y_x = f + v\delta(x-a) & \text{in Q} \\ y_x(0,t) = 0,\ y(1,t) = 0 & \text{for } t \in [0,T] \\ y(0) = y_0 & \text{for } t \in [0,1] \end{cases} \tag{A.2}$$

where $\delta(x-a)$ denotes the Dirac measure at point a.

A variational formulation of the above system (A.2) is provided by $y(t) \in L^2(0,T;V_0) \cap H^1(0,T;V_0')$, such that $y(0) = y_0$ and:

$$\begin{cases} \int_0^T < y_t, z >_{V_0' \times V_0} dt + \nu \int_0^T (y_x, z_x) dt + \int_0^T (y y_x, z) dt \\ \qquad = \int_0^T (f, z) dt + \int_0^T v z(a) dt \qquad \forall\ z \in L^2(0,T;V_0), \end{cases} \tag{A.3}$$

where $V_0 = \{z \in H^1(0,1) : z(1) = 0\}$ and $(\cdot,\cdot)$ denotes the scalar product in $L^2(0,1)$ defined by $(y,z) = \displaystyle\int_0^1 yz dx$.

A.1.1 Problem Formulation

Let us consider a target function $y_T \in L^2(0,1)$. We define the bounded control space as $\mathcal{U} = \{f \in L^2(0,T) : \text{for all } x \in [0,T] \ \ |f(x)| \leq S_{\rm b}\}$, where $S_{\rm b} \in \mathbb{R}$. The objective is to find a control $u \in \mathcal{U}$, forcing the solution at the point a, such that $y(T)$ is close to y_T in $(0,1)$ at a minimal cost (norm) for the control. To do this, we define the cost function J by:

$$J(v) = \alpha \parallel v \parallel^2_{\mathcal{U}} + \parallel y(T) - y_T \parallel^2_{L^2(0,1)} \tag{A.4}$$

where $\alpha \geq 0$.

Then, we define the optimal control problem $(\mathcal{CP})$ as:

$$\begin{cases} \text{Find } u \in \mathcal{U} \text{ such that:} \\ J(u) = \min_{v \in \mathcal{U}} J(v) \end{cases}$$

When $\alpha = 0$ there is no proof for existence of solution for $(\mathcal{CP})$. In this case, we are interested by finding a solution as close as possible to the infimum of the function.

In order to solve numerically $(\mathcal{CP})$ we need to define an associated discrete problem. Following [75] we consider:

- A time discretization step Δt, defined by $\Delta t = T/L$, where L is a positive integer. Then, if $t^l = l\Delta t$, we have $0 < t^1 < t^2 < \cdots < t^L = T$.
- A space discretization step h, defined by $h = 1/I$, where I is a positive integer. Then, if $x_i = (i-1)h$, we have $0 = x_1 < x_2 < \cdots < x_I < x_{I+1} = 1$.

Then we approximate V_0 by

$$V_{0h} = \{z \in \mathcal{C}^0[0,1] : z(1) = 0, z|_{(x_i, x_{i+1})} \in P_1, i = 1, \cdots, I\},$$

where P_1 is the space of the polynomials of degree ≤ 1. We define a_h and b_h by

$$a_h(y,z) = \int_0^1 y_x z_x dx, \qquad b_h(w,y,z) = \int_0^1 w y_x z dx.$$

Thus the discrete problem associated to $(\mathcal{CP})$ is defined by the following finite-dimensional minimization problem $(\mathcal{CP})_h^{\Delta t}$:

$$\begin{cases} \text{Find } u_h^{\Delta t} = \{u^l\}_{l=1\cdots L} \in \mathcal{U}^{\Delta t} \text{ such that:} \\ J_h^{\Delta t}(u_h^{\Delta t}) \leq J_h^{\Delta t}(v),\ \forall v = \{v^l\}_{l=1\cdots L} \in \mathcal{U}^{\Delta t}, \end{cases}$$

where the discrete control space $\mathcal{U}^{\Delta t}$ in $(\mathcal{CP})_h^{\Delta t}$ is initially equal to $[-S_{\mathrm{b}}\,, S_{\mathrm{b}}\,]^L$ and

$$J_h^{\Delta t}(v) = \alpha \Delta t \sum_{l=1}^{L} |v^l|^2 + l\left((1-\theta) \parallel y_h^{L-1} - y_T \parallel^2_{L^2(0,1)} + \theta \parallel y_h^L - y_T \parallel^2_{L^2(0,1)}\right)$$

with $\theta \in (0,1]$ y^L defined from the solution of the following full discretization of the Burgers equation (1):

$$\left\{\begin{array}{l} y_h^l \in V_{0h},\ l = 0, ..., L \text{ such that, } \forall z \in V_{0h}: \\ (y_h^0, z) = (y_0, z) \\ (\frac{y_h^1 - y_h^0}{\Delta t}, z) + \nu a_h(\frac{2}{3}y_h^1 + \frac{1}{3}y_h^0, z) + b_h(y_h^0, y_h^0, z) = (f^1, z) + \frac{2}{3}v^l z(a) \\ for \quad l \geq 2: \\ (\frac{\frac{3}{2}y_h^l - 2y_h^{l-1} + \frac{1}{2}y_h^{l-2}}{\Delta t}, z) + \nu a_h(y_h^l, z) + b_h(2y_h^{l-1} - y_h^{l-2}, 2y_h^{l-1} - y_h^{l-2}, z) \\ \qquad\qquad = (f^l, z) + v^l z(a). \end{array}\right. \tag{A.5}$$

In order to keep a certain regularity in the computed controls and also to reduce the degrees of freedom, we generate them by spline interpolation [12] through a number of $N_S = 8$ points included in $[-S_{\mathrm{b}}\,, S_{\mathrm{b}}\,]$. Thus the new control space becomes $\mathcal{U}_{N_S}^{\Delta t} = [-S_{\mathrm{b}}\,, S_{\mathrm{b}}\,]^{N_S}$ and we denote by $(\mathcal{CP})_{h,N_S,\alpha}^{\Delta t}$ the new associated minimization problem. For each control $u \in \mathcal{U}_{N_S}^{\Delta t}$, we obtain an associated $\overline{u} \in \mathcal{U}^{\Delta t}$, which is used in the discrete state equation.

A.1.2 Results

We consider the test problem defined as follows: $a = 0.5$, $T = 1$, $I = 128$, $N = 1500$, $\nu = 10^{-2}$, $S_{\mathrm{b}} = 20$, α is user defined, $y_0 = 0$ and

$$f(x,t) = \left\{\begin{array}{ll} 1 & \text{if } (x,t) \in (0, 1/2) \times (0,T), \\ 2(1-x) & \text{if } (x,t) \in [1/2, 1) \times (0,T), \end{array}\right.$$

We construct the target solution y_T using the predefined control $u_T(t) = 9 + \sin(t * 0.2 * \pi)$, $t \in [0,1]$. Figure A.1 shows u_T and the associated target solution $y_T = y(T; u_T)$. We point that u_T does not belong to $\mathcal{U}_{N_S}^{\Delta t}$ defined previously.

In order to solve numerically $(\mathcal{CP})_{h,N_S,\alpha}^{\Delta t}$ with α set to 0, 0.01, 0.1 and 1, respectively, we will use the three different optimization methods presented previously:

- A L-BFGS algorithm: This is a modification of the BFGS method [76]. This deterministic method is based on a limited-memory quasi-Newton algorithm well suited for large-scale bound-constrained or unconstrained optimization. It is out of the scope of this paper but the interested reader can find more details in literature (See, for instance, [12, 77]). It's applied with the same configuration as the one used in [75]. It starts from a null control.

- SD2A and GA algorithms applied with parameters specified in chapter 2. For SD2A, 0 is chosen as first initial condition.

In all algorithms, points already computed are stocked in memory in order to avoid recomputations. This technique is useful in particular for SD2A and GA techniques. Indeed, in HGA each individual can be present and repeated in various generations. In SD2A, if two consecutive initial conditions give the same minimum, dynamical system trajectory is projected on the maximum or minimum boundary border. This can occur various time during the optimization process.

All these parameters are fixed and used in all computations of this paper. Optimization algorithms are run on a 3Ghz PC with 512 Mb Memory. One functional evaluation takes around 4 seconds (real-time).

For each algorithm, optimized controls and associated solutions are presented in Figure A.2. Optimization data are summarized in Table A.1. Convergence histories for SD2A and GA are shown in Figure A.3. As it can be seen on Table A.1, as α decreases the iteration number of SD2A increases. This is due to the de-convexification of the cost function resulting on the apparition of local minima and thus on the diminution of recomputed points during SD2A optimization process.

As we can observe on Figure A.2 and Table A.1:

- For $\alpha = 1$, SD2A, GA and L-BFGS lead to the same result.
- For $\alpha = 0.1$ and $\alpha = 0.01$, SD2A cost function value is lower than GA and L-BFGS ones. Difference between SD2A and L-BFGS is short. However, SD2A results are better in term of optimal control. This is visible on Table A.1 on respective control values.
- For $\alpha = 0$, SD2A and GA over-perform the L-BFGS algorithm. SD2A is faster than GA and found a better solution.

As we can observe, for large values of α the use of a global optimization methods seems to be not necessary. On the other hand, for small values of α, the choice of a global method seems a better option because the problem becomes highly non-linear. In addition, in all cases, SD2A has given better results, and in a lower time, than GA.

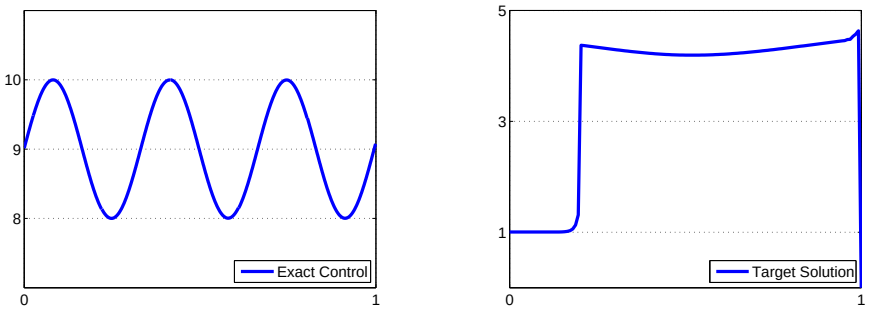


Figure A.1: **Left**: Predefined control $u_T(t) = 9+\sin(t*0.2*\pi), t \in [0,1]$. **Right**: Associated target solution y_T.

	L-BFGS	GA	SD2A
$\alpha = 1$			
Final Cost Function Value	6.6	6.6	6.6
Iteration Number	200	4000	600
Computational Time (mn)	13	266	42
$\alpha = 0.1$			
Final Cost Function Value	.95	1.1	.94
Control Value	.4	.44	.37
Iteration Number	200	4000	1200
Computational Time (mn)	13	266	80
$\alpha = 0.01$			
Final Cost Function Value	.15	.16	.14
Control Value	.045	.049	.04
Iteration Number	200	4000	2000
Computational Time (mn)	13	266	133
$\alpha = 0$			
Final Cost Function Value	3×10^{-2}	8×10^{-3}	5×10^{-3}
Iteration Number	200	4000	2000
Computational Time (mn)	13	266	133

Table A.1: Optimization data: From **Top** to **Bottom** $\alpha = 1, \alpha = 0.1, \alpha = 0.01$, and $\alpha = 0$. Cost function values, control value in the cost function (only for $\alpha = 0.1$ and $\alpha = 0.01$), Iteration number and computational time (real-time in minutes) after optimization algorithm are reported.

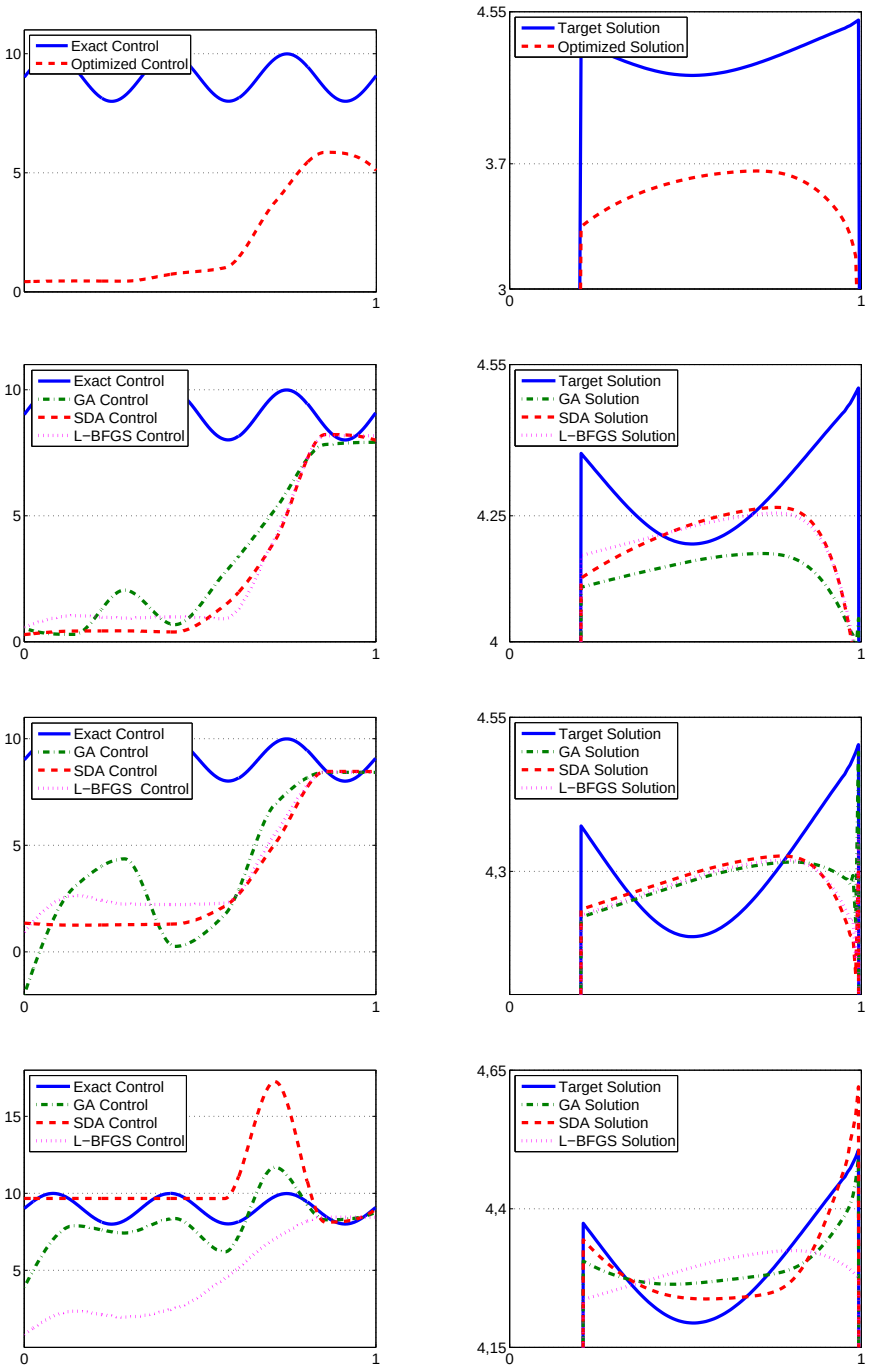


Figure A.2: From (**Top**) to (**Bottom**): Results for $\alpha = 1$, $\alpha = 0.1$, $\alpha = 0.01$ and $\alpha = 0$. (**Left**) Optimized control. (**Right**) Zoom on the three optimized solutions in order to precise visual differences.

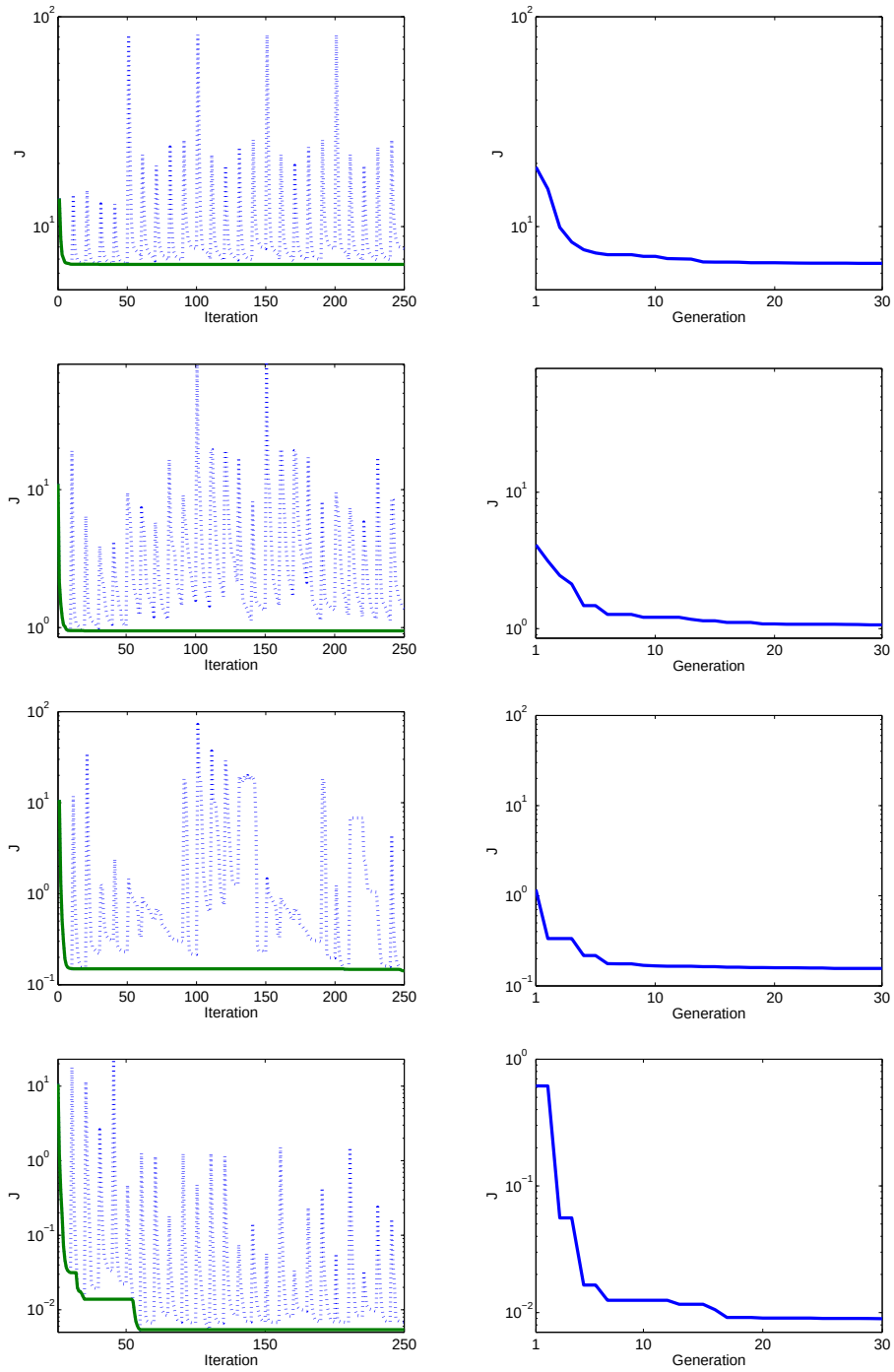


Figure A.3: SD2A (**left**), GA (**right**) Convergence histories: Best element (**solid line**) and convergence history (**dashed line**). From **top** to **bottom**: $\alpha = 1, \alpha = 0.1, \alpha = 0.01, \alpha = 0$.

A.2 Conclusions

Results obtained with our SD2A have been compared to those obtained with a L-BFGS method and a GA approach. The use of the global method has been justified for small balances of the control norm. Otherwise, a local method is efficient. This is an intuitive result as the control term increases the convexity of the considered cost function.

Appendix B

Quelques propriétés pour les systèmes dynamique d'ordre 2

Les résultats présentés ici ont été trouvés en tentant de résoudre le théorème 2 du chapitre 2 à l'aide des mêmes outils que dans le cas monodimensionel. Bien que nous ayons du changer de méthode, nous présentons ici quelques propriétés intéressantes. Ces travaux ont été réalisés avec l'aide du Professeur Jean-Paul Dufour (Université Montpellier II).

Soit $J : R^n \to R$, C^3 telle que:
$\cdot$ $Hess(J)$ soit inversible
$\cdot$ $\exists M \in R$ tel que $\forall \|x\| \geq M$, $J(x) = \frac{K}{2}\|x\|^2$
On considère le système suivant:

$$\begin{cases} \ddot{x}(t) + \dot{x}(t) = -\nabla J(x(t)) \\ x(0) = x_0 \\ \dot{x}(0) = \dot{x_0} \end{cases} \tag{B.1}$$

Posons:

$$\dot{x}(t) = y(t)$$

alors le système (B.1) s'écrit:

$$\begin{cases} \dot{x}(t) = y(t) \\ \dot{y}(t) = -y(t) - \nabla J(x(t)) \\ x(0) = x_0 \\ \dot{x}(0) = \dot{x_0} \end{cases} \tag{B.2}$$

Nous allons maintenant étudier le comportement des solutions de (B.2).

Lemma 33 *Les points critiques des solutions de (B.2) sont les extrémums de la fonctionnelle J.*

Preuve:

En effet, lorsqu'une trajectoire converge vers un point critique m_i, sa vitesse et son acceleration s'annulent lorsque $t \to \pm\infty$, ainsi on trouve que:

$$\lim_{t\to\pm\infty} [\ddot{x}(t) + \dot{x}(t)] = \lim_{t\to\pm\infty} [-\nabla J(x(t))] = -\nabla J(m_i) = 0$$

□

Lemma 34 *La $\omega - limite$ et la $\alpha - limite$ d'une solution γ du système (B.2) sont des points.*

Preuve:

Nous allons démontrer la propriété pour la $\omega - limite$, la démonstration reste aussi vraie pour la $\alpha - limite$ en changeant les limites vers $+\infty$ en -∞.

Supposons par l'absurde que la $\omega - limite$ de γ ait deux points distincts Z_1 et Z_2.

i.e: $\exists (t_n)_n, (t'_n)_n$ telles que:

$$\lim_{n\to+\infty} \gamma(t_n) = Z_1$$

$$\lim_{n\to+\infty} \gamma(t'_n) = Z_2$$

avec: $Z_1 = (z_1, \dot{z}_1)$ et $Z_2 = (z, 2, \dot{z}_2)$.

1- Considérons la fonction de Lyapounov:

$$L(x(t), y(t)) = \frac{\|y(t)\|_2^2}{2} - J(x(t))$$

On note: $\gamma(t) = (x(t), y(t))$. Alors:

$$\frac{d}{dt} L(x(t), y(t)) = \sum_{i=1}^{n} y_i(t)\dot{y}_i(t) - \frac{\partial J}{\partial x_i}(x(t))$$

Or comme $\gamma(t)$ est solution de (B.2) on a:

$$\dot{y}(t) = -y(t) - \nabla J(x(t))$$

Et alors:

$$\frac{d}{dt} L(x(t), y(t)) = -\sum_{i=1}^{n} y_i(t)^2 \qquad (*)$$

Donc le long de la trajectoire $\gamma(t)$ la fonction de Lyapounov est décroissante. Strictement lorsque la vitesse $y(t)$ est non nulle.

2-On montre ensuite que: $L(Z_1) = L(Z_2)$:

• $L(Z_1) \geq L(Z_2)$: Comme la fonction de Lyapounouv L est décroissante le long de la trajectoire γ:

$\forall t_{n_0}$, $\exists t'_{n'_0}$ telle:

$$\forall t' > t'_{n'_0}, L(\gamma(t_{n_0})) \geq L(\gamma(t'))$$

Donc par continuité des fonctions de Lyapounov:

$$\lim_{n\to+\infty} L(\gamma(t_n)) \geq \lim_{n\to+\infty} L(\gamma(t'_n))$$

c'est à dire:

$$L(Z_1) \geq L(Z_2)$$

• $L(Z_2) \geq L(Z_1)$: On procède de la même manière en inversant les rôles de Z_1 et Z_2.

3- Nous considérons maintenant deux cas:

• $\dot{z_1} \neq 0$:

La trajectoire $\gamma(t)$ effectue des *'aller-retours'* entre les points Z_1 et Z_2. À chaque fois la distance est au moins de $d(Z_1, Z_2)$.

Or une portion de cette trajectoire, de distance minimale:

$\cdot$ $T = \frac{\dot{z_1}}{2}$ si $\dot{z_2} \leq 0$.

$\cdot$ $T = min(\frac{\dot{z_1}}{2}, d(Z_1, Z_2))$ si $\dot{z_2} > 0$.

est, dans le plan de phase, au dessus du plan $\{\dot{x} = \frac{\dot{z_1}}{2}\}$. Donc à chaque trajet la fonction de Lyapounov, d'après (*), diminue d'au moins:

$$T \cdot \frac{\dot{z_1}}{2}$$

Alors $L(Z_1)$ ne peut pas être égal à $L(Z_2)$. Ce qui est absurde.

• $\dot{z_1} = \dot{z_2} = 0$:

Lorsque $t \to +\infty$, la vitesse de $\dot{x}(t)$ tend vers zéro:

En effet:

• Si $\dot{x}(t)$ ne tend pas vers zéro:

Il existe ϵ tel que $\dot{x}(t)$ soit supérieur à ϵ une infinité de fois sur une distance minimale D_ϵ.

Donc à chaque trajet la fonction de Lyapounov, d'après (*), diminue d'au moins: ϵD_ϵ. $L(z_1)$ ne peut pas être égal à $L(z_2)$, ce qui est absurde.

• Si $\dot{x}(t)$ tend vers zéro: la Hessienne de J étant inversible, l'espace de phase se décompose en ouverts qui sont chacun les bassins d'attractions des points critiques

attractifs. Pour une vitesse suffisamment faible la trajectoire γ va entrer dans un de ces bassins d'attraction, Ω_i, et converger vers le point critique associé m_i. Alors:

$$\lim_{t\to+\infty} \gamma(t) = m_i = Z_1 = Z_2$$

Ce qui contredit l'hypothèse: $Z_1 \neq Z_2$.

□

Lemma 35 *Les points critiques qui sont des minimums locaux pour la fonctionnelle J sont stables au sens de Lyapounov:*

Preuve:

Notons Z un tel point critique.
On considère la fonction de Lyapounov définie par $L_2 = L - J(Z)$, comme Z est un minimum local il existe un voisinage Ω_Z tel que:

$$\forall(z, \dot{z}) \in \Omega_Z \backslash \{Z\} \times R, L_2(z) > 0 \;\; et \;\; L_2(Z) = 0$$

Donc, d'après les théorème de stabilité 3 du chapitre 1, Z est stable au sens de Lyapounov.

□

Les Lemme 2 et 3 nous assurent donc que les trajectoires solutions de (B.2) ne peuvent avoir que trois aspects:

1. aller de l'infini à un extremum de J
2. aller d'un extremum de J à l'infini
3. aller d'un extremum de J à un autre

Nous allons montrer que, à un extremum m_i donné, nous avons forcément au moins une trajectoire de type 1 ou 3, c'est à dire:

Lemma 36 *Á chaque extremum m_i de J, il existe une trajectoire, solution du système (B.1), non bornée et qui atteint m_i.*

Preuve:

1- Les points critiques, du système (B.2) ne sont pas répulsifs:
Les trajectoires γ des solutions du système B.2 ont pour composantes:

$$X_\gamma(t) = (y(t), -y(t) - \nabla J(x(t)))$$

La composante $y(t)$, dans une partie du plan de phase, est attractive. Donc un point critique du système (B.2) ne peut être répulsif.

2- Supposons par l'absurde, pour un extremum de J donné, m_0, nous n'ayons que des trajectoires de type 2:
Alors on se trouve face à deux cas:

• Les trajectoires proviennent de points critiques distincts:
Alors on recouvre un voisinage de m_0, qui est une variété de dimension $2n-1$, par des sous-variétés de dimension strictement inférieure à $2n-1$, ce qui est impossible [78].

• Les trajectoires proviennent toutes du même point critique m_1: Alors la variété des trajectoires répulsives de m_1 est de dimension $2n-1$. Donc:
Le point critique m_1 est répulsif ce qui d'après le lemme 3 est impossible.

Donc il y a au moins une trajectoire qui provient de m_0 qui est non bornée.

□

Bibliography

[1] B. Ivorra, B. Mohammadi, J. Santiago, and D. Hertzog. Semi-deterministic global optimization algorithms. *Proceeding of 7ème colloque national en calcul des structures*, 2:253–259, 2005.

[2] B. Ivorra, B. Mohammadi, and P. Redont. Low-complexity global optimization by solution of bvp. *Journal of Global Optimization*, submitted, 2005.

[3] B. Ivorra, B. Mohammadi, and D.E. Santiago, J.G.and Hertzog. Semi-deterministic and genetic algorithms for global optimization of microfluidic protein folding devices. *International Journal of Numerical Method in Engineering*, 66:319–333, 2006.

[4] B. Ivorra, B. Mohammadi, L. Dumas, O. Durand, and Y. Moreau. Semi-deterministic recursive optimization methods for multichannel optical filters. *Numerical Mathematics and Advanced applications*, Accepted, to be published in 2006.

[5] J. B. Knight, A. Vishwanath, J. P. Brody, and R. H. Austin. Hydrodynamic focusing on a silicon chip: Mixing nanoliters in microseconds. *Physical Review Letters*, 80:3863–3866, 1998.

[6] D.E. Hertzog, B. Ivorra, B. Mohammadi, O. Bakajin, and J.G. Santiago. Optimization of a fast microfluidic mixer for studying protein folding kinetics. *Analytical chemistry*, 2006.

[7] B. Ivorra, B. Mohammadi, Q. Guillaume, T. Rim, and S. Delcourt. An hybrid optimization method for risk measure reduction of a credit portfolio under constraints. *Applied Mathematical Finance*, submitted, 2006.

[8] L. Dumas, V. Herbert, and F. Muyl. Hybrid method for aerodynamic shape optimization in automotive industry. *Computers and Fluids*, 33(5):849–858, 2004.

[9] D. Goldberg. *Genetic algorithms in search, optimization and machine learning.* Addison Wesley, 1989.

[10] J. Skaar and K.M. Risvik. A genetic algorithm for the inverse problem in synthesis of fiber gratings. *Journal of Lightwave Technology*, 16(10):1928–1932, 1998.

[11] C. M. Fonseca and J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1–16, 1995.

[12] B. Mohammadi and J-H. Saiac. *Pratique de la simulation numérique*. Dunod, 2002.

[13] H. Attouch and R. Cominetti. A dynamical approach to convex minimization coupling approximation with the steepest descent method. *Journal of Differential Equations*, 128(2):519–540, 1996.

[14] B. Mohammadi and O. Pironneau. *Applied Shape Optimization for Fluids*. Oxford University Press, 2001.

[15] H. Attouch and F. Alvarez. The heavy ball with friction dynamical system for convex constrained minimization problems. *Lecture Notes in Economic and Mathematic Systems*, 481:25–35, 2000.

[16] Hale J.K. *Ordinary differential equations.* Wiley-Interscience , New York., 1969,.

[17] Verhulst F. *Nonlinear differential equations and dynamical systems.* Springer-Verlag., 1990.

[18] B. Ivorra. Semi-deterministic global optimization. *PhD. University of Montpellier 2*, 2006.

[19] J.C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright. Convergence properties of the nelder-mead simplex method in low dimensions. *SIAM Journal of Optimization*, 9(1):112–147, 1998.

[20] T.F. Coleman and Y. Li. An interior, trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on Optimization*, 6:418–445, 1996.

[21] T.F. Coleman and Y. Li. On the convergence of reflective newton methods for large-scale nonlinear minimization subject to bounds. *Mathematical Programming*, 67(2):189–224, 1994.

[22] R. Fletcher. Practical methods of optimization. *Unconstrained Optimization, John Wiley and Sons*, 1, 1980.

[23] S.P. Han. A globally convergent method for nonlinear programming. *Journal of Optimization Theory and Applications*, 22:297, 1977.

[24] M.J.D. Powell. A fast algorithm for nonlinearly constrained optimization calculations. *Lecture Notes in Mathematics, Springer Verlag*, 630, 1978.

[25] M.J.D. Powell. The convergence of variable metric methods for nonlinearly constrained optimization calculations. *Nonlinear Programming*, 3, 1978.

[26] P.E. Gill, W. Murray, and M.H. Wright. *Practical Optimization*. London, Academic Press, 1981.

[27] C.A. Floudas. Handbook of test problems in local and global optimization. *Kluwer Academic Publishers*, 2000.

[28] J. Chow, G. Town, B. Eggleton, M. Ibsen, K. Sugden, and I. Bennion. Multiwavelength generation in an erbium-doped fiber laser using in-fiber com filters. *IEEE Photonic Technology Letter*, 8(1):60–62, 1996.

[29] D. Wei, T. Li, Y. Zhao, and S. Jian. Multiwavelength erbium-doped fiber ring lasers with overlap-written fiber bragg gratings. *Optics Letter*, 25(16):1150–1152, 2000.

[30] H. Helmers, O. Durand, G.H. Duan, E. Gohin, J. Landreau, J. Jacquet, and I. Riant. 45 nm tunability in c-band obtained with external cavity laser including novel sampled fibre bragg grating. *Electronic Letter*, 38(24):1535–1536, 2002.

[31] J.E. Rothenberg, H. Li, Y. Li, J. Popelek, Y. Sheng, Y. Wang, R.B. Wilcox, and DJ. Zweiback. Fiber bragg gratings and phase-only sampling for high channel counts. *IEEE Photonic Technology Letter*, 14(9):1309–1311, 2002.

[32] T. Erdogan. Fiber grating spectra. *J. of lightwave technology*, 15(8):1277–1294, 1997.

[33] G. Tremblay, J.N. Gillet, Y. Sheng, M. Bernier, and G. Paul-Hus. Optimizing fiber bragg gratings using a genetic algorithm with fabrication-constraint encoding. *Journal of lightwave technology*, 23(12):4382–4386, 2005.

[34] B.J. Eggleton, P.A. Krug, L. Polodian, and F. Ouellette. Long periodic superstructure bragg gratings in optical fibres. *Electronic Letter*, 30(19):1620–1622, 1994.

[35] W.H. Loh, F.Q. Zhou, and J.J. Pan. Novel designs for sampled grating-based multiplexers-demultiplexers. *Optics Letter*, 34(21):1457–1459, 1999.

[36] M. Ibsen, M.K. Durkin, M.J. Cole, and R.I. Laming. sinc-sampled fiber bragg gratings for identical multiple wavelength operation. *IEEE Photonic Technology Letter*, 10(6):842–844, 1998.

[37] A.V. Buryak, K.Y. Kolossovski, and D.Y. Stepanov. Optimization of refractive index sampling for multichannel fiber bragg gratings. *IEEE Journal of Quantum Electronic*, 39:91–98, 2003.

[38] H. Ishii, Y. Tohmori, Y. Yoshikuni, T. Tamamura, and Y. Kondo. Multiple-phase-shift super structure grating dbr lasers for broad wavelength tuning. *IEEE Photonic Technology Letter*, 5(6):613–615, 1993.

[39] C. Bock and J. Prat. Scalable wdma/tdma protocol for passive optical networks that avoids upstream synchronization and features dynamic bandwidth allocation. *Journal of Optical Network*, 4:226–236, 2005.

[40] Y. Moreau, P. Coudray, K. Kribich, and P. Etienne. Systeme d'acces multiple par code (cdma) en technologie organique-inorganique. *Proceeding of JNOG*, 13(3):365–367, 2000.

[41] J.A. Salehi. Code division multiple-access techniques in optical fiber networks - part i : Fundamental principles. *IEEE Trans. Comm.*, 73(8), 1989.

[42] M. Kavehrad and D. Zaccarin. Optical code-division-multiplexed systems based on spectral encoding of noncoherent sources. *Jour. of Lightwave Tech.*, 13(3), 1995.

[43] C. Lam, R. Vrijien, M. Wu, and E. Yablonovitch. Multi wavelength optical code division multiplexing. *1.SPIE Optical Engineering Press*, 1999.

[44] D.E. Hertzog, X. Michalet, M. Jager, X. Kong, J.G. Santiago, S. Weiss, and O. Bakajin. Femtomole mixer for microsecond kinetic studies of protein folding. *Analytical Chemistry*, 75(24):7169–7178, 2004.

[45] Roder H. Stepwise helix formation and chain compaction during protein folding. *Proceedings of the National Academy of Sciences of the USA*, 101:1793–1794, 2004.

[46] C. M. Jones, E. R. Henry, Y. Hu, C.-K. Chan, S. D. Luck, A. Bhuyan, H. Roder, J. Hofrichter, and W. A. Eaton. Fast events in protein folding initiated by nanosecond laser photolysis. *Proceedings of the National Academy of Sciences of the USA*, 90:11860–11864, 1993.

[47] S. J. Hagen and W. A. Eaton. Two-state expansion and collapse of a polypeptide. *Journal of Molecular Biology*, 301:1037–1045, 2000.

[48] K. M. Pryse, T. G. Bruckman, B. W. Maxfield, and E. L. Elson. Kinetics and mechanism of the folding of cytochrome c. *Biochemistry*, 31(22):5127–5136, 1992.

[49] M. Jacob, G. Holtermann, D. Perl, J. Reinstein, T. Schindler, M. A. Geeves, and F. X. Schmid. Microsecond folding of the cold shock protein measured by a pressure jump technique. *Biochemistry*, 38:2882–2891, 1999.

[50] C.K. Chan, Y. Hu, S. Takahashi, D. L. Rousseau, and W. A. Eaton. Submillisecond protein folding kinetics studied by ultrarapid mixing. *Proceedings of the National Academy of Sciences of the USA*, 94:1779–1784, 1997.

[51] Pollack L., M. W. Tate, N. C. Darnton, J. B. Knight, S. M. Gruner, W. A. Eaton, and R. H. Austin. Compactness of the denatured state of a fast-folding protein measured by submillisecond small angle x-ray scattering. *Proceedings of the National Academy of Sciences of the USA*, 96:10115–10117, 1999.

[52] S.-H. Park, M. C. R. Shastry, and H. Roder. Folding dynamics of the b1 domain of protein g explored by ultrarapid mixing. *Nature, Structural Biology*, 6:943–947, 1999.

[53] J. P. Brody, P. Yager, R.E. Goldstein, and R.H. Austin. Biotechnology at low reynolds numbers. *Biophysical Journal*, 71(6):3430–3441, 1996.

[54] R. Russell, I.S. Millet, M.W. Tate, L.W. Kwok, B. Nakatani, S.M. Gruner, S.G.J. Mochrie, V.S. Pande, S. Doniach, D. Herschlag, and L. Pollack. Rapid compaction during rna folding. *Proceedings of the National Academy of Sciences of the USA*, 99:4266–4271, 2002.

[55] M. C. R. Shastry, S. D. Luck, and H. Roder. A continuous-flow capillary mixer to monitor reactions on the microsecond time scale. *Biophysical Journal*, 74:2714–2721, 1998.

[56] J. M. Ottino. *The Kinematics of Mixing: Stretching, Chaos, and Transport*. Cambridge University Press, 1989.

[57] N. Darnton, O. Bakajin, R. Huang, B. North, J. Tegenfeldt, E. Cox, J. Sturn, and R. H. Austin. Condensed matter. *Journal of Physics*, 13:4891–4902, 2001.

[58] W. M. Deen. *Analysis of Transport Phenomena*. New York, Oxford University Press, 1998.

[59] T. Hughes and A. Brooks. A multi-dimensional upwind scheme with no crosswind diffusion, in t. hughes, ed., finite element methods for convection dominated flows. *ASME, New York*, 34:19–35, 1979.

[60] P. Deuflhard. A modified newton method for the solution of ill-conditioned systems of nonlinear equations with application to multiple shooting. *Numerical Mathematics*, 32:289–315, 1974.

[61] http://www.investopia.com.

[62] R. Bruyere. *Credit Derivatives and Structured Credit*. Wiley Finance, 2005.

[63] R.T. Rockafellar and S. Uryasev. Conditional value-at-risk for general loss distributions. *Journal of Banking & Finance*, 26:1443–1471, 2002.

[64] P. Artzner, D. Delbaen, J.M. Eber, and D. Heath. Coherent measures of risk. *Risk*, 1998.

[65] P. Krokhmal, J. Palmquist, and S. Uryasev. Conditional value-at-risk for general loss distributions. 2001.

[66] P.J. Schönbucher. *Credit Derivatives Pricing Models*. Wiley Finance, 2003.

[67] M. Sellers and A. Davidson. Modelling default risk: Private firm model. *KMV Corporation*, 1998.

[68] Nelsen R. *An introduction to Copulas*. Springer-Verlag, 1999.

[69] L. David. On default correlation: a copula function approach. *Journal of Fixed income*, 9:43–45, 2000.

[70] J. C. Nash. The choleski decomposition. *Compact Numerical Methods for Computers: Linear Algebra and Function Minimisation*, 2, 1990.

[71] P. Artzner, D. Delbaen, J.M. Eber, and D. Heath. Thinking coherently. *Risk*, 10:68–71, 1997.

[72] L. Debiane, B. Ivorra, B. Mohammadi, F. Nicoud, A. Ern, T. Poinsot, and H. Pitsch. Temperature and pollution control in flames. In *Proceeding of the Summer Program*, pages 367–375, Center for Turbulence Research, NASA AMES/Stanford University, USA, 2004.

[73] L. Debiane, B. Ivorra, B. Mohammadi, F. Nicoud, A. Ern, T. Poinsot, and H. Pitsch. A low-complexity global optimization algorithm for temperature and pollution control in flames with complex chemistry. *International Journal of Computational Fluid Dynamics*, Accepted, to be published in 2006.

[74] P. Azerad, D. Isebe, B. Ivorra, B. Mohammadi, and F. Bouchette. *Optimal shape design of coastal structures minimizing coastal erosion*. CIRM, 2006.

[75] A. M. Ramos, R. Glowinski, and J. Periaux. Pointwise control of the burgers equation and related nash equilibrium problems: Computational approach. *Journal of optimization theory and applications*, 112(3):499–516, 2002.

[76] C. G. Broyden, R. Fletcher, D. Goldfarb, and D. F. Shanno. Bfgs method. *Journal of the Institute of Mathematics and Its Applications*, 6:76–90, 1970.

[77] J. Liu, D.C.and Nocedal. On the limited memory bfgs method for large-scale optimization. *Mathematical Programming*, 45:503–528, 1989.

[78] P. Abraham and J. Robbin. Transversal mappings and flows. 1967.

Résumé - Résoudre un problème d'optimisation avec une méthode découlant de la discrétisation d'un système dynamique est équivalent à résoudre un problème à valeurs aux limites. Il est alors naturel d'essayer d'appliquer la théorie sur la résolution des problèmes à valeur aux limites afin de nous aider à effectuer une optimisation de notre problème initial. Ce qui nous permet d'introduire une nouvelle classe de méthodes d'optimisation globale. Une autre idée, introduite dans ce travail, est de voir les algorithmes génétiques comme provenant de la discrétisation de systèmes dynamiques stochastiques. Ils rentrent ainsi dans le cadre d'application de notre méthodologie. Ensuite, nous appliquons certaines méthodes décrites précédemment à trois problèmes concrets d'optimisation: Synthèse de filtres optiques à base de fibres à réseaux de Bragg, optimisation de forme d'un mélangeur microfluidique utilisé pour le repliement de protéines, optimisation sous contrainte d'un portefeuille de crédits.

Mots-clés: *Optimisation globale, Systèmes dynamiques, Méthodes déterministes, Algorithmes génétiques, Gradients simplifiés, Fibres à réseaux de Bragg, Mélangeurs microfluidiques, Portefeuilles de crédits.*

Abstract - Many minimization algorithms can be viewed as discrete forms of Cauchy problems for an ordinary differential equation (ODE) or a system of ODEs in the space of control parameters. We will see that if one introduces an extra information on the infimum, these algorithms can be formulated as Boundary Value Problems (BVP) for the same equations. A motivating idea is therefore to apply algorithms solving BVPs to global optimization. It is in particular shown that GAs be interpreted as a discrete form of BVPs for a set of coupled ODEs. Therefore, the BVP analysis has also been applied to them to improve their performances leading to the construction of a new algorithm. We illustrate the previous ingredients through the following optimization problems: optical multiplexer fibers design, shape optimization of fast-microfluidic-mixer devices for protein folding, portfolio optimization under constraints.

Key-words: *Global optimization, Dynamical systems, Deterministic methods, Genetic algorithms, simplified sensitivity, Fiber Bragg gratings, Microfluidic mixer, Credit Portfolio.*

Ivorra Benjamin

Université Montpellier II
Laboratoire I3M- CNRS UMR 5149
CC 051 - Place Eugène Bataillon
34095 Montpellier Cedex 5 (France)

email: ivorra@math.univ-montp2.fr / tourbes@hotmail.com