DC programming and DCA in image processing : compressed sensing, segmentation and restoration

Thi Bich Thuy Nguyen

▶ **To cite this version:**

Thi Bich Thuy Nguyen. DC programming and DCA in image processing : compressed sensing, segmentation and restoration. Other [cs.OH]. Université de Lorraine, 2014. English. NNT : 2014LORR0350 . tel-01751600

HAL Id: tel-01751600
https://hal.univ-lorraine.fr/tel-01751600v1

Submitted on 29 Mar 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

AVERTISSEMENT

Ce document est le fruit d'un long travail approuvé par le jury de soutenance et mis à disposition de l'ensemble de la communauté universitaire élargie.

Il est soumis à la propriété intellectuelle de l'auteur. Ceci implique une obligation de citation et de référencement lors de l'utilisation de ce document.

D'autre part, toute contrefaçon, plagiat, reproduction illicite encourt une poursuite pénale.

Contact : ddoc-theses-contact@univ-lorraine.fr

LIENS

Code de la Propriété Intellectuelle. articles L 122. 4
Code de la Propriété Intellectuelle. articles L 335.2- L 335.10
http://www.cfcopies.com/V2/leg/leg_droi.php
http://www.culture.gouv.fr/culture/infos-pratiques/droits/protection.htm

THÈSE

en vue de l'obtention du titre de

DOCTEUR DE L'UNIVERSITÉ DE LORRAINE

(arrêté ministériel du 7 Août 2006)

Spécialité Informatique

présentée par

Nguyen Thi Bich Thuy

Titre de la thèse :

La programmation DC et DCA en analyse d'image
Acquisition comprimée, Segmentation et Restauration

soutenue le 11 Decembre 2014

Composition du Jury :

Président	Tao PHAM DINH	*Professeur, INSA-Rouen*
Rapporteurs	Jalal FADILI	*Professeur, Université de Caen*
	Sébastien LEFÈVRE	*Professeur, Université de Bretagne Sud*
Examinateurs	Azeddine BEGHDADI	*Professeur, Université de Paris 13*
	Olivier PIETQUIN	*Professeur, Université de Lille*
	Charles SOUSSEN	*MCF, HDR, Université de Lorraine*
Directrice de thèse	Hoai An LE THI	*Professeur, Université de Lorraine*

Thèse préparée au sein de laboratoire
d'Informatique Théorique et Appliquée (LITA)
Université de Lorraine, Metz, France

Remerciements

Je souhaite remercier en premier lieu Madame, le Professeur LE THI Hoai An, ma directrice de thèse, pour l'aide qu'elle m'a apportée, pour son soutien permanent et pour ses précieux conseils, pour les discussions que l'on a pu avoir et qui se sont toujours révélées très intéressantes et instructives. Sa grande passion et sa capacité dans la recherche fondamentale et théorique sur un large spectre de domaines d'applications ont été, et seront toujours, une grande source d'inspiration pour moi. C'est un grand plaisir et un privilège pour moi d'avoir travaillé avec elle au cours des quatre dernières années à Université de Lorraine, Metz. Je voudrais aussi sincèrement la remercier pour son intérêt pour ma vie et ma famille, en France et au Vietnam.

Je tiens à remercier particulièrement Monsieur PHAM DINH Tao, Professeur à L'INSA de Rouen pour ses conseils, et son suivi dans mes recherches. Je voudrais lui exprimer toutes ma reconnaissance pour les discussions très intéressantes qu'il a menées pour me suggérer de nouvelles voies de recherche.

Je souhaite vivement remercier Monsieur Jalal FADILI, professeur à l'Université de Caen et Monsieur Sébastien LEFÈVRE, professeur à l'Université en Informatique Université de Bretagne Sud, de m'avoir fait l'honneur d'accepter la charge de rapporteur de ma thèse , ainsi que d'avoir participé au jury.

Je souhaite également exprimer ma gratitude à Monsieur Azeddine BEGHDADI, professeur à l'Université de Paris 13, Monsieur Olivier PIETQUIN, professeur à l'Université de Lille, Monsieur Charles SOUSSEN, MCF, HDR, Université de Lorraine de participer au jury de thèse.

Je remercie particulièrement le Docteur LE Hoai Minh et le Doctorant VO Xuan Thanh pour les discussions intéressantes que nous avons eues lors de nos collaborations. Je les remercie également pour le partage dans la vie.

Mes remerciements s'adressent également au Gouvernement Vietnamien qui a financé mes études pendant trois ans.

Un grand merci à mes collègues du LITA, qui m'ont aidée pendant quatre ans. Je n'oublie

pas de remercier HUS, VNU (Hanoi University of Science, Vietnam National University, Hanoi) pour leur soutien.

Je remercie tous mes collègues français et vietnamiens rencontrés à Metz, pour les moments agréables lors de mon séjour en France. J'associe à ces remerciements tous les grands amis et collègues, dans l'ordre alphabétique, Manh Cuong, Duy Nhat, Trong Phuc, Duc Quynh, Anh Son, Ho Thanh, Ta Thuy, Tran Thuy, Anh Vu, ... pour leurs partages dans le travail et dans la vie.

Je voudrais adresser mes remerciements paticuliers à ma famille et mes fils. Je les remercie pour leur patience et leur soutien sans réserve, qui sont les pierres angulaires de ma vie de doctorat.

Enfin à tous ceux qui m'ont soutenue de près ou de loin, et à tous ceux qui m'ont incitée même involontairement, à faire mieux, veuillez trouver ici le témoignage de ma profonde gratitude.

Résumé

L'image est une des informations les plus importantes dans la vie. Avec le développement rapide des dispositifs d'acquisition d'images numériques par exemple les appareils photo numériques, les cameras de telephones, les appareils d'imagerie médicale ou les dispositifs d'imagerie satellite..., les besoins de traitement et d'analyse des images sont de plus en plus croissants. Ils concernent les problèmes de l'acquisition, du stokage des images, de l'amélioration ou de l'information d'extraction d'une image,... Dans cette thèse, nous étudions le traitement et l'analyse des problèmes: acquisition comprimée, apprentissage de dictionnaire et débruitage d'images, segmentation d'images.

La méthode que nous décrivons se base sur les approches d'optimisation déterministe, nommées la programmation DC (Difference of Convex functions) et DCA (Difference of Convex Algorithms), pour la résolution des problèmes d'analyse d'images cités précédemment.

1. Acquisition comprimée: une technique de traitement du signal pour acquérir et reconstruire un signal respectant les limites traditionnelles du théorème d'échantillonnage de Nyquist–Shannon, en trouvant la solution la plus parcimonieuse d'un système linéaire sous-déterminé. Cette méthode apporte la parcimonie ou la compressibilité du signal lorsqu'il est représenté dans une base ou un dictionnaire approprié qui permet au signal entier d'être déterminé à partir de certains mesures relatives. Dans cette thématique, nous nous intéressons à deux problèmes. Le premier est de trouver la représentation parcimonieuse d'un signal. Le second est la récupération du signal à partir de ses mesures compressées sur une base incohérente ou un dictionnaire. Les deux problèmes ci-dessus conduisent à résoudre un problème d'optimisation non convexe. Nous étudions trois modèles avec quatre approximations pour ces problèmes. Des algorithmes appropriés basés sur la programmation DC et DCA sont présentés.

2. Apprentissage du dictionnaire: Nous avons vu la puissance et les avantages de la représentation parcimonieuse des signaux dans l'acquisition comprimée. La représentation parcimonieuse d'un signal entier dépend non seulement des algorithmes de représentation mais aussi de la base ou du dictionnaire qui sont utilisés dans la représentation. Ainsi conduit un problème critique et les autres applications d'une manière naturelle. Au lieu d'utiliser une base fixe, comme wavelets (ondeletes) ou Fourier, on peut apprendre un dictionnaire, la matrice D, pour optimiser la représentation parcimonieuse d'une large classe de signaux donnés. La matrice D est appelée le dictionnaire appris. Pour ce problème, nous avons proposé un algorithme efficace basé sur DCA qui comprend deux étapes: le première étape - codage parcimonieux; le seconde étape - dictionnaire mis à jour. Une application de ce problème, débruitage d'images, est également considérée.

3. Segmentation d'images: il s'agit de partitionner une image numérique en segments multiples (ensembles des pixels). Le but de la segmentation est de simplifier et/ou de modifier la représentation d'une image en une forme qui est plus significative et plus facile à analyser. Nous avons développé une méthode efficace pour la segmentation d'images via le clustering flou avec la pondération de variables. Nous étudions également une application médicale qui est le problème de comptage de cellules. Nous proposons une combinaison de phase de segmentation et des opérations morphologiques pour compter automatiquement le nombre de cellules. Notre approche donne des résultats prometteurs dans la comparaison avec l'analyse manuelle traditionnelle en dépit de la densité cellulaire très élevée.

Abstract

Image is one of the most important information in our lives. Along with the rapid development of digital image acquisition devices such as digital cameras, phone cameras, the medical imaging devices or the satellite imaging devices..., the needs of processing and analyzing images is more and more demanding. It concerns with the problem of image acquiring, storing, enhancing or extracting information from an image,... In this thesis, we are considering the image processing and analyzing problems including: compressed sensing, dictionary learning and image denoising, and image segmentation.

Our method is based on deterministic optimization approach, named the DC (Difference of Convex) programming and DCA (Difference of Convex Algorithms) for solving some classes of image analysis addressed above.

1. Compressed sensing is a signal processing technique for efficiently acquiring and reconstructing a signal, which is breaking the traditional limits of sampling theory of Nyquist–Shannon by finding solutions to underdetermined linear systems. This takes advantage of the signal's sparseness or compressibility when it is represented in a suitable basis or dictionary, which allows the entire signal to be determined from few relative measurements. In this problem, we are interested in two aspects phases. The first one is finding the sparse representation of a signal. The other one is recovering the signal from its compressed measurements on an incoherent basis or dictionary. These problems lead to solve a NP–hard nonconvex optimization problem. We investigated three models with four approximations for each model. Appropriate algorithms based on DC programming and DCA are presented.

2. Dictionary learning: we have seen the power and the advantages of the sparse representation of signals in compressed sensing. Finding out the sparsest representation of a set of signals depends not only on the sparse representation algorithms but also on the basis or the dictionary used to represent them. This leads to the critical problems and other applications in a natural way. Instead of using a fixed basis such as wavelets or Fourier, one can learn the dictionary, a matrix D, to optimize the sparsity of the representation for a large class of given signals (data). The matrix D is called the learned dictionary. For this problem, we proposed an efficient DCA based algorithm including two stages: sparse coding and dictionary updating. An application of this problem, image denoising, is also considered.

3. Image segmentation: partitioning a digital image into multiple segments (sets of pixels). The goal of segmentation is to simplify and/or change the representation of an image into a form that is more meaningful and easier to analyze. We have developed an efficient method for image segmentation via feature weighted fuzzy clustering model. We also study an application of image segmentation for cell counting problem in medicine. We propose a combination of segmentation phase and morphological operations to automatically count the number of cells. Our approach gives promising results in comparison with the traditional manual analysis in despite of the very high cell density.

Publications

Refereed papers

- Le Thi Hoai An, Nguyen Thi Bich Thuy, Le Hoai Minh, *Sparse signal recovery by Difference of Convex functions Algorithms*, Intelligent Information and Database Systems, Lecture Notes in Computer Science, Volume 7803, pp. 387–397, Springer–Verlag, 2013. The 5-*th* Asian Conference on Intelligent Information and Database Systems (ACIIDS 2013), Kuala Lumpur, Malaysia, 18–20 March, 2013.

- Le Hoai Minh, Nguyen Thi Bich Thuy, Ta Minh Thuy, Le Thi Hoai An, *Image Segmentation via Feature Weighted Fuzzy Clustering by a DCA based algorithm*, Advanced Computational Methods for Knowledge Engineering, Studies in Computational Intelligence. Volume 479, Springer, ISSN: 1860–949X (Print) 1860–9503 (Online), pp. 53–63 (2013). The 1-*th* International Conference on Computer Science, Applied Mathematics and Applications (ICCSAMA 2013), Warsaw, Poland, 9–10 May, 2013.

- Le Thi Hoai An, Le Minh Tam, Nguyen Thi Bich Thuy, *A novel approach to automated cell counting based on a Difference of Convex functions Algorithm (DCA)*, Computational Collective Intelligence. Technologies and Applications, Lecture Notes in Computer Science Volume 8083, pp 336–345, 2013. The 5-*th* International Conference on Computational Collective Intelligence, Technologies and Applications (ICCCI 2013), Craiova, Romania, 11–13 September, 2013.

Communications in national / International conferences

- Bich Thuy Nguyen Thi, Minh Tam Le, Hoai An Le Thi, *Cell Segmentation and Counting by a DCA based approach*, presentation at EURO 2013 (the 25th European conference for Operational Research), Rome, Italy, 1/7–4/7/2013.

- Vo Xuan Thanh, Le Thi Hoai An, Nguyen Thi Bich Thuy, Pham Dinh Tao, *DC Programming and DCA for Dictionary Learning*, presentation at IFORS 2014 (the 20th Conference of the International Federation of Operational Research Societies), Barcelona, Spain, 13/7–18/7/2014.

Contents

1 Methodology **19**

1.1 Introduction . . . 19

1.2 DC programming and DCA . . . 20

1.2.1 Fundamentals of DC Annalysis . . . 20

1.2.2 DC optimality and DCA . . . 24

1.3 Conclusion . . . 26

2 Compressed sensing **27**

2.1 Introduction . . . 27

2.1.1 Development history . . . 27

2.1.2 Mathematical models . . . 30

2.1.3 Existing approaches . . . 32

2.2 Our approaches . . . 35

2.2.1 The considered models . . . 35

2.2.2 Approximations of ℓ_0–norm . . . 35

2.3 DC programming and DCA for solving the ℓ_0–norm problem . . . 40

2.3.1 Linear Constraint (LC) model . . . 41

2.3.2 Least Square Constraint (LSC) Model . . . 41

2.3.3 Regularization Least Square (RLS) model . . . 43

2.4 Convergence properties . . . 43

2.5 Numerical experiments . . . 44

2.5.1 Comparative algorithms . . . 45

2.5.2 Setup and parameters . . . 45

2.5.3 Sparse representation problem . . . 45

2.5.4 Sparse recovery problem . . . 47

2.6 Conclusion . . . 51

3 Dictionary learning and application to image denoising **63**

3.1 Introduction . . . 63

3.2 Related works . . . 64

3.3 DC Programming and DCA for Dictionary learning . . . 65

3.3.1 Sparse coding stage, update W . . . 65

3.3.2 Dictionary updating: update D . . . 70

3.4 Application to image denoising . . . 72

3.5 Conclusion . . . 76

4 Image segmentation and application to automated cell counting **97**

4.1 Image segmentation . . . 97

4.1.1 Introduction . . . 97

4.1.2 A DC formulation of the problem (4.1) . . . 99

4.1.3 DCA applied to (4.11) . . . 101

4.1.4 Finding a good starting point of DCA . . . 102

4.1.5 Computational experiments . . . 102

4.1.6 Conclusion . . . 114

4.2 Application to Cell counting problem . . . 121

4.2.1 Introduction . . . 121

4.2.2 Morphological Operations . . . 121

4.2.3 Computational experiments . . . 122

4.2.4 Conclusion . . . 123

List of Figures

2.1 Approximation of step function by a continuous function. . . . 36

2.2 Exponential (EXP) approximation of l_0–norm. . . . 37

2.3 Smoothly Clipped Absolute Deviation (SCAD) approximation of l_0–norm. . . . 37

2.4 Piecewise Linear 1 (PiL1) approximation of l_0–norm. . . . 38

2.5 Piecewise Linear 2 (PiL2) approximation of l_0–norm. . . . 39

2.6 Success rates using incoherent sensing matrix, $m = 64, n = 256$ . . . 50

2.7 Success rates using incoherent sensing matrix, $m = 100, n = 2000$. . . . 50

2.8 Success rates using highly coherent sensing matrix, $m = 100, n = 2000$. . . . 51

2.9 *Prob* 1. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc* . . . 52

2.10 *Prob* 2. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc* . . . 53

2.11 *Prob* 3. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc* . . . 54

2.12 *Prob* 4. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc* . . . 55

2.13 *Prob* 5. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc* . . . 56

2.14 *Prob* 6. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc* . . . 57

2.15 *Prob* 7. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc* . . . 58

2.16 *Prob* 8. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc* . . . 59

2.17 *Prob* 9. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc* . . . 60

2.18 *Prob* 10. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc* . . . 61

2.19 *Prob* 11. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc* . . . 62

3.1 Sample from the tested images. . . . 75
3.2 The performance of DCA with different values of the parameters λ. . . . 75
3.3 The dependence of the performance of DCA with different sizes of patches and different types of images. . . . 75
3.4 Image *1*. . . . 77
3.5 Image *2*. . . . 78
3.6 Image *3*. . . . 79
3.7 Image *4*. . . . 80
3.8 Image *5*. . . . 81
3.9 Image *6*. . . . 82
3.10 Image *7*. . . . 83
3.11 Image *8*. . . . 84
3.12 Image *9*. . . . 85
3.13 Image *10*. . . . 86
3.14 Image *11*. . . . 87
3.15 Image *12*. . . . 88
3.16 Image *13*. . . . 89
3.17 Image *14*. . . . 90
3.18 Image *15*. . . . 91
3.19 Image *16*. . . . 92
3.20 Image *17*. . . . 93
3.21 Image *18*. . . . 94
3.22 Image *19*. . . . 95
3.23 Image *20*. . . . 96

4.1 Accuracy. . . . 104
4.2 CPU Time running in seconds. . . . 105
4.3 Image 113044 . . . 106
4.4 Image 12003 . . . 107
4.5 Image 134052 . . . 108
4.6 Image 124084 . . . 109
4.7 Image 113016 . . . 110
4.8 Image 35070 . . . 111
4.9 Image 157032 . . . 112
4.10 Image Peppers . . . 115

4.11 Image Shapes 116
4.12 Image 196027 117
4.13 Image 35049 118
4.14 Image 41004 119
4.15 Image 238011 120
4.16 Some morphological operations: (a): after segmentation, (b): fill holes, (c): remove small objects, (d): overlapping cells, (e): separated cells by watershed 122
4.17 The rows show respectively: Original images, images segmented by **DCA-SI**, **SCAD**, thresholding 123
4.18 Original(a), Segmented by **DCA-SI** (b) and Count result(c) images 123

List of Tables

2.1 Some nonconvex approximations of ℓ_0-norm 34
2.2 DC decomposition $\varphi = g - h$ and calculation of ∂h. The notation $\text{sgn}(t)$ denotes sign of t. 42
2.3 The average values $||x||_0$ and $||x - x_0||$ in the noiseless model $s = Ax_0$. 46
2.4 The values $||x||_0$ and $||x - x_0||$ in the noisy model $s = Ax_0 + e$ 46
2.5 The values of $||x||_0$, $d = ||x - x_0||$ on 11 noiseless datatest. 47
2.6 The values of $||x||_0$, $d = ||x - x_0||$ on 11 noisy datatest. 48

3.1 The denoising image and PSNR (dB) of DCA v.s K-SVD and PALM 74

4.1 The results of PWCO(%) of 4 methods 103
4.2 CPU Time running in seconds 104
4.3 The value of Borsotti function $Q(I)$ 113
4.4 Performance time (seconds) 114
4.5 Cell counting results 122

Introduction générale

Cadre général et nos motivations

L'image est une des informations les plus importantes dans la vie. Avec le développement rapide des dispositifs d'acquisition d'images numériques par exemple les appareils photo numériques, les cameras de telephones, les appareils d'imagerie médicale ou les dispositifs d'imagerie satellite..., les besoins de traitement et d'analyse des images sont de plus en plus croissants. Nous devons donc proposer des nouvelles approches plus rapides, plus efficaces, pouvant traiter des données de grande taille. Dans ce contexte, nous avons utilisé une approche d'optimisation déterministe nommée la programmation DC (Difference of Convex functions) et DCA (Difference of Convex Algorithms), pour la résolution de certaines classes de problèmes en image, acquisition comprimée, restauration et segmentation.

On peut distinguer deux branches d'optimisation déterministe: la programmation convexe et la programmation non convexe. Un problème d'optimisation convexe consiste en la minimisation d'une fonction convexe (Objectif) sous des contraintes convexes. Lorsque la double convexité dans la fonction Objectif et dans les contraintes n'est pas vérifiée, nous sommes face à un problème d'optimisation non convexe. La double convexité d'un programme convexe permet d'établir des caractérisations (sous forme de conditions nécessaires et suffisantes) de solutions optimales et ainsi de construire des méthodes itératives convergeant vers des solutions optimales. Théoriquement, on peut résoudre tout programme convexe, mais encore faut-il bien étudier la structure du programme convexe en question pour proposer des variantes performantes peu couteuses et donc capables d'atteindre des dimensions réelles très importantes.

L'absence de cette double convexité rend la résolution d'un programme non convexe difficile voire impossible avec les méthodes actuelles. Contrairement à la programmation convexe, les solutions optimales locales et globales sont à distinguer dans un programme non convexe. D'autre part, si l'on dispose des caractérisations d'optimalité locale utilisables, au moins pour la classe des programmes non convexes assez réguliers, qui permettent la construction des méthodes convergeant vers des solutions locales, (algorithmes locaux) il n'y a, par contre, pas de caractérisations d'optimalité globale sur lesquelles sont basées les méthodes itératives convergeant vers des solutions globales (algorithmes globaux). L'analyse et l'optimisation convexes modernes se voient ainsi contraintes à une extension logique et naturelle à la non convexité et la non différentiabilité.

Les méthodes numériques conventionnelles de l'optimisation convexe ne fournissent que des minima locaux bien souvent éloignés de l'optimum global.

L'optimisation non convexe connaît une explosion spectaculaire depuis les années 90. En effet, dans les milieux industriels, on a commencé à remplacer les modèles convexes par des modèles non convexes. Ces derniers sont plus complexes mais plus fiables et présentent mieux la nature des problèmes étudiés. Durant ces dernières années, la recherche en optimisation non convexe a largement bénéficié des efforts des chercheurs et s'est enrichie de nouvelles approches. Il existe deux approches différentes mais complémentaires en programmation non convexe:

1. Approches globales combinatoires: basées sur les techniques combinatoires de la Recherche Opérationnelle. Elles consistent à localiser les solutions optimales à l'aide des méthodes d'approximation, des techniques de coupe, des méthodes de décomposition, de séparation et d'évaluation. Elles ont connu de très nombreux développements importants au cours de ces dernières années à travers les travaux de H. Tuy (Horst and Tuy [1996]) (reconnu comme le pionnier), R. Horst, H. Benson, P.M. Pardalos, H.Konno, Le Dung Muu, Le Thi Hoai An, and Pham Dinh Tao. L'inconvénient majeur des méthodes globales est leur lourdeur (encombrement en place-mémoire) et leur coût trop important. Elles ne sont pas applicables aux problèmes d'optimisation non convexes réels qui sont souvent de très grande dimension.
2. Approches locales et globales d'analyse convexe: basées sur l'analyse et l'optimisation convexe. Ici la programmation DC et DCA joue un rôle central, car la plupart des problèmes d'optimisation non convexe sont formulés/reformulés sous la forme DC. Sur le plan algorithmique, l'essentiel repose sur les algorithmes de l'optimisation DC (DCA) introduits par Pham Dinh Tao en 1985 à l'état préliminaire et développés intensivement à travers de nombreux travaux communs de Le Thi Hoai An et Pham Dinh Tao depuis 1993 pour devenir maintenant classiques et de plus en plus utilisés par des chercheurs et praticiens de par le monde, dans différents domaines des sciences appliquées (pour ne citer que quelques-uns, voir la liste des références dans le site: http://www.lita.univ-lorraine.fr/∼lethi/)

La programmation DC et DCA considère le problème DC de la forme:

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

où g et h sont des fonctions convexes définies sur $\mathbb{R}^n$ et à valeurs dans $\mathbb{R} \cup \{+\infty\}$, semi-continues inférieurement et propres. La fonction f est appelée fonction DC avec les composantes DC g et h, et $g - h$ est une décomposition DC de f. DCA est basée sur la dualité DC et des conditions d'optimalité locale. La construction de DCA implique les composantes DC g et h et non la fonction DC f elle-même. Or, chaque fonction DC admet une infinité des décompositions DC qui influencent considérablement sur la qualité (la rapidité, l'efficacité, la globalité de la solution obtenue,...) de DCA. Ainsi, du point de vue algorithmique, la recherche d'une "bonne" décomposition DC et d'un "bon" point initial est très importante dans le développement de DCA pour la résolution d'un programme DC.

Les travaux de cette thèse sont basés sur la programmation DC et DCA. Cette démarche est justifiée par de multiples arguments (Le Thi [2012a], Pham Dinh and Le Thi [2014]):

- Les approches d'optimisation globale telles que Branch et Bound, le plan de Coupe,... pour les programmes DC ne fonctionnent pas dans les programmes DC à grande échelle auxquels nous sommes souvent confrontés.
- DCA est une philosophie plutôt qu'un algorithme. Pour chaque problème, nous pouvons concevoir une famille d'algorithmes basés sur DCA. La flexibilité de DCA sur le choix de la décomposition DC peut offrir des schémas DCA plus performants que des méthodes standards.
- L'analyse convexe fournit des outils puissants pour prouver la convergence de DCA dans un cadre général. Ainsi, tous les algorithmes basés sur DCA bénéficient (au moins) des propriétés de convergences générales du schéma DCA générique qui ont été démontrées.
- DCA est une méthode efficace, rapide et évolutive pour la programmation non convexe. A notre connaissance, DCA est l'un des rares algorithmes de la programmation non convexe, non différentiable qui peut résoudre des programmes DC de très grande dimension. Les programmations DC et DCA ont été appliquées avec succès pour la modélisation DC et la résolution de nombreux et divers problèmes d'optimisation non convexes dans différents domaines des sciences appliquées (voir par exemple la liste des références dans http://www.lita.univ-lorraine.fr/∼lethi/)

Il est important de noter qu'avec les techniques de reformulation en programmation DC et les

décompositions DC appropriées, on peut retrouver la plupart des algorithmes existants en programmation convexe/non convexe comme les cas particuliers de DCA.

Nos contributions

Dans ce travail, nous nous intéressons particulièrement aux trois problèmes de traitement d'image: acquisition comprimée, apprentissage du dictionnaire (dictionary learning) et segmentation de l'image.

Nous considérons deux problèmes dans le domaine de l'acquisition comprimée (Compressed sensing) qui sont: la représentation et la reconstruction parcimonieuses. Comme dans certaines recherches récentes de la théorie d'acquisition comprimée, nous avons vu la puissance et les avantages des représentations parcimonieuses. L'acquisition comprimée est basée sur le fait critique que nous pouvons représenter beaucoup de signaux ou d'images en utilisant seulement quelques coefficients non-nuls dans une base ou un dictionnaire approprié. Nous récupérons ensuite ces signaux ou images avec un petit nombre de mesures linéaires par résolution un système mal posé $Ax = b$, $A \in \mathbb{R}^{m\times n}, m \ll n$. Avec la condition ci-dessus, les signaux ou les images sont parcimonieuses. Ce problème est transformé en un problème d'optimisation non linéaire.

Ces deux problèmes conduisent à un problème d'optimisation non-convexe NP–difficile (NP–hard). L'approche standard ℓ_1–norme permet normalement de résoudre ces types de problèmes. Mais il y a quelques hypothèses qui peuvent ne pas être satisfaites dans de nombreux cas, tels que certaines expériences concernant des problèmes d'apprentissage automatique signalés dans Bradley and Mangasarian [1998]. Ils montrent qu'une approche basée sur l'optimisation concave donne de meilleurs résultats que celle basée sur la ℓ_1–norme.

Dans ce contexte, nous étudions l'approches de ℓ_0–norme par des fonctions non convexes, continues qui donnent des résultats plus précis.

Heureusement, l'efficacité de la programmation DC et DCA nous permet d'utiliser quatre approximations de ℓ_0–norme pour trois modèles respectivement: contraintes linéaires, contraintes moindres carrées et régularisations moindres carrées. Par conséquent, nous étudions 12 algorithmes pour résoudre ces problèmes. Pour évaluer l'efficacité de ces algorithmes, nous effectuons la comparaison avec cinq algorithmes connus qui concernent la norme ℓ_0 et la norme ℓ_1.

Le deuxième problème traité dans cette thèse est l'apprentissage du dictionnaire. Nous proposons un algorithme qui comporte deux phases: la première phase est le problème de codage parcimonieux et la deuxième phase est le problème de la mise à jour du dictionnaire. Une application de réduction de bruit d'image est testée pour vérifier l'efficacité de l'algorithme proposé.

Le dernier problème traité dans cette thèse est la segmentation d'image. Nous proposons une approche de segmentation par clustering flou avec pondération de variables. Habituellement, les variables peuvent être divisées en trois catégories: les variables pertinentes, redondantes et non pertinentes. Les variables pertinentes sont essentielles pour le processus de classification, les variables redondantes n'apportent aucune nouvelle information à classifier, tandis que les variables non pertinentes ne fournissent aucune information utile. Chaque variable est affectée à une valeur continue dans l'intervalle $[0, 1]$, et est nommée "un poids" (les variables pertinentes auront un poids élevé). Nous nous basons sur l'approche DC, développée dans Le Thi et al. [2007c]. Nous proposons une reformulation en tant que programmes DC. Puis, une décomposition DC appropriée est effectuée. Dans nos expériences, nous comparons les résultats de notre algorithme avec trois autres algorithmes, **SCAD** (Frigui and Nasui [2004]) - un algorithme de clustering flou avec pondération de variables, **DCAFCM** (Le Thi et al. [2007c]) et **FCM** (Bezdek [1981]) - deux algorithmes avec le modèle flou. Les résultats montrent l'efficacité de la pondération de variables, ce qui permet d'améliorer la performance de la tâche de segmentation. En outre, les expériences montrent la supériorité de calcul de notre algorithme par rapport aux autres algorithmes.

Nous appliquons notre approche à une application médicale, qui est le problème de comptage de cellules. Ceci est une tâche importante dans le diagnostic de nombreuses maladies, mais le comptage automatique n'est pas facile. En combinant avec des opérations morphologiques, nous introduisons une méthode efficace pour résoudre ce problème. Les tests sont réalisés pour des ensembles de données réelles, nous effectuons trois algorithmes et les comparons avec les résultats réels obtenus par méthode comptage manuel. Les résultats sont très prometteurs et montrent l'efficacité de notre approche.

Organisation de la thèse

La thèse est composée de quatre chapitres. Le premier chapitre décrit de manière succincte la programmation DC et DCA. Il présente les outils théoriques et algorithmiques servant de références aux autres chapitres. Chacun des trois chapitres suivants est consacré à la présentation d'une classe de problèmes abordés ci–dessus.

Le second chapitre est consacré aux deux problèmes de l'acquisition comprimée: représentations parcimonieuses et reconstruction parcimonieuse.

L'apprentissage du dictionnaire est traité dans le troisième chapitre, avec une application réelle pour le débruitage d'images.

Nous abordons enfin la segmentation de l'image en proposant une approche basée sur DCA et son application au problème de comptage des cellules dans ce dernier chapitre.

Chapter 1

Methodology

1.1 Introduction

DC programming and DCA, which constitute the backbone of nonconvex programming and global optimization, were introduced by Pham Dinh Tao in their preliminary form in 1985 (Pham Dinh [1986], Pham Dinh and Bernoussi [1988]), and extensively developed by Le Thi Hoai An and Pham Dinh Tao since 1994 to become now classic and increasingly popular (see e.g. the list of reference in: http://www.lita.univ-lorraine.fr/∼lethi/). These theoretical and algorithmic tools are applied with great success in different renowned laboratories (to name a few: Princeton, Stanford, MIT, Berkeley, Carnegie Mellon, Cornell, Imperial College, Institut für Allgemeine Mechanik (IAM, RWTH-Aachen), California, Mannheim, Heidelberg, Courant Institute of Mathematical Sciences, Minnesota, The University of North Carolina at Chapel Hill, Michigan, Iowa, Florida, Tokyo Institute of Technology, Fribourg, EPFL, National-ICT-Australia-(NICTA), University of Sydney, Fudan, The Chinese University of Hongkong, Hanoi Institute of Mathematics, Coimbra, Vienna, Copenhague, Louvain, Pukyong, Namur, Google, Yahoo, Nasa, Siemens, etc) for modeling and solving nonconvex programs.

The theoretical results on DC Programming and DCA can be found in Le Thi [1994, 1997], Pham Dinh and Le Thi [1997, 1998], Le Thi et al. [1999], Le Thi and Pham Dinh [2002, 2005], Le Thi et al. [2012a] and Pham Dinh and Le Thi [2014] while numerical methods based on DCA for solving difficult problems such as bilevel programming problems, nonconvex quadratic programming problems, binary quadratic programming problems, optimization over efficient / weakly efficient set, trust region subproblems, linear complementarity problems, etc, can be found in various joint works of Le Thi Hoai An and Pham Dinh Tao (see e.g. Le Thi and Pham Dinh [1997], Le Thi et al. [2002, 2012b], Pham Dinh and Le Thi [1998] and the list of references in Le Thi [home page]).

DC Programming and DCA have been successfully used by researchers and practitioners to model and solve their nonconvex programs from many fields of Applied Science, whose Data Mining and Machine Learning remains, to date, the domain of predilection. In particular, DCA is widely used in various areas of unsupervised learning (Pan et al. [2013], Le Thi et al. [2007a,b], Le Hoai et al. [2013b], Le Thi et al. [2014c, 2007b,c, 2008a], Le Hoai and Ta [2014], Ta [2014]), supervised learning (Wang et al. [2010], Le Thi et al. [2013e, 2008e, 2013b, 2008b,c]), semi–supervised learning (Wang et al. [2009], Tian et al. [2012], Yang and Wang [2013], Le Hoai et al. [2013a]), learning with sparsity/uncertainty (Le Thi et al. [2013e], Cheng Soon and Le Thi [2013], Phan et al. [2014], Le Thi et al. [2013c], Thiao et al. [2010]), in dictionary learning (Fawzi et al. [2014]) ... For other domains, to cite a few, in finance - risk management, portfolio selection (Mokhtar et al. [2014], Mohri and Medina [2014], Le Thi and Moeini [2014], Le Thi et al. [2012b], Le Thi and Tran [2012], Le Thi et al. [2009a,b, 2014a], Pham Dinh et al. [2014], Pham et al. [2013], Le Thi and Moeini [2006],

Pham Dinh et al. [2009], Nguyen et al. [2011]); in transport-logistic (Zhong and Aghezzaf [2011, 2009], Ndiaye et al. [2008]), in communication systems - routing (Ta et al. [2012a], Nguyen and Le Thi [2011], Ta et al. [2010a, 2012b, 2010b], Ta [2012], Nguyen [2012], Pham [2013], Le Thi and Pham Dinh [2014]), wireless networks (Wu et al. [2014], Vucic et al. [2010], Le Thi et al. [2008d]); in production management - supply chain design, scheduling (Le Thi et al. [2007d], Nguyen et al. [2007], Le Thi et al. [2009d], Le Hoai et al. [2012], Nguyen and Le Thi [2011], Le Thi et al. [2013d, 2009c], Le Thi and Tran [2014]); in bioinformatics (Ying et al. [2009], Le Thi et al. [2008b, 2013a, 2008e]); in data security - cryptography, anomaly detection (Le Hoai et al. [2008, 2010], Le Thi et al. [2009e, 2014b]); in computer vision and image/signal processing (Weber et al. [2006], Schnörr [2007], Kokiopoulou et al. [2009], Gasso et al. [2009], Le Hoai et al. [2013c], Le Thi et al. [2008a]). See Le Thi [home page] for a more complete (but not exhaustive) list.

In this chapter we will present a brief introduction of DC (Difference of Convex functions) programming and DCA (DC Algorithms) which constitutes the backbone of our work.

1.2 DC programming and DCA

The materials presented in this section are extracted from Le Thi [1997] and Pham Dinh and Le Thi [1997].

Their original key idea relies on the structure DC of objective function and constraint functions in nonconvex programs which are explored and exploited in a deep and suitable way. *The resulting DCA introduces the nice and elegant concept of approximating a nonconvex (DC) program by a sequence of convex ones: each iteration of DCA requires solution of a convex program.*

Their popularity resides in *their rich, deep and rigorous mathematical foundations, and the versality, flexibility, robustness, inexpensiveness and efficiency of DCA's compared to existing methods, their adaptation to specific structures of addressed problems and their ability to solve real-world large-scale nonconvex programs.* Recent developments in convex programming are mainly devoted to reformulation techniques and scalable algorithms in order to handle large-scale problems. Obviously, they allow for enhancement of DC programming and DCA in high dimensional nonconvex programming.

For beginning, let us present some fundamental notations of convex analysis and DC programming

1.2.1 Fundamentals of DC Annalysis

Definitions and properties

This paragraph is devoted to a brief recall of convex analysis to facilitate the reading of its content. For more details, we refer to the work of Laurent [1972], of Rockafellar [1970] and of Auslender [1976].

Let X be the Euclidean space $\mathbb{R}^n$, equipped with the canonical inner product $\langle ., . \rangle$ and its Euclidean norm $\|x\| = \langle x, x \rangle^{\frac{1}{2}}$. The dual vector space of X is denoted by Y, which can be identified with X itself. We use the basic tools of modern convex analysis where a function can take the infinity value $+\infty$ (Rockafellar [1970]). For $f : X \longrightarrow \mathbb{R} \cup \{+\infty\}$, its effective domain of f, denoted by $\mathrm{dom}(f)$, is

$$\mathrm{dom}(f) = \{x \in X : f(x) < +\infty\} \tag{1.1}$$

and the epigraph of f, denoted by $\mathrm{epi}(f)$, is

$$\mathrm{epi}(f) = \{(x, \lambda) \in X \times \mathbb{R} : f(x) \leq \lambda\}.$$

If $\mathrm{dom}(f) \neq \emptyset$ then we say that the function f is *proper*.

A proper function $f :\longrightarrow \mathbb{R} \cup \{+\infty\}$ is called *convex* if its epigraph is a convex set in $X \times \mathbb{R}$. This is equivalent to

$$f((1-\lambda)x^1 + \lambda x^2) \leq (1-\lambda)f(x^1) + \lambda f(x^2), \forall x^1, x^2 \in X \ , \ \forall \lambda \in]0,1[. \tag{1.2}$$

It amounts to say that the finite function f is convex on its effective domain. In the sequel $\mathrm{Conv}(X)$ denotes the set of convex proper functions on X. It is clear that $\mathrm{Conv}(X)$ is a convex cone with apex at the origin.

In (1.2), if the strict inequality holds, for all $x^1, x^2 \in \mathrm{dom}\ (f)$ with $x^1 \neq x^2$ then f is called *strictly convex* function on $\mathrm{dom}\ (f)$.

A proper function f is called *strongly convex* on a convex set $C \subset \mathrm{dom}\ (f)$ if there exists a number $\rho > 0$ such that

$$f((1-\lambda)x^1 + \lambda x^2) \leq (1-\lambda)f(x^1) + \lambda f(x^2) - (1-\lambda)\lambda\frac{\rho}{2}\|x^1 - x^2\|^2, \tag{1.3}$$

for all $x^1, x^2 \in C$, and for all $\lambda \in]0,1[$. It is equivalent to saying that $f - \frac{\rho}{2}\|.\|^2$ is convex on C. The *modulus of strong convexity* of f on C, denoted by $\rho(f, C)$ or $\rho(f)$ if $C = X$, is given by

$$\rho(f, C) = Sup\{\rho \geq 0 : f - \frac{\rho}{2}\|.\|^2 \text{ is convex on } C\} > 0. \tag{1.4}$$

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

Remark 1.1 *f strongly convex $\Longrightarrow$ f strictly convex $\Longrightarrow$ f convex.*

Let f be a convex proper function on X, a vector $y_0 \in Y$ is called a *subgradient* of f at a point $x_0 \in \mathrm{dom}(f)$ if

$$\langle y_0, x - x_0\rangle + f(x_0) \leq f(x) \quad \forall x \in X.$$

The set of all subgradients of f at x_0 is called the *subdifferential* of f at x_0 and is denoted by $\partial f(x_0)$. Let $\epsilon > 0$, a vector y_0 is called ϵ-subgradient of f at point x_0 if

$$\langle y_0, x - x_0\rangle + f(x_0) \leq f(x) + \epsilon \quad \forall x \in X.$$

Then the set of all ϵ-subgradients of f at point x_0 is called the the ϵ-subdifferential of f at x_0 and is denoted by $\partial_\epsilon f(x_0)$.

A proper function $f : X \longrightarrow \mathbb{R}$ is called *lower semi-continuous* (l.s.c) at a point $x_0 \in X$ if

$$\lim_{x \to x_0} \inf f(x) \geq f(x_0).$$

or equivalently

$$\forall \epsilon > 0\ \exists\ \eta > 0 \text{ such that } \|x - x_0\| \leq \eta \Longrightarrow f(x) \geq f(x_0) - \epsilon$$

Let $\Gamma_0(X)$ be the set of all l.s.c proper convex functions on X.

Definition 1.1 *The* conjugate function *$f^\star$ of $f \in \Gamma_0(X)$ is defined by*

$$f^\star(y) = \sup\{\langle x, y\rangle - f(x) : x \in X\}. \tag{1.5}$$

i.e., $f^\star$ is the pointwise supremum of the family of (continuous) affine functions $y \mapsto \langle x, y\rangle - f(x)$ on Y.

The function f is polyhedral convex if it is the sum of a pointwise supremum of a finite collection of affine functions and the indicator function of a nonempty polyhedral convex set.

The main properties are summarized in the following proposition that will be needed for further:

Proposition 1.1 *(Rockafellar [1970])*
If $f \in \Gamma_0(X)$ then:

- $f \in \Gamma_0(X) \Longleftrightarrow f^\star \in \Gamma_0(Y)$. *Moreover* $f = f^{\star\star}$,
- $y \in \partial f(x) \Longleftrightarrow f(x) + f^\star(y) = \langle x, y\rangle$ *and* $y \in \partial f(x) \Longleftrightarrow x \in \partial f^\star(y)$,
- $\partial f(x)$ *is a closed convex set, for any* $x \in dom(f)$,
- $\partial f(x)$ *is equal to a singleton* $\{y\}$ *iff* f *is differentiable at* x *and* $\nabla f(x) = y$,
- $f(x_0) = min\{f(x), x \in X\} \Longleftrightarrow 0 \in \partial f(x_0)$.

DC functions: A function $f : \Omega \mapsto \mathbb{R}$ defined on a convex set convex $\Omega \subset \mathbb{R}^n$ is called DC on Ω if it can be presented in the form of difference of two convex functions on Ω, i.e.

$$f(x) = g(x) - h(x),$$

where g and h are convex functions on Ω, $g - h$ is called a DC decomposition of f. We denote by $DC(\Omega)$ be the set of all DC functions on Ω.

DC functions have many important properties, in particular $DC(\Omega)$ is closed with respect to frequently used operations in optimization. Specifically

Proposition 1.2 *(Le Thi [1997], Pham Dinh and Le Thi [1997])*

(i) $DC(\Omega)$ *is a vector space spanned by the convex cone* $Conv(\Omega) : DC(\Omega) = Conv(\Omega) - Conv(\Omega)$.
(ii) The pointwise supremum of a finite collection of finite DC functions on Ω *is DC on* Ω,
The pointwise infimum of a finite collection of finite DC functions on Ω *is DC on* Ω,
(iii) Let $f \in DC(\Omega)$, *then* $|f(x)|, f^+(x) = max\{0, f(x)\}$ *and* $f^-(x) = min\{0, f(x)\}$ *belong to* $DC(\Omega)$.

These results have been generalized to convex functions $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ (Le Thi [1997]).

Remark 1.2 *Given a DC function* f *and a DC decomposition* $f = g - h$, *then for any finite convex function* φ, $f = (g + \varphi) - (h + \varphi)$ *gives another DC decomposition of* f. *Thus, a DC function has infinitely many DC decompositions.*

Classification of nonconvex programs

Due to the preponderance and wealthy properties of DC functions, the passage from Conv(Ω) to the vector space $DC(\Omega)$ permits to expand significantly convex programming to nonconvex programming. The field of optimization problems involving DC functions is very large and covers most of problems encountered in applications.

However, we cannot deal with any nonconvex differentiable/nondifferentiable program. The following classification is well-known:

(1) $\sup\{f(x) : x \in C\}$, where f and C are convex

(2) $\inf\{g(x) - h(x) : x \in X\}$, where g and h are convex

$$(3) \quad \inf\{g(x) - h(x) : x \in C, f_1(x) - f_2(x) \leq 0\},$$

where g, h, f_1, f_2 and C are convex, these seem to be large enough to contain substantially all nonconvex programs encountered in real life. Problem (1) is a special case of Problem (2) with $g = \chi_C$, the indicator function of C and $h = -f$. Problem (2) can be modified in the form equivalent to (1)

$$\inf\{t - h(x) : g(x) - t \leq 0\}.$$

While Problem (3) can be transformed into the form (2) by using exact penalty related to the DC constraints $f_1(x) - f_2(x) \leq 0$. Its solution can also be reduced, under certain technical conditions, to that of a sequence of Problems (1). Problem (2) is called *a DC program*. It is a major interest both from practical and theoretical point of view. From the theoretical point of view, we can note that, as remarked above, the class of DC functions is remarkably closed with respect to operations frequently used in optimization. Moreover, there is an elegant duality theory (Pham Dinh [1975, 1976], Toland [1978], Urruty [1986], Le Thi [1994, 1997], Le Thi and Pham Dinh [1997]) which, as with Lagrangian duality in convex optimization, has profound practical implications for numerical methods.

DC programming and DCA (DC Algorithms) were introduced by Pham Dinh Tao (Pham Dinh [1986], Pham Dinh and Bernoussi [1988]) in their preliminary form. In fact, DCA is a generalization of subgradient algorithms which were studied by the same author on convex maximization (Pham Dinh [1975, 1986]). These theoretical and algorithmic tools are extensively developed by Le Thi Hoai An and Pham Dinh Tao since 1994 (see e.g. Le Thi [1994, 1997], Le Thi and Pham Dinh [1997], Pham Dinh and Le Thi [1997, 1998], Le Thi and Pham Dinh [2001], Le Thi et al. [2002], Le Thi and Pham Dinh [2003, 2005] and Le Thi et al. [2005, 2012a], Pham Dinh and Le Thi [2014], Le Thi [home page]) to become now classic and increasingly popular.

DC Duality

In convex analysis, the concept of duality (conjugate function, dual problem, etc.) is a very powerful fundamental concept. For convex programs and in particular linear, a duality theory has been developed over several decades (Rockafellar [1970]). More recently, an important concept of duality in nonconvex analysis has been proposed and developed, first for convex maximization problems, before reaching the DC programming. DC duality introduced by Toland (1978) can be regarded as a natural and logical generalization of earlier works of Pham Dinh Tao (1975) on convex maximization. We will present below the main results on optimal conditions (local and global) and the DC duality. For more details, the reader is referred to the document of Le Thi and Pham Dinh [1997].

A standard DC program is of the form ($g, h \in \Gamma_0(X)$)

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

DC duality associates a primal DC program with its dual, which is also a DC program with the same optimal value,

$$\alpha = \inf\{h^*(y) - g^*(y) : y \in \mathbb{R}^n\} \quad (D_{dc})$$

by using the fact that every function $\varphi \in \Gamma_0(\mathbb{R}^n)$ is characterized as a pointwise supremum of a collection of affine functions, say

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

There is a perfect symmetry between (P_{dc}) and its dual (D_{dc}): the dual of (D_{dc}) is exactly (P_{dc}).

A standard DC program with a convex constraint C (a nonempty closed convex set in $\mathbb{R}^n$)

$$\alpha = \inf\{f(x) := g(x) - h(x) \ : \ x \in C\} \tag{1.6}$$

can be expressed in the form of (P_{dc}) by adding the indicator function χ_C of C ($\chi_C(x) = 0$ if $x \in C, +\infty$ otherwise) to the function g. The vector space of DC functions, $DC(\mathbb{R}^n) = \Gamma_0(\mathbb{R}^n) - \Gamma_0(\mathbb{R}^n)$, forms a wide class encompassing most real-life objective functions and is closed with respect to usual operations in optimization. DC programming constitutes so an extension of convex programming, sufficiently large to cover most nonconvex programs (Pham Dinh and Le Thi [1997, 1998], Le Thi and Pham Dinh [2003, 2005], Le Thi [home page] and references quoted therein), but not too in order to leverage the powerful arsenal of the latter.

1.2.2 DC optimality and DCA

Polyhedral DC program is a DC program in which at least one of the functions g and h is polyhedral convex. Polyhedral DC programming, which plays a central role in nonconvex optimization and global optimization and is the foundation of DC programming and DCA, has interesting properties (from both a theoretical and an algorithmic point of view) on local optimality conditions and the finiteness of DCA's convergence.

DC programming investigates the structure of $DC(\mathbb{R}^n)$, DC duality and local and global optimality conditions for DC programs. The complexity of DC programs clearly lies in the distinction between local and global solution and, consequently, the lack of verifiable global optimality conditions.

We have developed necessary local optimality conditions for the primal DC program (P_{dc}), by symmetry those relating to dual DC program (D_{dc}) are trivially deduced

$$\partial h(x^*) \cap \partial g(x^*) \neq \emptyset \tag{1.7}$$

(such a point x^* is called critical point of $g - h$ or (1.7) a generalized Karusk-Kuhn-Tucker (KKT) condition for (P_{dc})), and

$$\emptyset \neq \partial h(x^*) \subset \partial g(x^*). \tag{1.8}$$

The condition (1.8) is also sufficient (for local optimality) in many important classes of DC programs. In particular it is sufficient for the next cases quite often encountered in practice:

– In polyhedral DC programs with h being a polyhedral convex function. In this case, if h is differentiable at a critical point x^*, then x^* is actually a local minimizer for (P_{dc}). Since a polyhedral convex function is differentiable everywhere except for a set of measure zero, one can say that a critical point x^* is almost always a local minimizer for (P_{dc}).

The transportation of global solutions between (P_{dc}) and (D_{dc}) is expressed by:

$$[\bigcup_{y^* \in \mathcal{D}} \partial g^*(y^*)] \subset \mathcal{P} \ , \ [\bigcup_{x^* \in \mathcal{P}} \partial h(x^*)] \subset \mathcal{D} \tag{1.9}$$

where $\mathcal{P}$ and $\mathcal{D}$ denote the solution sets of (P_{dc}) and (D_{dc}) respectively. The first (second) inclusion becomes equality if the function h (resp. g^*) is subdifferentiable on $\mathcal{P}$ (resp. $\mathcal{D}$). They show that solving a DC program implies solving its dual. Note also that, under technical conditions, this transportation also holds for local solutions of (P_{dc}) and (D_{dc}) (Pham Dinh and Le Thi [1997, 1998], Le Thi and Pham Dinh [2003, 2005], Le Thi [home page] and references quoted therein).

Based on local optimality conditions and duality in DC programming, the DCA consists in constructing of two sequences $\{x^k\}$ and $\{y^k\}$ of trial solutions of the primal and dual programs respectively, such that the sequences $\{g(x^k) - h(x^k)\}$ and $\{h^*(y^k) - g^*(y^k)\}$ are decreasing, and $\{x^k\}$ (resp. $\{y^k\}$) converges to a primal feasible solution x^* (resp. a dual feasible solution y^*) satisfying local optimality conditions and

$$x^* \in \partial g^*(y^*), \quad y^* \in \partial h(x^*). \tag{1.10}$$

The sequences $\{x^k\}$ and $\{y^k\}$ are determined in the way that x^{k+1} (resp. y^{k+1}) is a solution to the convex program (P_k) (resp. (D_{k+1})) defined by ($x^0 \in$ dom ∂h being a given initial point and $y^0 \in \partial h(x^0)$ being chosen)

$$(P_k) \quad \inf\{g(x) - [h(x^k) + \langle x - x^k, y^k\rangle] : x \in \mathbb{R}^n\}, \tag{1.11}$$

$$(D_{k+1}) \quad \inf\{h^*(y) - [g^*(y^k) + \langle y - y^k, x^{k+1}\rangle] : y \in \mathbb{R}^n\}. \tag{1.12}$$

The DCA has the quite simple interpretation: at the k-th iteration, one replaces in the primal DC program (P_{dc}) the second component h by its affine minorization $h^{(k)}(x) := h(x^k)+\langle x-x^k, y^k\rangle$ defined by a subgradient y^k of h at x^k to give birth to the primal convex program (P_k), the solution of which is nothing but $\partial g^*(y^k)$. Dually, a solution x^{k+1} of (P_k) is then used to define the dual convex program (D_{k+1}) obtained from (D_{dc}) by replacing the second DC component g^* with its affine minorization $(g^*)^{(k)}(y) := g^*(y^k) + \langle y - y^k, x^{k+1}\rangle$ defined by the subgradient x^{k+1} of g^* at y^k : the solution set of $(D_{k+1})$is exactly $\partial h(x^{k+1})$. The process is repeated until convergence. DCA performs a double linearization with the help of the subgradients of h and g^* and the DCA then yields the next scheme: (starting from given $x^0 \in$ dom ∂h)

$$y^k \in \partial h(x^k); \quad x^{k+1} \in \partial g^*(y^k), \ \forall k \geq 0. \tag{1.13}$$

DCA's convergence properties:

DCA is a descent method without linesearch, but with global convergence, which enjoys the following properties: (C and D are two convex sets in $\mathbb{R}^n$, containing the sequences $\{x^k\}$ and $\{y^k\}$ respectively).

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

- $g(x^{k+1}) - h(x^{k+1}) = g(x^k) - h(x^k)$ iff $y^k \in \partial g(x^k) \cap \partial h(x^k), y^k \in \partial g(x^{k+1}) \cap \partial h(x^{k+1})$ and $[\rho(g, C) + \rho(h, C)]\|x^{k+1} - x^k\| = 0$. Moreover if g or h are strictly convex on C then $x^k = x^{k+1}$.
  In such a case DCA terminates at the k^{th} iteration (finite convergence of DCA)
- $h^*(y^{k+1}) - g^*(y^{k+1}) = h^*(y^k) - g^*(y^k)$ iff $x^{k+1} \in \partial g^*(y^k) \cap \partial h^*(y^k), x^{k+1} \in \partial g^*(y^{k+1}) \cap \partial h^*(y^{k+1})$ and $[\rho(g^*, D) + \rho(h^*, D)]\|y^{k+1} - y^k\| = 0$. Moreover if g^* or h^* are strictly convex on D, then $y^{k+1} = y^k$.
  In such a case DCA terminates at the k^{th} iteration (finite convergence of DCA).

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

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

iv) DCA has a linear convergence for general DC programs.

v) DCA has a finite convergence for polyhedral DC programs.

DCA's distinctive feature relies upon the fact that DCA deals with the convex DC components g and h but not with the DC function f itself. DCA is one of the rare algorithms for nonconvex nonsmooth programming. Moreover, a DC function f *has infinitely many DC decompositions which have crucial implications for the qualities* (convergence speed, robustness, efficiency, globality of computed solutions,...) of DCA. For a given DC program, the choice of *optimal* DC decompositions is still open. Of course, this depends strongly on the very specific structure of the problem being considered. In order to tackle the large-scale setting, one tries in practice to choose g and h such that sequences $\{x^k\}$ and $\{y^k\}$ can be easily calculated, *i.e.,* either they are in an explicit form or their computations are inexpensive. Very often in practice, the solution of (D_k) to compute the sequence $\{y^k\}$ is explicit because the calculation of a subgradient of h is explicitly obtained by using the usual rules for calculating subdifferential of convex functions. But the solution of the convex program (P_k), if not explicit, should be achieved by efficient algorithms well-adapted to its special structure, in order to handle the large-scale setting.

1.3 Conclusion

In this chapter, we introduce a brief presentation of DC programming and DCA. The fundamentals of DC analysis, such as DC functions, DC duality, global and local optimality in DC optimization are also presented.

For a comprehensive study of DC programming and DCA, refer to Le Thi [1994, 1997], Le Thi and Pham Dinh [2002, 2005], Pham Dinh and Le Thi [1997, 1998, 2014] and the reference therein. The solution of a nonconvex problem by DC programming and DCA should have two tasks: looking for an appropriate DC decomposition and looking for a good starting point. In the next three chapters we will investigate DC programming and DCA to deal with compressed sensing, restoration image and segmentation image.

Chapter 2

Compressed sensing

2.1 Introduction

Compressed Sensing or Compressive Sensing (CS), introduced by Donoho (Donoho [2006b]) and Candès et al. (Candes et al. [2006a]), is a new signal processing technique. It provides a framework for efficiently acquiring a signal and after that recovering it from very few measurements when the interested signal is very sparse in some basis. The number of measurements needed to be stored is far lower than the Shannon-Nyquist rate while maintaining the essential information.

The CS has already became a key concept in various areas of applied mathematics, computer science, and electrical engineering and it is applied to various fields including radar imaging, signal extraction, aerial laser scanning, medical imaging, surface metrology, through wall radar imaging, space based imaging, ground penetrating radar imaging in archeology, geophysics, oil-exploration, landmine detection, forensics, civil engineering, etc,... (Compressive Sensing [home page]).

The theoretical foundation of compressed sensing has links with and also explores methodologies from various other fields such as applied harmonic analysis, frame theory, geometric functional analysis, numerical linear algebra, optimization theory and random matrix theory,...

In this section, we will briefly present the development history of CS, its mathematical models and also the most recent methods which have been used for finding sparse approximation of a signal.

2.1.1 Development history

The term compressed sensing was only introduced recently but it is gaining a great interest among researchers. There were certain roots and predecessors in some application areas such as image processing, geophysics, medical imaging, computer science,... as well as in mathematics, which laid the foundation for later developments.

It can be said that CS is a derivation and combination of *signal compression*, *sparse representation* and

1. A part of this chapter is published under the title:
Le Thi Hoai An, Nguyen Thi Bich Thuy, Le Hoai Minh, *Sparse signal recovery by Difference of Convex functions Algorithms*, Intelligent Information and Database Systems, Lecture Notes in Computer Science, Volume 7803, pp. 387–397, Springer–Verlag, 2013. The 5-*th* Asian Conference on Intelligent Information and Database Systems (ACIIDS 2013), Kuala Lumpur, Malaysia, 18–20 March, 2013.

sparse recovery.

Compressed sensing was derived from the process of **signal compression**. Let us consider a process of signal compression, which was described in Theodoridis et al. [2012], and see how the ideas of CS were initialed.

Given a signal, such as a speech, an image or an audio. The heart of any compression technique is to transform the signal into a suitably chosen domain in which the signal can be sparsely represented, i.e. only a relatively few signal's components in this domain are large and the rest are closed to zero. Then only the large coefficients are chosen to be coded and the rest are considered as zero.

For example, in JPEG-2000, an image, represented as a vector of the intensities (gray levels) of the image pixels, is transformed via the discrete wavelet transform (DWT). The result vector comprises only a few large components. This operation has the form:

$$S = \Phi^T s \tag{2.1}$$

where $s \in R^n$ is the vector of the "raw" signal samples , S is the transformed vector of s. Φ is a $n \times n$ transformation matrix, Φ^T is transpose matrix of Φ. Often, Φ is an orthonormal matrix, i.e. $\Phi^T\Phi = I$.

Basically, a transform is a projection of a vector on a new set of coordinate axes, which comprise the columns of the transformation matrix Φ. Some examples of transforms are wavelet, discrete Fourier (DFT) and discrete cosine (DCT) transforms,...

Because the transformation matrix is orthonormal, one can write that:

$$s = \Psi S \tag{2.2}$$

where $\Psi = \Phi$.

Equation (2.1) is known as the *analysis* and equation (2.2) as the *synthesis* equation (Theodoridis et al. [2012]).

Compression via such transforms exploits the fact that many signals in nature, which are rich in context, can be compactly represented in an appropriately chosen basis, depending on the modality of the signal. A standard compression task comprises the following stages:

1. Obtain l components of S, via the analysis step (2.1).
2. Retain k most significant components of them.
3. These k values are coded as well as their respective locations in the transform vector S.
4. The original signal s (or the approximation) can be obtained via the synthesis equation (2.2) (after storage or transmission), in case of recovering or restoration. Note that, in S only k most significant components are used, which are coded, while the rest are set to equal zero.

However, this process is extremely wasteful in practice: one processes (transforms) large signal vectors of l coordinates, where l can be quite large and then uses only a small percentage of the transformed coefficients and the rest are simply ignore. Moreover, one has to store/transmit the location of the respective large coefficients that were chosen to be coded.

This raises a fundamental question: Can one use $N(k < N < l)$ measurements to recover all the necessary information?

The work of Donoho (Donoho [2006b]) showed that this is indeed possible. Finding the answers for this question leads to the solution of an underdetermined system of linear equations, under the constraint that the unknown target vector is a sparse one. The importance of such techniques becomes even more apparent when one uses a overcomplete dictionaries instead of an orthonormal basis. That is the core idea of compressed sensing.

The importance of such techniques becomes even more apparent when instead of an orthonormal basis, as discussed before, by an expansion, in terms of what is known as overcomplete dictionaries. That is the core idea of compressed sensing.

Before the introduction of CS, there were two branches of methods which were developed parallelly and independently, namely: Sparse representation and Sparse recovery.

In **sparse representation**, there are some important results that formed the core ideas in this field.

The first one appeared in the work of Stephane Mallat and Zhifeng Zhang in 1993 (Mallat and Zhang [1993]), with the introduction of dictionaries concept, which replaces the traditional and critically sampled wavelet transform. With a given dictionary, the problem of finding the sparsest representation/approximation turns out to be significantly harder than in the case of sparsity with respect to a basis where the expansion coefficients are unique. The study of B. K. Natarajan in 1995 (Natarajan [1995]) showed that the general ℓ_0–norm, i.e. finding the sparsest solution of an underdetermined system, is *NP–hard*. Stephane Mallat and Zhifeng Zhang (Mallat and Zhang [1993]) presented some ideas that later became the center of this field such as a greedy pursuit technique. This technique approximates a sparse solution to an underdetermined linear system of equations, characterize the dictionaries by their coherence measure.

In 1998, Scott Shaobing Chen, David Donoho, and Michael Saunders (Chen et al. [1998]) introduced another pursuit technique that used the ℓ_1–norm for evaluating the sparsity. They showed that the task of sparsest solution could be tackled as a convex programming task that often leads to the proper solution.

In 2001, Donoho and Huo (Donoho and Huo [2001]) defined and partly answered what later became a key question in this field: Can one guarantee the success of a pursuit technique? Under what conditions? The analysis in Donoho and Huo [2001], provided necessary theoretical basics, which later grow with hundreds of interested researches, various workshops, conferences and an exponentially growing number of papers.

In parallel, many developments have been realized for the **sparse recovery** problem.

The first algorithm connected to sparse recovery problem was introduced by the French mathematician, Prony, in 1795 (Prony. [1975]). The author proposed a method, called *Prony method*, to estimate non–zero amplitudes and the corresponding frequencies of a sparse trigonometric polynomial from a small number of equispaced samples by solving an eigenvalue problem.

The use of ℓ_1 minimization already appeared in the Ph.D. thesis of B. Logan (Logan [1965]) in relevance of sparse frequency estimation, where the author observed that ℓ_1 minimization problem can recover exactly a frequency–sparse signal from undersampled data that provided a small enough sparsity.

Donoho and Logan [1992] can be considered as the first theoretical work on sparse recovery using ℓ_1 minimization. Nevertheless, geophysicists observed, in the late 1970's and 1980's, that ℓ_1 minimization can be successfully employed in reflection seismology. In this case, the sparse reflection function, that indicates changes between subsurface layers, is found (Taylor et al. [1979], Santosa and Symes [1986]).

In NMR (Nuclear Magnetic Resonance) spectroscopy domain, the idea links with recover sparse Fourier spectra from undersampled non equispaced samples, that was first introduced in the 1990s (Schmieder et al. [1993]). It has seen a significant development since then.

One of the direction of image processing is using total–variation minimization. This idea is closed to the ℓ_1 minimization and compressive sensing, firstly appeared in the paper of Rudin, Osher and Fatemi (Rudin et al. [1992]), and it was widely applied later on.

That are the major milestones which marked the development of CS domain. In the next part, we will present more detail on two problems: sparse representation and sparse recovery. The mathematical models of these problems will be considered.

2.1.2 Mathematical models

Sparsity is the signal structure behind many compression algorithms, which is used in transform coding. It is the most prevalent signal structure used in compressed sensing. To introduce the notion of sparsity, we rely on a signal representation in a given basis $\Psi = \{\psi_i, \psi_i \in \mathbb{R}^l\}_{i=1}^n$. A vector $x \in \mathbb{R}^l$ is called *k-sparse* in the basis or frame Ψ if there exists a vector $\theta \in \mathbb{R}^n$ with only $k \ll n$ non–zero entries such that $x = \Psi\theta$.

When $l < n$, for any vector $x \in \mathbb{R}^l$, there exist infinitely many decompositions $\theta \in \mathbb{R}^n$ such that $x = \Psi\theta$. In general setting, we refer to Ψ as a sparsifying dictionary or redundant dictionary D. Finding out the sparsest θ for a given vector x in the dictionary D leads to the problem of *Sparse signal representation*.

Hence the success of compressed sensing depends on the sparsity of the signal or image, that means, before sampling or compressing, the signal or image must be in the sparse representation. So it can be divided into two main stages: *Sparse signal representation* and *Sparse signal recovery*.

To state the problem, let $s = (s_1, s_2, ..., s_l), s \in \mathbb{R}^l$ be the studied signal.

We note that, if s itself is sparse, that is, it has very few non–zero coefficients, the cardinal

$$||s||_0 := \sharp\{i : s_i \neq 0\}$$

is small, then we can directly use s in the problem of compressed sensing.

Otherwise, if s is not really sparse, but there exists an orthonormal basis or a frame D such that $s = Dx$ with x being sparse, then we are going to find its sparse representation. It leads to the problem of sparse signal representation.

• Problem 1: Sparse representation problem

Given a signal $s \in \mathbb{R}^l$ and a dictionary $D = \{d_1, d_2, ..d_n\} \in \mathbb{R}^{l\times n}, n \gg l$. With the assumption that the dictionary is overcomplete, in general, there are many representations of $s = \sum_{i=1}^{n} x_i d_i$. The purpose of this phase is to represent s by using as few as possible atoms in D (each column vector d_i in D is called an *atom*). Set $x = (x_1, x_2, .., x_n)$, it leads to the problem:

$$\begin{cases} \min ||x||_0 \\ s.t: \ s = Dx. \end{cases}$$

If x (or s) is already sparse, it is said that x can be recovered from very few non–adaptive linear measurements. That is, given a matrix $\Phi \in \mathbb{R}^{m\times n}, m \ll n$, from m linear measurements $y = \Phi x, y \in \mathbb{R}^m$, we can recover x by solving the problem: sparse signal recovery.

• Problem 2: Sparse recovery problem

The sparse recovery problem is introduced as follows:

$$\begin{cases} \min ||x||_0 \\ s.t: \ y = \Phi x. \end{cases}$$

We see that, two above problems have the same form, however the matrix D and Φ have to satisfy the different conditions. D is a dictionary or a basis, which is generated based on a mathematical model such as wavelets, contourlets expansion,.. or learned from a training set. While Φ is the measurement matrix which is usually a random matrix.

In what follows, we will consider both problems, sparse representation and sparse recovery, by the same

optimisation problem below:

$$(P0): \begin{cases} \min ||x||_0 \\ s.t.\ b = Ax. \end{cases} \tag{2.3}$$

where $x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$, $m \ll n$.

Moreover, we will study the noise–aware version of the problem. Because of the introduce of the noise variable, the original signal cannot be exactly recovered, so we must find the sparsest representation that is as close as possible to the original signal. This difficult task can be solved by using relaxation technique, that replaces the exact constraint ($Ax = b$) by a quadratic penalty function ($||Ax - b||^2 < \epsilon$). This leads to two models: the least–square constraint model and regularization least square model.

The following part has been fully presented in (Donoho and Elad [2003], Rinaldi [2009]). For the reader's convenience we will give below a brief description.

The uniqueness of the optimal solution of the sparse representation problem and sparse recovery problem

There exist conditions to guarantee the uniqueness of the optimal solution for problem (2.3). The important properties of the sensing matrix A which were most widely known are *Restricted Isometry Property* (RIP)(Candes and Tao [2005]), *Spark* (Donoho and Elad [2003] and *Coherence* or *Mutual Coherence* (Mallat and Zhang [1993], Donoho and Huo [2001], Donoho and Elad [2003]).

Definition 2.1 *(Restricted Isometry Property). A matrix A satisfies the restricted isometry property of order k with the restricted isometry constant δ_k if δ_k is the smallest constant such that*

$$(1 - \delta_k)||x||_2^2 < ||Ax||_2^2 < (1 + \delta_k)||x||_2^2 \tag{2.4}$$

holds for all k-sparse signal x.

RIP is the condition to ensure that the k-sparse solution of the sparse signal recovery problem is unique.

Definition 2.2 *The spark of a matrix A is the smallest number of columns that form a linearly dependent set.*

By using the *spark*, we can give the first simple criterion for ensuring that the sparsest representation of a given input signal is unique:

Theorem 2.1 *(Donoho and Elad [2003]). Let us consider an input signal $b \in \mathbb{R}^m$ and a dictionary $A \in \mathbb{R}^{m \times n}$. If there exists a solution $\widetilde{x}$ of problem (2.3) such that:*

$$||\widetilde{x}||_0 < spark(A)/2, \tag{2.5}$$

then $\widetilde{x}$ is the unique sparsest representation of b.

The proof of this theorem is described in Donoho and Elad [2003].

However, calculating $spark(A)$ is a very tough task as a combinatorial search over all possible subsets of columns from A is required. Thus we need a simpler criterion to ensure the uniqueness condition. The concept of *mutual coherence* introduced in (Mallat and Zhang [1993], Donoho and Huo [2001], Donoho and Elad [2003]) can be used to obtain a new condition:

Definition 2.3 *The mutual coherence of a dictionary A, denoted by $\mu(A)$, is defined as the maximal absolute scalar product between two different atoms of A.*

$$\mu(A) = \max_{1 \leq j,k \leq n, j \neq k} |a_j^T a_k|. \tag{2.6}$$

The *mutual coherence* of a dictionary measures the similarity between the dictionary atoms. For an orthogonal matrix A, $\mu(A) = 0$. For an overcomplete matrix ($m < n$) we necessarily have $\mu(A) > 0$. If $\mu(A) = 1$, it implies the existence of two parallel atoms, and this causes confusion in the construction of sparse atom compositions.

Lemma 2.1 *(Donoho and Elad [2003]). Given a dictionary $A \in \mathbb{R}^{m \times n}$, the following relationship holds:*

$$spark(A) \geq 1 + \mu(A)^{-1}.$$

By using *mutual coherence* we attain the following theorem:

Theorem 2.2 *(Donoho and Elad [2003]). Let us consider an input signal $b \in \mathbb{R}^m$ and a dictionary $A \in \mathbb{R}^{m \times n}$. If there exists a solution $\widetilde{x}$ of problem (2.3) such that:*

$$||\widetilde{x}||_0 < (1 + \mu(A)^{-1})/2, \tag{2.7}$$

then $\widetilde{x}$ is the unique sparsest representation of b.

Note that the Theorem 2.2 is less powerful than Theorem 2.1 as it uses the *mutual coherence*, which represents a lower bound of *spark*.

Error–Constrained Approximation

In most practical situations we can not assume that the target signal can be exactly reconstructed by using the collection of some vectors in the basis. Then a noise–aware variant of the problem described in the previous section must be considered. The goal is to find a sparsest representation as close as possible to the original signal. The exact constraint $Ax = b$ is often relaxed by a quadratic penalty function $||Ax - b||_2^2$. Such relaxation allows us:

1. To define a quasi–solution in case no exact solution exists (even in cases where A has more rows than columns);
2. To exploit ideas from optimization theory;
3. To measure the quality of a candidate solution.

The problem (2.3) can be presented in error–tolerant version with an error tolerance $\epsilon > 0$:

$$(P1) : \begin{cases} \min ||x||_0. \\ s.t : ||Ax - b||_2^2 \leq \epsilon \end{cases} \tag{2.8}$$

or by choosing a proper parameter λ, the above optimization problem can be changed to:

$$(P2) : \min ||Ax - b||_2^2 + \lambda ||x||_0. \tag{2.9}$$

2.1.3 Existing approaches

ℓ_0–norm minimization problem is known as a NP–hard problem (Mallat and Zhang [1993]). It needs for a combinatorial search for its minimization and it is too sensible with noise. To deal with the fact that ℓ_0–norm is a discontinuous function, in several works, one approximates the ℓ_0–norm by a continuous, convex or nonconvex function. In the literature, we can divide these approaches into three categories: greedy algorithm approach, convex approach and non–convex approach.

2.1.3.1 Greedy approaches

The first approach is *Greedy algorithms* which addresses the sparsity issue directly such as Matching Pursuit (MP) (Mallat and Zhang [1993]), Orthogonal Matching Pursuit (OMP) (Pati et al. [1993]), Stagewise OMP (StOMP)(Donoho et al. [2006]), Subspace Pursuit (SP)(Dai and Milenkovic [May 2009]) or Compressive Sampling Matching Pursuit (CoSaMP) (Needell and Tropp [2009]). Their key idea is to find the "best matching" projections of a signal onto an overcomplete dictionary by searching, at each iteration, the best atom from the dictionary to maximize its inner product with the signal, and then subtract the contribution due to that atom. The process is repeated until the signal is satisfactorily decomposed.

An intrinsic feature of the algorithm is that when stopped after a few steps, it yields an approximation using only a few atoms. When the dictionary is orthogonal, the method works perfectly. If the object is composed of only $m \ll n$ atoms and the algorithm is run for m steps, it exactly recovers the underlying sparse structure. When the dictionary is not orthogonal, the situation is less clear. Because the algorithm is myopic, it might choose wrongly in the first few iterations and end up spending most of its time for correcting the mistakes made in the first few terms.

2.1.3.2 Convex approaches

The representative convex method is replacing the ℓ_0-norm by the ℓ_1–norm. The problem (2.3) can be replaced by the ℓ_1–optimization problem which is called basis pursuit (BP) (Chen et al. [1998]):

$$(P_{\ell_1}) \begin{cases} \min ||x||_1 \\ s.t : b = Ax, \end{cases} \tag{2.10}$$

or ℓ_1-regularized problem (also called LASSO (Tibshirani [1996])):

$$(P_{\ell_1} - regularized) \min \frac{1}{2}||Ax - b||_2^2 + \lambda||x||_1, \tag{2.11}$$

where $||x||_1 = \sum_{i=1}^{n} |x_i|$ is the ℓ_1–norm of the vector x. It has been proved that under certain conditions, the solutions of the ℓ_1-norm minimization problem and ℓ_0-norm minimization problem are equivalent (Donoho [2006a]). In Compressive Sensing, the ℓ_1–norm minimizer can be solved by using many efficient and stable algorithms such as the algorithms in $\ell_1 - magic$ of Candes et al. (Dantzig [1963], Candes et al. [2006a,b], Candes and Tao [2006, 2007]), the split Bregman (W. Yin et al. [2008], Goldstein and Osher [2009], alternating direction method of multipliers (ADMM) (Boyd et al. [2011]), proximal gradient methods (Combettes and Pesquet [2011]) and so on.

In Liang et al. [2014], the authors approximated ℓ_0-norm with a class of partly smooth convex functions and then proposed Forward Backward (FB) splitting algorithm (Bauschke and Combettes [2011]) for solving the non-smooth, convex optimization problems.

2.1.3.3 Non–convex approaches

The ℓ_1 minimization problem (2.10) is a convex optimization problem and thus tractable. However, to recovery exactly the signal, it requires some conditions. It has been proved in Chartrand [2007] that nonconvex minimizations are able to recover sparsity in a more efficient way and require fewer measurements than BP.

Nonconvex continuous approaches were extensively developed in which the ℓ_0 term is approximated by a continuous, nonconvex function. Some approximations were proposed to approximate the ℓ_0-norm.

The first one is concave exponential (EXP) approximation developed by Bradley and Mangasarianin in 1998 (Bradley and Mangasarian [1998]) and later in the works of Hosein Mohimani et al. (Mohimani et al. [2007, 2009]), applied in CS in the works of Rinaldi et al. (Rinaldi et al. [2010], Rinaldi [2011]). Logarithmic approximation and ℓ_p-norm with $p < 1$ were studied by Rao and Kreutz-Delgado (Rao and Kreutz-Delgado [1999]) and Fu (Fu [1998]). In Chartrand [2007] and Chartrand and Yin [2008], Chartrand and Yin considered the case of $0 \leq p \leq 1$ and applied to compressed sensing. Other very often used approximations are Smoothly Clipped Absolute Deviation (SCAD) function (Fan and Li [2001]), the capped–ℓ_1 (CaP or PiL1) function (Peleg and Meir [2008]) and the piecewise linear (PiL2) approximation (Le Thi [2012b]). In Zhao and Li [2012], Zhao and Li introduced a function which is a combination of ℓ_p-norm ($0 < p < 1$) and the *log* function to approximate ℓ_0.

$$F_\epsilon(x) = \sum_{i=1}^{n} log(|x_i| + \epsilon) + \sum_{i=1}^{n} (|x_i| + \epsilon)^p, 0 < p < 1.$$

In Esser et al. [2013], Esser et al. proposed the minimization of a non-convex function $\ell_1 - \ell_2$ and after in Yin et al. [2014] Yin et al. developed an algorithm based on DCA for solving this problem.

We can give an overview of the nonconvex approximations of ℓ_0-norm in the Table 2.1:

Table 2.1: Some nonconvex approximations of ℓ_0-norm

Approximation	**Formula**	
Exp (Bradley and Mangasarian [1998])	$\sum_{i=1}^{n}(1 - e^{-\alpha\|x_i\|})$	
SCAD (Fan and Li [2001])	$\sum_{i=1}^{n} \phi(x_i)$ where $\phi(t) = \begin{cases} \alpha\|t\| \text{ if } 0 \leq \|t\| \leq \alpha, \\ -\frac{\|t\|^2 - 2\alpha\beta\|t\| + \alpha^2}{2(\beta-1)} \text{ if } \alpha \leq \|t\| \leq \alpha\beta, \\ \frac{(\beta+1)\alpha^2}{2} \text{ if } \|t\| \geq \alpha\beta, \end{cases}$	
ℓ_p (Rao and Kreutz-Delgado [1999], Fu [1998])	$(\sum_{i=1}^{n} \|x_i\|^p)^{\frac{1}{p}}$	
log (Weston et al. [2003])	$\sum_{i=1}^{n}(log(\|x_i\| + \epsilon))$	
PiL1 or capped-ℓ_1 (Peleg and Meir [2008])	$\sum_{i=1}^{n} \min\{1, \alpha\|x_i\|\}$	
PiL2 (Le Thi [2012b])	$\sum_{i=1}^{n} \frac{1}{a-b}(\|x_i\| - b)$	
$\ell_p + log$ (Zhao and Li [2012])	$\sum_{i=1}^{n} log(\|x_i\| + \epsilon) + \sum_{i=1}^{n}(\|x_i\| + \epsilon)^p$	$0 < p < 1$
ℓ_{1-2} (Esser et al. [2013])	$\sum_{i=1}^{n} \|\|x_i\|\|_1 - \sum_{i=1}^{n} \|\|x_i\|\|_2$	

Dealing with these approximations, in the context of compressed sensing, several algorithms have been developed such as iteratively reweighted ℓ_1 (IRL1) (Candes and Randall [2008], Foucart and Lai [2009], Zhao and Li [2012]), iteratively reweighted least-squares (IRLS) (Gorodnitsky and Rao [1997], Rao and Kreutz-Delgado [1999], Chartrand [2007], Chartrand and Yin [2008], Daubechies et al. [2010], Lai et al. [2013]), Successive Linear Approximation (SLA) (Bradley and Mangasarian [1998], Rinaldi et al. [2010], Rinaldi [2011]), Local Linear Approximation (LLA)(Zou and Li [2008]), Two-stage ℓ_1 (Zhang [2009]), Adaptive Lasso (Zou [2006]), Local Quadratic Approximation (LQA) algorithm (Zou and Li [2008], Fan and Li [2001]), Difference of Convex functions Algorithm (DCA) (Thiao et al. [2008], Gasso et al. [2009], Le Thi et al. [2013c]), Yin et al. [2014]), proximal alternating linearized minimization algorithm (PALM)(Attouch et al. [2010]),... Recently, Le Thi et al. (Le Thi et al. [2014d]) gave a rigorous study on DC (Difference of Convex functions) approximation approaches for sparse optimization on both theoretical and algorithmic aspects. In their work, a unifying DC approximation, including all standard approximations, of the ℓ_0–norm is proposed. Furthermore, DCA schemes developed in that paper cover all standard nonconvex algorithms for dealing with ℓ_0-norm.

2.2 Our approaches

Motivated by the success of DCA in the previous works we propose to develop it for sparse signal recovering and sparse representation problem. We consider the problems $(P0)$, $(P1)$, $(P2)$ in which the ℓ_0 term is replaced by the piecewise concave approximation (Le Thi et al. [2008b]), the SCAD approximation (Le Thi et al. [2008e]), piecewise linear approximation (Cheng Soon and Le Thi [2013], Le Thi [2012b]) that appeared as the best approximations in Le Thi et al. [2014d]. The resulting problems are reformulated as DC programs and then solved by DCA.

2.2.1 The considered models

In this section, we study three models for both problems: sparse representation problem and sparse recovery problem. Three models are respectively named as: Linear constraint model, Least–Square constraint model and Regularization least square model.

2.2.1.1 Linear constraint model (LC)

The first model is given by:

$$(LC)\begin{cases}\min||x||_0\\ s.t.\ Ax=b.\end{cases} \tag{2.12}$$

2.2.1.2 Least-Square constraint Model (LSC)

The second model is presented as follows:

$$(LSC)\begin{cases}\min||x||_0\\ s.t.\ ||Ax-b||^2\leq\varepsilon.\end{cases} \tag{2.13}$$

2.2.1.3 Regularization least square model (RLS)

The last considered model is:

$$(RLS)\ \min\frac{1}{2}\|Ax-b\|^2+\lambda||x||_0,\ \lambda>0. \tag{2.14}$$

2.2.2 Approximations of ℓ_0–norm

The ℓ_0-norm results in a combinatorial optimization problem, and hence is not practical for large scale problems. We will replace ℓ_0-norm by an approximation function such that the approximate problem of (2.12) can be expressed as a DC program to which DCA is applicable.

Define the step function $step:\mathbb{R}\mapsto\mathbb{R}$ by $step(t)=1$ for $t\neq 0$ and $step(t)=0$ for $t=0$. Then for $x\in\mathbb{R}^n$ we have $\|x\|_0=\sum_{i=1}^n step(x_i)$.

The main idea is approximating the step function by an approximate function which is as close as possible to the step function and continuous at zero.

This idea is described in Figure 2.1:

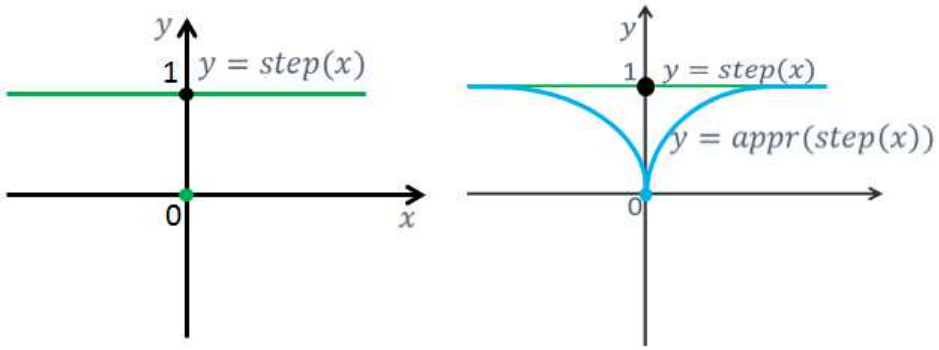


Figure 2.1: Approximation of step function by a continuous function.

2.2.2.1 Approximation 1: Exponential function (Exp)

The first non convex approximation was presented by Bradley and Mangasarian in Bradley and Mangasarian [1998] where the step function is approximated by:

$$\pi(t) \simeq 1 - e^{-\alpha t}, \quad \alpha > 0$$

and then the zero–norm $||x||_0$ is approximated by $||x||_0 \simeq \sum_{i=1}^{n}(1 - e^{-\alpha x_i})$.

In Le Thi et al. [2008b], the authors proposed a DC formulation of function π. Motivated by the efficiency of this DC decomposition, we use it for our problem.

For $t \in \mathbb{R}$, let η be the function defined by:

$$\eta(t, \alpha) = 1 - e^{-|\alpha t|}, \tag{2.15}$$

where $\alpha > 0$ and e denotes the base of the natural logarithm.

The step function $step(x_i)$ can be approximated by:

$$step(x_i) \simeq \eta(x_i, \alpha).$$

Then we have an approximation of the zero–norm $||x||_0$:

$$||x||_0 \simeq \sum_{i=1}^{n} \eta(x_i, \alpha). \tag{2.16}$$

This approximation is presented in Figure 2.2:

It is easy to see that $\eta(t)$ is a DC function of the form:

$$\eta(t) = g_1(t) - h_1(t) \tag{2.17}$$

where:

$$g_1(t) = |\alpha t|; \quad h_1(t) = |\alpha t| - 1 + e^{-|\alpha t|}. \tag{2.18}$$

Moreover, $g_1(t)$ is a polyhedral function which gives some interesting properties for convergence of DCA.

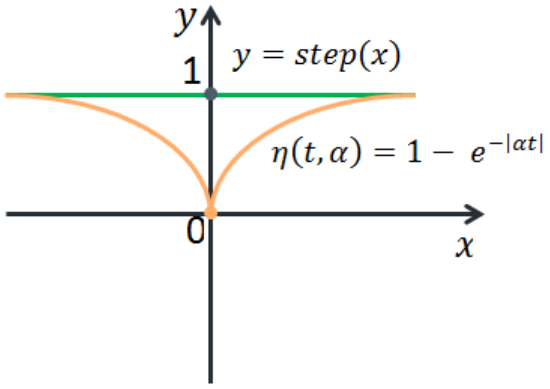


Figure 2.2: Exponential (EXP) approximation of l_0–norm.

2.2.2.2 Approximation 2: Smoothly Clipped Absolute Deviation (SCAD)

In this section, we use the approximation introduced in (Fan and Li [2001], Le Thi et al. [2008e]). J. Fan and R. Li (Fan and Li [2001]) introduced SCAD in the context of feature selection in regression. In Le Thi et al. [2008e], Le Thi et al. reformulated the SCAD function as a DC function and then developed an efficient algorithm based on DC Programming and DCA for solving it.

For $\beta > 2$ and $\alpha > 0$, SCAD approximation of ℓ_0–norm is given in Le Thi et al. [2008e] as follows: $\|x\|_0 \simeq \sum_{i=1}^{n} \phi(x_i)$, where the function $\phi(t)$ is defined by:

$$\phi(t) = \begin{cases} \alpha t \text{ if } 0 \leq t \leq \alpha, \\ -\frac{t^2 - 2\alpha\beta t + \alpha^2}{2(\beta - 1)} \text{ if } \alpha \leq t \leq \alpha\beta, \\ \frac{(\beta+1)\alpha^2}{2} \text{ if } t \geq \alpha\beta, \\ \phi(-t) \text{ if } t < 0. \end{cases} \tag{2.19}$$

The Figure 2.3 displays the SCAD approximation:

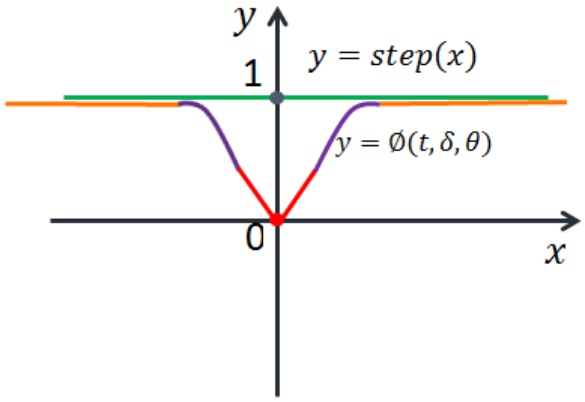


Figure 2.3: Smoothly Clipped Absolute Deviation (SCAD) approximation of l_0–norm.

Clearly, $\phi(t)$ can be expressed as a DC function of the form:

$$\phi(t) = g_1(t) - h_2(t), \tag{2.20}$$

where the function $h_2(t)$ is given by:

$$h_2(t) = \begin{cases} 0 \text{ if } 0 \leq t \leq \alpha, \\ -\frac{(t-\alpha)^2}{2(\beta-1)} \text{ if } \alpha \leq t \leq \alpha\beta, \\ \alpha t - \frac{(\beta+1)\alpha^2}{2} \text{ if } t \geq \alpha\beta, \\ h_2(-t) \text{ if } t \leq 0, \end{cases} \tag{2.21}$$

which is clearly convex, and $g_1(t)$ is already defined in (2.18).

2.2.2.3 Approximation 3: Piecewise Linear 1 (PiL1)

This approximation was introduced by Peleg and Meir in 2008 (Peleg and Meir [2008]), under the name capped-ℓ_1 regularizer. Then, in 2013, Cheng Soon and Le Thi proposed a DC formulation for this approximation (Cheng Soon and Le Thi [2013]).

For parameter $\alpha > 0$, let ϑ be the function defined by:

$$\vartheta(t, \alpha) = \min\{1, \alpha|t|\} = \begin{cases} 1 \text{ if } |t| \geq \frac{1}{\alpha} \\ \alpha t \text{ if } |t| \leq \frac{1}{\alpha} \end{cases} \quad t \in \mathbb{R}. \tag{2.22}$$

Figure (2.4) is the graph of the function $\vartheta(t, \alpha)$.

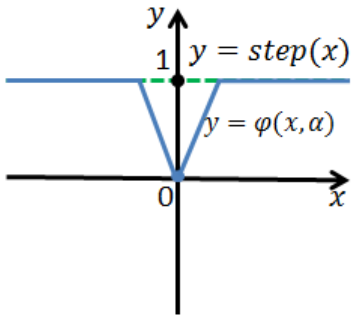


Figure 2.4: Piecewise Linear 1 (PiL1) approximation of l_0–norm.

It is easy to see that $0 \leq \vartheta(t, \alpha) \leq step(t)$ and $\vartheta(t, \alpha) \to step(t)$ as $\alpha \to \infty$, for any $x \in \mathbb{R}$. An approximation for ℓ_0–norm is given as follows (as in Cheng Soon and Le Thi [2013]):

$$||x||_0 \simeq \sum_{i=1}^{n} \vartheta(x_i, \alpha).$$

In what follows, for a given α, we will use $\vartheta(t)$ instead of $\vartheta(t, \alpha)$. Here, $\vartheta(t)$ can be decomposed as a DC function of the form:

$$\vartheta(t) = g_3(t) - h_3(t), \tag{2.23}$$

where the function $g_3(t)$ is defined as:

$$g_3(t) = 1 + \alpha|t| \tag{2.24}$$

and $h_3(t)$ is given by:

$$h_3(t) = max\{1, \alpha|t|\}, \tag{2.25}$$

which are clearly convex functions.

2.2.2.4 Approximation 4: Piecewise Linear 2 (PiL2)

In this part, we will consider another piecewise linear function of l_0–norm. This function was introduced in Le Thi [2012b].

For the parameters $u > 0, v > 0, u \geq v$, for $t \in \mathbb{R}$, we consider the function:

$$\zeta(t, u, v) = \frac{1}{u - v}(|t| - v). \tag{2.26}$$

Let $\rho(t, u, v)$ be defined by:

$$\rho(t, u, v) = \begin{cases} 1 \text{ if } |t| \geq u \\ \zeta(t, u, v) \text{ if } v \leq |t| \leq u, \quad t \in \mathbb{R} \\ 0 \quad \text{otherwise.} \end{cases} \tag{2.27}$$

The graph of this approximation is given in Figure (2.5).

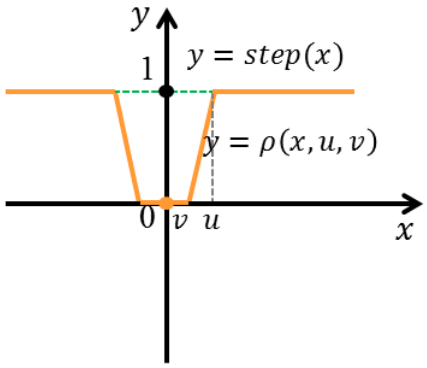


Figure 2.5: Piecewise Linear 2 (PiL2) approximation of l_0–norm.

Hence $||x||_0$ can be approximated as follows:

$$||x||_0 \simeq \sum_{i=1}^{n} \rho(x_i, u, v).$$

In what follows, for a given $u > 0, v > 0, u \geq v$, we use $\rho(t)$ instead of $\rho(t, u, v)$.

We have:

$$\rho(t) = min\{1, \zeta^+(t)\} = (1 + \zeta^+(t)) - max\{1, \zeta^+(t)\} \tag{2.28}$$

where:

$$\zeta^+(t) = \max\{0, \zeta(t)\}. \tag{2.29}$$

Set:

$$g_4(t) = (1 + \zeta^+(t)). \tag{2.30}$$

and:

$$h_4(t) = \max\{1, \zeta^+(t)\}. \tag{2.31}$$

We observe that $g_4(t)$ and $h_4(t)$ are convex functions, so $\rho(t)$ can be presented as a DC function:

$$\rho(t) = g_4(t) - h_4(t). \tag{2.32}$$

2.3 DC programming and DCA for solving the ℓ_0–norm problem

In this section, we propose the DC algorithms corresponding to three models (LC, LSC, RLS) and four above approximations.

Let $\varphi : \mathbb{R} \mapsto \mathbb{R}$ be a continuous function that approximates *step*, i.e. $\varphi(t) \approx step(t)$ for all $t \in \mathbb{R}$. Then for $x \in \mathbb{R}^n$ we have:

$$\|x\|_0 \approx \sum_{i=1}^{n} \varphi(x_i).$$

Using this approximation, we can formulate the approximate problem of the compressed sensing problem (2.12), (2.13), (2.14) in the form:

$$\min \left\{ F(x) := \Delta(x) + \sum_{i=1}^{n} \varphi(x_i) : x \in \Omega \right\}, \tag{2.33}$$

where

$$\Omega = \left\{ \begin{array}{l} \Omega_1 = \{x \in \mathbb{R}^n : Ax = b\} \text{ for LC-model(2.12)}, \\ \Omega_2 = \{x \in \mathbb{R}^n : ||Ax - b||^2 \leq \varepsilon\} \text{ for LSC-model (2.13)}, \\ \Omega_3 = \{x \in \mathbb{R}^n\} \text{ for RLS-model (2.14)}, \end{array} \right\} \tag{2.34}$$

and

$$\Delta(x) = \left\{ \begin{array}{l} 0 \text{ for LC-model and LSC-model (2.12), (2.13)} \\ \frac{1}{2}||Ax - b||^2 \text{ for RLS-model (2.14)}, \end{array} \right\}. \tag{2.35}$$

It is easy to see that $\Delta(x)$ is convex function on $\mathbb{R}^n$.

Suppose that φ can be expressed as a DC function of the form

$$\varphi(t) = g(t) - h(t), \quad \forall t \in \mathbb{R}, \tag{2.36}$$

where g and h are convex functions on $\mathbb{R}$. Then the problem (2.33) can be expressed as a DC program as follows

$$\min\{G(x) - H(x) : x \in \mathbb{R}^n\}, \tag{2.37}$$

where

$$G(x) = \chi_\Omega(x) + \Delta(x) + \sum_{i=1}^{n} g(x_i), \quad H(x) = \sum_{i=1}^{n} h(x_i)$$

are clearly convex functions on $\mathbb{R}^n$.

DCA applied to (2.37) consists of computing, at each iteration l,

- $y^l \in \partial H(x^l)$ that is equivalent to $y_i^l \in \partial h(x_i^l)$ for all $i = 1, \ldots, n$.

- Compute x^{l+1} by

$$x^{l+1} \quad \in \quad \text{argmin}\{G(x) - \langle y^l, x\rangle : x \in \mathbb{R}^n\}. \tag{2.38}$$

We will present the algorithm detail for each model in the sub-sections below.

2.3.1 Linear Constraint (LC) model

In the first model, LC-model, since Ω is a polyhedral convex set, if either g or h is polyhedral then (2.37) is a polyhedral DC program. In such case, DCA applied to (2.37) benefits from convergence properties of polyhedral DC programs. Using four approximations presented in the previous section, we have the DCA to solve the approximation problems is described as follows.

LC-DCA
Input: matrix $A \in \mathbb{R}^{m\times n}$ $(m << n)$, $b \in \mathbb{R}^m$
Output: the sparsest vector $x \in \mathbb{R}^n$ such that $Ax = b$
Initialization: Let $x^0 \in \mathbb{R}^n$ be a best guess, $\epsilon > 0$ be a small tolerance, $l = 0$
Repeat:

1. Compute $y_i^l \in \partial h(x_i^l),\ \forall i = 1, \ldots, n.$

2. Compute x^{l+1} as a solution to the convex program

$$\min\left\{\sum_{i=1}^{n} g(x_i) - \langle y^l, x\rangle : Ax = b\right\}. \tag{P_k}$$

3. $l = l + 1$
Until $\|x^l - x^{l-1}\|/(1 + \|x^l\|) < \epsilon$ or $|F(x^l) - F(x^{l-1})|/(1 + F(x^l)) < \epsilon$.

The implementation of algorithm LC-DCA according to each specific approximation function differs from each other in the computation of $y_i^l \in \partial h(x_i^l)$ in the step 1 and the subproblem in the step 2. The computation of $y_i^l \in \partial h(x_i^l)$ is given in Table 2.2. The subproblem in case of approximations EXP, SCAD, PiL1 has the form:

$$\min\left\{\alpha\|x\|_1 - \langle y^l, x\rangle : Ax = b\right\}$$
$$\Leftrightarrow \min\left\{\alpha\sum_{i=1}^{n} t_i - \langle y^l, x\rangle : Ax = b,\ -t_i \le x_i \le t_i\ \forall i = 1, \ldots, n\right\},$$

which is a linear program.

And the subproblem in case of approximation PiL2 has the form:

$$\min\left\{\frac{1}{u-v}\sum_{i=1}^{n}\max(v, |x_i|) - \langle y^l, x\rangle : Ax = b\right\}$$
$$\Leftrightarrow \min\left\{\frac{1}{u-v}\sum_{i=1}^{n} t_i - \langle y^l, x\rangle : Ax = b,\ -t_i \le x_i \le t_i, v \le t_i\ \forall i = 1, \ldots, n\right\},$$

which is also a linear program.

Observe that for all considered approximation functions, we always have g is polyhedral convex, so (2.37) for LC-model is a polyhedral DC program. Thus, LC-DCA applied to these approximations has finite convergence (Pham Dinh and Le Thi [1997, 1998], Le Thi and Pham Dinh [2005]). Some other convergence properties are stated in the theorem 2.3 below.

2.3.2 Least Square Constraint (LSC) Model

In this section, we consider the second model, named as *Least-Square Constraint (LSC)*. Replace the ℓ_0-norm by one of four approximations above we have a DC program. The DC algorithm for the result problem is described as follow:

Table 2.2: DC decomposition $\varphi = g - h$ and calculation of ∂h. The notation $\text{sgn}(t)$ denotes sign of t.

Name	$g(t)$	$h(t)$	$\overline{t} \in \partial h(t)$
EXP	$\alpha\|t\|$	$\alpha\|t\| - 1 + e^{-\alpha\|t\|}$	$\text{sgn}(t)(\alpha\|t\| - e^{-\alpha\|t\|})$
SCAD	$\alpha\|t\|$	$\begin{cases} 0 & \text{if } \|t\| \le \alpha \\ \frac{(\|t\|-\alpha)^2}{2(\beta-1)} & \text{if } \alpha \le \|t\| \le \alpha\beta \\ \alpha\|t\| - \frac{(\beta+1)\alpha^2}{2} & \text{otherwise} \end{cases}$	$\begin{cases} 0 & \text{if } \|t\| \le \alpha \\ \text{sgn}(t)\frac{\|t\|-\alpha}{\beta-1} & \text{if } \alpha < \|t\| < \alpha\beta \\ \text{sgn}(t)\alpha & \text{otherwise} \end{cases}$
PiL1	$\alpha\|t\|$	$\max\{1, \alpha\|t\|\} - 1$	$\begin{cases} 0 & \text{if } \|t\| \le \frac{1}{\alpha} \\ \text{sgn}(t)\alpha & \text{otherwise} \end{cases}$
PiL2	$\frac{\max\{v, \|t\|\}}{u - v}$	$\frac{\max\{u, \|t\|\}}{u - v} - 1$	$\begin{cases} 0 & \text{if } \|t\| \le u \\ \frac{\text{sgn}(t)}{u-v} & \text{otherwise} \end{cases}$

LSC-DCA
Input: matrix $A \in \mathbb{R}^{m\times n}$ $(m << n)$, $b \in \mathbb{R}^m$, $\varepsilon > 0$
Output: the sparsest vector $x \in \mathbb{R}^n$ such that $||Ax - b||^2 \le \varepsilon$
Initialization: Let $x^0 \in \mathbb{R}^n$ be a best guess, $\epsilon > 0$ be a small tolerance, $l = 0$
Repeat:

1. Compute $y_i^l \in \partial h(x_i^l),\ \forall i = 1, \ldots, n.$

2. Compute x^{l+1} as a solution to the convex program

$$\min\left\{\sum_{i=1}^{n} g(x_i) - \langle y^l, x\rangle : ||Ax - b||^2 \le \varepsilon\right\}. \tag{P_k}$$

3. $l = l + 1$
Until $\|x^l - x^{l-1}\|/(1 + \|x^l\|) < \epsilon$ or $|F(x^l) - F(x^{l-1})|/(1 + F(x^l)) < \epsilon$.

The computation of $y_i^l \in \partial h(x_i^l)$ is given in Table 2.2. The subproblem in case of approximations EXP, SCAD, PiL1 has the form:

$$\min\left\{\alpha\|x\|_1 - \langle y^l, x\rangle : ||Ax - b||^2 \le \varepsilon\right\}$$

$$\Leftrightarrow \min\left\{\alpha\sum_{i=1}^{n} t_i - \langle y^l, x\rangle : ||Ax - b||^2 \le \varepsilon,\ -t_i \le x_i \le t_i\ \forall i = 1, \ldots, n\right\},$$

which is a least square program.

And the subproblem in case of approximation PiL2 has the form:

$$\min\left\{\frac{1}{u - v}\sum_{i=1}^{n} \max(v, |x_i|) - \langle y^l, x\rangle : ||Ax - b||^2 \le \varepsilon\right\}$$

$$\Leftrightarrow \min\left\{\frac{1}{u - v}\sum_{i=1}^{n} t_i - \langle y^l, x\rangle : ||Ax - b||^2 \le \varepsilon,\ -t_i \le x_i \le t_i, v \le t_i\ \forall i = 1, \ldots, n\right\},$$

which is also a least square program.

2.3.3 Regularization Least Square (RLS) model

In the last part, we consider the model RLS. Considering four approximations of ℓ_0-norm with this model, we have the DC algorithm for the result problem as follow:

RLS-DCA
Input: matrix $A \in \mathbb{R}^{m\times n}$ $(m << n)$, $b \in \mathbb{R}^m$
Output: the sparsest vector $x \in \mathbb{R}^n$ which minimizes the function $\frac{1}{2}||Ax-b||^2 + \lambda||x||_0$
Initialization: Let $x^0 \in \mathbb{R}^n$ be a best guess, $\epsilon > 0$ be a small tolerance, $l = 0$
Repeat:

1. Compute $y_i^l \in \partial h(x_i^l),\ \forall i = 1,\dots,n.$

2. Compute x^{l+1} as a solution to the convex program

$$\min\left\{\frac{1}{2}||Ax-b||^2 + \lambda\{\sum_{i=1}^{n} g(x_i) - \langle y^l, x\rangle\right\}. \qquad (P_k)$$

3. $l = l + 1$
Until $\|x^l - x^{l-1}\|/(1+\|x^l\|) < \epsilon$ or $|F(x^l) - F(x^{l-1})|/(1+F(x^l)) < \epsilon$.

The computation of $y_i^l \in \partial h(x_i^l)$ is given in Table 2.2 but note that in this case, y_i^l is multiplied with a factor λ.

The subproblem in case of approximations EXP, SCAD, PiL1 has the form:

$$\min\left\{\frac{1}{2}||Ax-b||^2 + \lambda\alpha\|x\|_1 - \langle y^l, x\rangle\right\}$$

$$\Leftrightarrow \min\left\{\frac{1}{2}||Ax-b||^2 + \lambda\alpha\sum_{i=1}^{n} t_i - \langle y^l, x\rangle :\ -t_i \le x_i \le t_i\ \forall i = 1,\dots,n\right\},$$

which is a least square program.

And the subproblem in case of approximation PiL2 has the form:

$$\min\left\{\frac{1}{2}||Ax-b||^2 + \frac{\lambda}{u-v}\sum_{i=1}^{n}\max(v,|x_i|) - \langle y^l, x\rangle\right\}$$

$$\Leftrightarrow \min\left\{\frac{1}{2}||Ax-b||^2 + \frac{\lambda}{u-v}\sum_{i=1}^{n} t_i - \langle y^l, x\rangle : ||Ax-b||^2 \le \varepsilon,\ -t_i \le x_i \le t_i, v \le t_i\ \forall i = 1,\dots,n\right\},$$

which is also a least square program.

The convergence properties of 12 algorithms LC–{EXP, SCAD, PiL1, PiL2}, LSC–{EXP, SCAD, PiL1, PiL2}, RLS–{EXP, SCAD, PiL1, PiL2} are stated in the following theorem.

2.4 Convergence properties

For simplifying the presentation, we use the notation(s) F (resp. G and H) to denote the objective function (resp. DC components of F) of all considered optimization problems.

Theorem 2.3 *(Convergence properties of 12 algorithms: LC–EXP, LC–SCAD, LC–PiL1, LC–PiL2, LSC–EXP, LSC–SCAD, LSC–PiL1, LSC–PiL2, RLS–EXP, RLS–SCAD, RLS–PiL1 and RLS–PiL2)*

Suppose that the problems of the form (2.33) *are solvable, i.e. their solutions exist. Then we have the following properties.*
i) DCA generates a sequence $\{x^k\}$ *such that the sequence* $\{F(x^k)\}$ *is monotonously decreasing.*
ii) If x^* *is a limit point of* $\{x^k\}$ *then* x^* *is a critical point.*
iii) In LC–EXP, LC–SCAD, LC–PiL1, LC–PiL2, LSC–PiL1, LSC–PiL2, RLS–PiL1, and RLS–PiL2: the sequence $\{x^k\}$ *converges to* x^* *after a finite number of iterations. The point* x^* *is almost always a local minimizer of the corresponding optimization problem* (2.33)*. Especially,*
- *In LC–PiL1 (resp. LSC–PiL1, RLS–PiL1), if*

$$x_i^* \notin \{-\frac{1}{\alpha}; \frac{1}{\alpha}\} \quad \forall i = 1..n \tag{2.39}$$

then x^* *is a local minimizer of* (2.33).
- *In LC–PiL2 (resp. LSC–PiL2, RLS–PiL2), if*

$$x_i^* \notin \{-u; u\} \quad \forall i = 1..n \tag{2.40}$$

then x^* *is a local minimizer of* (2.33).

Proof: (i) and (ii) are direct consequences of the convergence properties of DCA for general DC programs (Pham Dinh and Le Thi [1997], Le Thi and Pham Dinh [2005]).

Below, we are going to prove iii).

In LC–PiL1 (resp. LSC–PiL1, RLS–PiL1), the second DC component of (2.33), says H, is a polyhedral convex function. Moreover, if the condition (2.39) holds, H is differentiable at x^*. Then using the convergence property of DCA for polyhedral DC programs (Pham Dinh and Le Thi [1997],Le Thi and Pham Dinh [2005]), we deduce that x^* is local minimizer of (2.33). Moreover, since a polyhedral convex function is almost always differentiable, say, it is differentiable everywhere except on a set of measure zero, we can say that x^* is almost always a local minimizer of (2.33).

The proof is similar for LC–PiL2 (resp. LSC–PiL2, RLS–PiL2) since its second DC component H is also polyhedral convex and is differentiable if the condition (2.40) holds.

In LC–EXP (resp. LC–SCAD), the first DC component of (2.33), says G, is a polyhedral convex function, so is G^*. Hence, the dual DC program of (2.33) is a polyhedral DC program.

Thus, if y^* is the limit point of the sequence $\{y^k\}$ generated by LC–EXP (resp. LC–SCAD), we can say that y^* is almost always a local minimizer of the dual DC program of (2.33).

On the other hand, according to the property of transportation of local minimizers in DC programming, we have the following (Le Thi and Pham Dinh [2005]): let y^* be a local minimizer of the dual program of (2.33) and $x^* \in \partial G^*(y^*)$. If H is differentiable at x^* then x^* is a local minimizer of (2.33). Combining this property with the facts that y^* is almost always a local minimizer of the dual DC program of LC-EXP (resp. LC-SCAD) and H is differentiable everywhere, we conclude that x^* is almost always a local minimizer of (2.33). The proof is then complete. □

2.5 Numerical experiments

In this section, we perform our experiments on two problems: sparse representation and sparse recovery. For the sparse recovery problem, two testing scenarios will be considered. The first one tests on ground truth data. The data for this test is taken from Sparco Toolbox (Sparco [home page]). The second one

tests on generated data with different types of sensing matrix and different values of sparsity of x. This test evaluates the accuracy rate of each approximation and compares the efficiency of DCA-based algorithms with two other iterative algorithms (iteratively reweighted ℓ_1 - IRL1 and iteratively reweighted least squares - IRLS).

2.5.1 Comparative algorithms

To evaluate the efficiency of our approach, we make the comparisons with five algorithms: two convex algorithms: *ℓ1eq* (Candes and Tao [2006]), *ℓ1qc* (Candes et al. [2006b]) from $\ell_1 - magic$ package; and three nonconvex algorithms including *SL0* (Mohimani et al. [2007, 2009]), reweighted-ℓ_1 (RWL1)(Candes et al. [2008]) and FOCUSS (Gorodnitsky and Rao [1997]). The last three nonconvex algorithms are in fact special versions of DCA (Le Thi et al. [2014d]).

2.5.2 Setup and parameters

The parameters of each algorithm are chosen as follows: $\alpha \in [0.0001; 10]$ for approximation EXP; $\alpha \in [0.001; 5]$ and $\beta \in [4; 1000]$ for approximation SCAD; $\alpha \in [0.0001; 10]$ for approximation PiL1; $u \in [0.01; 100]$ and $v \in [0.00001; 250]$ for approximation PiL2. All algorithms stop with the tolerance $\tau = 10^{-4}$.

In the least–square constraint (LSC) model, $\varepsilon = 10^{-5}$. In the Regularization least square (RLS) model, we use a trade–off parameter λ. In our tests, the values of λ is taken on $[0.5; 5]$ with the step–size of 0.5.

The parameters for *RWL1, FOCUSS*, $\ell_1 eq$, $\ell_1 qc$ and *SL0* are chosen as described in the related papers which have been cited above.

For all the tests, we used a pseudo-inverse solution of $Ax = b$ as the initial points.

All our algorithms were implemented in Visual C++ 2008, and run on a PC Intel Core(TM)2 Quad CPU $Q9505$, 2.83 GHz and 4GB RAM. CPLEX 12.3 was used for solving linear and convex quadratic problems.

2.5.3 Sparse representation problem

Test protocol:
Given a signal s, assume that we know its sparsest representation x_0 on the dictionary D (or matrix A), so we have: $s = Ax_0$. For testing our algorithms, we solve the problem:

$$\begin{cases} \min ||x||_0 \\ s.t.\ s = Ax. \end{cases}$$

In this test, it is expected to get a solution as sparse as possible and closest to x_0. We tested on generated data with two models: the noiseless model $s = Ax_0$ and the noisy model $s = Ax_0 + \eta$, where η is an additive white Gaussian noise (a popular noise) with variance is set to 0.005 in simulation. We recall that: $s \in \mathbb{R}^m; x, x_0 \in \mathbb{R}^n; A \in \mathbb{R}^{m \times n}, m \leq n; k$ is sparsity of $x_0 : k = ||x_0||_0$. In our experiments $n = 512$, $m = 120$, $k = 20$.

The experiments are then repeated 100 times (with the same parameters, but for differently randomly generated sources and matrices). To evaluate the solution's quality, we used two criteria: the value of $||x||_0$ and the value of $||x - x_0||$. The first one presents the $k-sparse$ value. The second one describes the difference between the original signal and its estimation. The best quality is achieved when $||x||_0$ and $||x - x_0||$ are smallest.

Comments on the results:
The Table 2.3 and Table 2.4 show the results of all algorithms in two cases: noiseless $s = Ax_0$ and Gaussian noise $s = Ax_0 + \eta$.

From the numerical results, it is observed that: In the case of noiseless data, *RWL1*, *LC-Exp, LC-SCAD* and *LC-PiL1* gave the best results. Their solutions are exactly the same as x_0 for all the trials. *ℓ1eq* is following with the sparsity of its solutions always equal to 20 but the components of its solutions are different a bit in comparision with x_0. *LC-PiL2* gave the exact solutions in almost cases, but its solutions are sometimes not very sparse. *LC-PiL2* gave the exact solutions in 93 out of 100 trials. *FOCUSS* and *SL0* are the worst algorithms. *SL0* gave the solutions with a larger number of nonzero components than those of FOCUSS but most of them are close to 0, thus its solutions are closer to x_0 than the solutions of *FOCUSS*.

In the case of noisy data, all DCA-based algorithms gave very sparse solutions and *RLS-PiL1* is the best algorithm. It can present the signal s exactly as the sparse vector x_0. *ℓ1qc* is the worst algorithm. It solutions have many nonzero components and the differences $||x - x_0||$ are the biggest.

Comparing the four approximations in the same model, it is observed that *PiL1* is the best approximation. It usually gave the solutions sparser and closer to x_0 than the others.

Table 2.3: The average values $||x||_0$ and $||x - x_0||$ in the noiseless model $s = Ax_0$.

Algorithm	$\|x\|_0$	$\|x - x_0\|$
LC-Exp	**20**	**0.0**
LC-SCAD	**20**	**0.0**
LC-PiL1	**20**	**0.0**
LC-PiL2	22	0.002
FOCUSS	39	0.08
RWL1	**20**	**0.0**
ℓ1eq	20	0.000198
SL0	217	0.010784

Table 2.4: The values $||x||_0$ and $||x - x_0||$ in the noisy model $s = Ax_0 + e$

Algorithm	$\|x\|_0$	$\|x - x_0\|$
LSC-Exp	20	0.000026
LSC-SCAD	20	0.000545
LSC-PiL1	20	0.000002
LSC-PiL2	20	0.000047
RLS-Exp	20	0.000053
RLS-SCAD	20	0.000042
RLS-PiL1	**20**	**0.0**
RLS-PiL2	20	0.000024
ℓ1qc	34	0.012928

2.5.4 Sparse recovery problem

2.5.4.1 Test with ground truth data

In this experiment, we test the sparse recovery algorithms on 11 datasets ($Prb1 - Prb11$) from Sparco Toolbox (Sparco [home page]).

We used two criteria for evaluating: the sparse value of the recovered signal $||x||_0$ and the difference between the recovered signal and the original signal $||x - x_0||$. The best quality achieved when both of $||x||_0$ and $||x - x_0||$ are smallest. Our experiments are performed in two different cases: noiseless measurements $b = Ax$ and noisy measurements $b = Ax + \eta$ where η is an additive white Gaussian noise with variance 0.005.

For noiseless data, we performed the experiments on the algorithms based on the LC model (2.12) which are *LC-EXP, LC-SCAD, LC-PiL1, LC-PiL2* (DCA-based algorithms) and $\ell_1 eq$, *SL0, FOCUSS, RWL1.*

For the noisy data, the algorithms based on LSC model (2.13) and RLS model (2.14) ($\ell_1 eq$ and DCA-based algorithms LSC-EXP, LsC-SCAD, LSC-PiL1, LSC-PiL2, RLS-EXP, RLS-SCAD, RLS-PiL1, RLS-PiL2) have been used for testing.

The table 2.5 and table 2.6 show the results of all algorithms in two cases, noiseless and noisy measurements, respectively.

Table 2.5: The values of $||x||_0$, $d = ||x - x_0||$ on 11 noiseless datatest.

Prob		**LC-EXP**	**LC-SCAD**	**LC-PiL1**	**LC-PiL2**	FOCUSS	RWL1	ℓ1eq	SL0
1	$\|\|x\|\|_0$	**4**	**4**	**4**	**4**	**4**	**4**	5	4
$\|\|x_0\|\|_0 = 4$	d	**0**	**0**	**0**	0.000188	0.001183	**0**	0.004102	0.003516
2	$\|\|x\|\|_0$	**71**	**71**	**71**	**71**	**71**	**71**	**71**	71
$\|\|x_0\|\|_0 = 71$	d	**0**	**0**	**0**	**0**	**0**	**0**	**0**	0.000014
3	$\|\|x\|\|_0$	**121**	**121**	**121**	**121**	951	**121**	121	121
$\|\|x_0\|\|_0 = 121$	d	**0**	**0**	**0**	0.000482	7.250896	**0**	0.000124	0.027687
4	$\|\|x\|\|_0$	**119**	**119**	**119**	119	119	**119**	119	119
$\|\|x_0\|\|_0 = 119$	d	**0**	**0**	**0**	0.004099	0.000877	**0**	0.000002	0.018318
5	$\|\|x\|\|_0$	**63**	**63**	**63**	63	1234	**63**	63	190
$\|\|x_0\|\|_0 = 63$	d	**0**	**0**	**0**	0.001985	1.387025	**0**	0.000923	0.872966
6	$\|\|x\|\|_0$	190	190	190	190	1149	**191**	600	582
$\|\|x_0\|\|_0 = 191$	d	1.82	1.82	1.82	1.82	65.0431	0.0	3.461079	2.5436
7	$\|\|x\|\|_0$	**20**	**20**	**20**	20	20	**20**	20	20
$\|\|x0\|\|_0 = 20$	d	**0**	**0**	**0**	0.000672	0.000018	**0**	0.000001	0.016376
8	$\|\|x\|\|_0$	**20**	**20**	**20**	20	20	**20**	20	20
$\|\|x_0\|\|_0 = 20$	d	**0**	**0**	**0**	0.000446	0.00002	**0**	0.000001	0.012800
9	$\|\|x\|\|_0$	**12**	**12**	**12**	**12**	**12**	**12**	**12**	12
$\|\|x_0\|\|_0 = 12$	d	**0**	**0**	**0**	**0**	**0**	**0**	**0**	0.000237
10	$\|\|x\|\|_0$	**12**	**12**	**12**	**12**	**12**	**12**	**12**	**12**
$\|\|x_0\|\|_0 = 12$	d	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
11	$\|\|x\|\|_0$	**32**	**32**	**32**	32	32	**32**	32	32
$\|\|x_0\|\|_0 = 32$	d	**0**	**0**	**0**	0.008903	0.229081	**0**	0.000005	0.010723

Table 2.6: The values of $||x||_0$, $d = ||x - x_0||$ on 11 noisy datatest.

Prob		**LSC-EXP**	**LSC-SCAD**	**LSC-PiL1**	**LSC-PiL2**	**RLS-EXP**	**RLS-SCAD**	**RLS-PiL1**	**RLS-PiL2**	ℓ1qc
1	$\|\|x\|\|_0$	36	36	36	36	**4**	4	4	4	1040
$\|\|x_0\|\|_0 = 4$	d	0.097057	0.09789	0.100591	0.097894	**0.027499**	0.054221	0.052234	0.068706	1.990803
2	$\|\|x\|\|_0$	71	71	71	71	71	71	**71**	71	71
$\|\|x0\|\|_0$= 71	d	0.113566	0.113566	0.102963	0.102965	0.161987	0.104843	**0.101305**	0.102965	0.178764
3	$\|\|x\|\|_0$	138	136	138	135	119	114	119	128	257
$\|\|x0\|\|_0$= 121	d	0.077145	0.074499	0.077348	0.073009	0.264391	0.362375	0.28378	0.081561	0.179695
4	$\|\|x\|\|_0$	119	119	119	119	116	112	**119**	119	274
$\|\|x0\|\|_0$= 119	d	0.003162	0.003163	0.003164	0.003163	0.253054	0.22639	**0**	0.054544	0.170888
5	$\|\|x\|\|_0$	101	128	132	130	113	**63**	118	127	302
$\|\|x0\|\|_0$= 63	d	0.101573	0.14555	0.148264	0.216718	0.13318	**0.043091**	0.142223	0.279876	0.338292
6	$\|\|x\|\|_0$	187	586	184	207	191	**191**	191	191	598
$\|\|x0\|\|_0$= 191	d	0.048051	3.98564	1.821778	1.823755	0.006154	**0.004088**	0.013876	0.014726	3.426183
7	$\|\|x\|\|_0$	20	20	**20**	20	20	20	20	47	20
$\|\|x0\|\|_0$= 20	d	0.006773	0.006773	**0.000414**	0.032666	0.0493	0.049797	0.049798	0.247249	0.27709
8	$\|\|x\|\|_0$	20	20	20	20	20	**20**	20	20	20
$\|\|x0\|\|_0$= 20	d	0.006922	0.006922	0.001025	0.093901	0.052308	**0**	0.052836	0.190834	0.283188
9	$\|\|x\|\|_0$	12	12	12	12	12	12	**12**	**12**	54
$\|\|x0\|\|_0$= 12	d	0.003196	0.010421	0.001381	0.000246	0.060521	0.021206	**0**	**0**	0.070952
10	$\|\|x\|\|_0$	12	12	12	12	12	12	12	**12**	496
$\|\|x0\|\|_0$= 12	d	0.428025	0.42741	0.425982	0.538994	0.427912	0.427912	0.427912	**0.425304**	3.794085
11	$\|\|x\|\|_0$	32	32	32	32	32	**32**	32	32	37
$\|\|x0\|\|_0$= 32	d	0.003368	0.003843	0.00371	0.056866	0.003544	**0.00216**	0.003551	0.056944	0.009704

In the case of noiseless data, *RWL1* is the best since it recovered exactly the original signal on 11/11 problems, followed by (*LC-EXP*, *LC-SCAD*, *LC-PiL1*) with the ratio of 10/11. *FOCUSS*, *ℓ1eq* and *LC-PiL2* recovered exactly on 3 problems but *LC-PiL2* gave the solutions more exact and sparser than the others. *SL0* gave exactly recovered signal only on the *Prob* 11. However, *FOCUSS* is the least stable algorithm. In some trials, the recovered signals of *FOCUSS* were far different from the original signals, for example in *Prob* 3, 5, 6, 11.

In the case of noisy data, between two models *RLS* and *LSC*, the algorithms based on the *RLS* models are the best. Their solutions are the closest to the original signals and have the ℓ_0–norm smallest. In this group, *RLS-SCAD* is better than the others. It gave the closest solutions to original signals on 4/11 problems, followed by *RLS-PiL1* on 3/11 problems and the next is *RLS-PiL2* with 2/11 problems. In general, the solutions given by these algorithms are very sparse and close to the original signals.

In the group of algorithms based on the *RLS* models, the *LSC-DCA* group (*EXP-LSC, SCAD-LSC, PiL1-LSC, PiL2-LSC*) gave the better solutions than $\ell_1 qc$. In this group, the approximation *EXP* and *PiL1* are better than *PiL2* and *SCAD* (see the Table 2.6).

Among four approximations, it is observed that the approximation *PiL1* is the best. The algorithms using this approximation gave the solutions sparsest and closest to original signals and they are very stable. The results of the algorithms using (*Exp, SCAD*) are the same and better than the ones using *PiL2*. Note that both *SCAD* and *PiL2* depend on two parameters so it is more difficult to choose the suitable parameter values.

The figures of recovered signals (from figure 2.9 to 2.19) proved the superiority of DCA based algorithms in comparison with original signal and other algorithms.

2.5.4.2 Test with synthetic data

In this section, we studied how the sensing matrix affects on the results of signal recovery problem. Two important properties of A which are *Restricted Isometry Property* (RIP) and *coherence* (or *mutual coherence*) were considered.

We tested on three scenarios. In the first test, we considered random Gaussian matrices which are incoherent and have small RIP constants with high probability. In the second test, we also considered random Gaussian matrices but with the scale m/n smaller (fewer measurements). In the last test, the highly coherent matrices were considered. Note that the *coherence* $\mu(A)$ of a matrix A measures the similarity between the matrix columns. If $\mu(A) = 1$, it implies the existence of two parallel atoms, and this causes confusion in the reconstruction of sparse coefficients.

The test was done on LC-models and the results were compared with *RWL1* and *FOCUSS*.

The test protocol is as follows: we sampled a random $m \times n$ matrix A and generated a target signal $x \in \mathbb{R}^n$ with $||x||_0 = k$. The k nonzero spike positions were chosen randomly, and the nonzero values were chosen randomly from a zero-mean unit-variance Gaussian distribution. We then computed the measurement $b = Ax$ and applied each solver to produce a reconstruction x^* of x. The reconstruction was considered as success if the relative error satisfies $\frac{||x-x^*||}{||x||} \leq 10^{-2}$. We ran 100 independent trials at each sparsity level $k \in \{5, 10, \ldots, 35\}$ and recorded the corresponding success rates.

The parameter for *RWL1* was chosen as $\epsilon = 0.1$. For *FOCUSS*, we chose $\epsilon = 10^{-6}$. For our algorithms, the parameters were set to $\alpha = 0.05$ (for *EXP, SCAD*, and *PiL1*), $\beta = 40$ (for *SCAD*), $u = 0.001$, $v = 0.101$ (for *PiL2*).

In our first scenario, we tested on RIP matrix. Set $m = 64, n = 256$, a random $m \times n$ matrix A with i.i.d. Gaussian entries is sampled. The coherence of A was approximately 0.55. The sparsity level k was taken in $\{10, 15, \ldots, 35\}$.

The experiment results were given in Figure 2.6

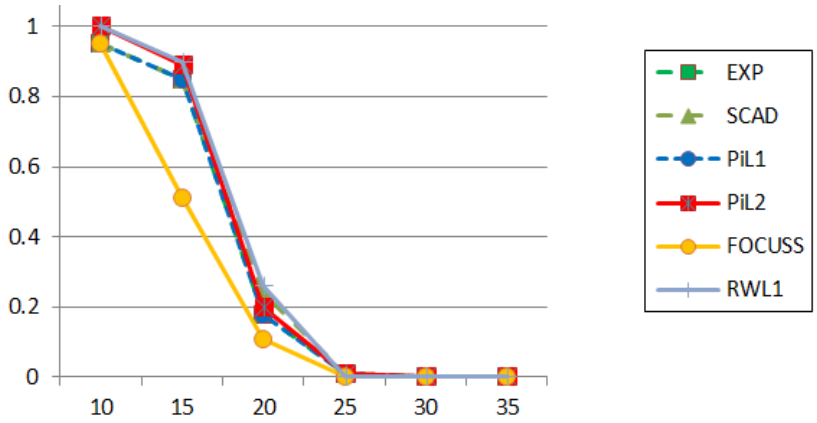


Figure 2.6: Success rates using incoherent sensing matrix, $m = 64, n = 256$

In the second scenario, we focused on the random Gaussian matrix A but with m very smaller compared to n. The size of A was $m \times n = 100 \times 2000$. The coherence of A was also approximately 0.55. The sparsity level k was taken in $\{5, 10, 15, \ldots, 35\}$. We also ran 100 independent trials at each sparsity level and recorded the corresponding success rates. The results are shown in figure 2.7.

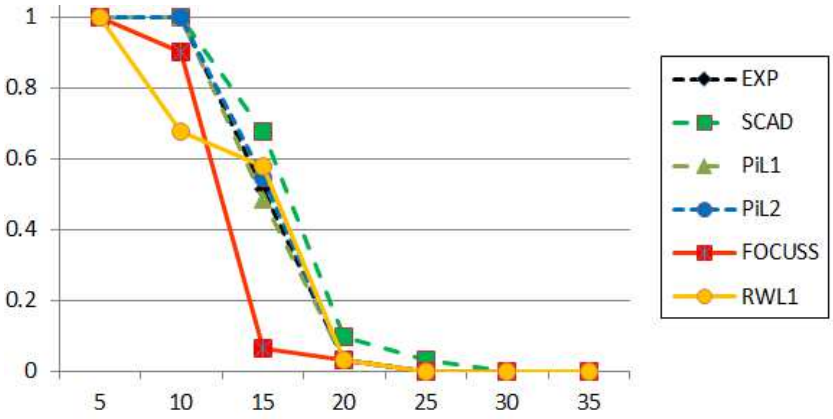


Figure 2.7: Success rates using incoherent sensing matrix, $m = 100, n = 2000$.

In the third scenario, we considered the case in which matrix A is highly coherent. Specifically, A is a random partial discrete cosine transform (DCT) matrix with size $m \times n = 100 \times 2000$ and its columns were computed by

$$A_i = \frac{1}{\sqrt{m}} \cos(2i\pi\xi_i/F), \quad \forall i = 1, \ldots, n \tag{2.41}$$

where ξ_i's are random vector uniformly distributed in $([0, 1]^m)$, and $F \in \mathbb{N}$ is a *refinement factor*.

The number F is closely related to the conditioning of A in the sense that larger F corresponds to large coherence of A. In our test, $F = 20$ and coherence of A was greater than 0.999. The sparsity of x was also in the set $\{5, 10, \ldots, 35\}$. Figure 2.8 presents the results of our experiment.

Comments on the results:
It is clear that the recovery problem is more difficult when the observed signal has fewer measurements or

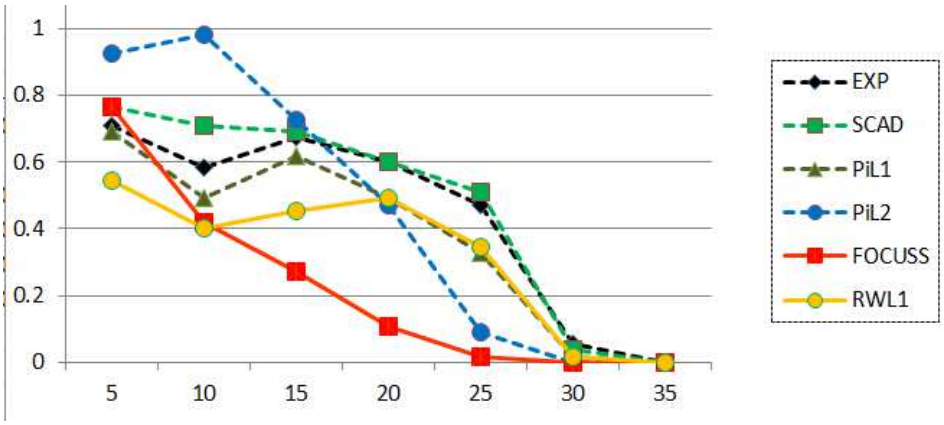


Figure 2.8: Success rates using highly coherent sensing matrix, $m = 100, n = 2000$.

when the sensing matrix has the higher coherence. When k is small and A is incoherent, all the algorithms recovery with high success rate. When k is greater or A is coherent, all the algorithms recovery with lower success rate.

In the first scenario, Figure 2.6, the six algorithms have the same performances but *LC-SCAD*, *LC-PiL2* and *RWL1* are slightly better than the others and *FOCUSS* is the worst algorithm.

In second scenario, there are some differences between the results of these algorithms. It can be seen in Figure 2.7 that when k is small, i.e., x is very sparse, all the DCA based algorithms have exactly recovered with higher success rate than *RWL1* and *FOCUSS*. But when k increases, the success rate decreases. The highest success rate belongs to *LC-SCAD*, the followings are *RWL1*, *LC-EXP*, *LC-PiL1* and *LC-PiL2*. *FOCUSS* has the lowest success rate in this case.

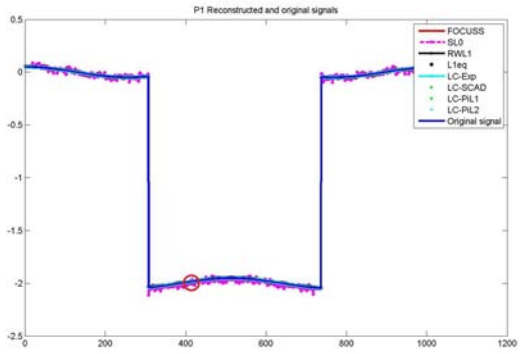
In the third scenario, Figure 2.8, when k is small, *LC-PiL2* and *LC-SCAD* have the highest success rate but when k becomes greater, *LC-SCAD* and *LC-EXP* have the highest success rate while *LC-PiL2* is worse. *FOCUSS* and *RWL1* give the success rate lower than the others. However in the case A is highly coherent and x is not very sparse, *RWL1* has a high rate of success.

The experiments show the efficiency of each approximation and also the power of DCA. It is observed that *LC-SCAD* gives the best performance in all tests even in case A is highly coherent. The following is *LC-EXP*. *LC-PiL2* and *LC-PiL1* are both good when A is incoherent or x is very sparse. All DCA based algorithms have higher success rate than *FOCUSS* and *RWL1* when A is highly coherent.

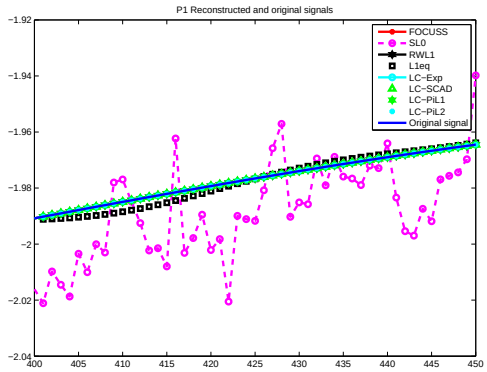
2.6 Conclusion

In the domain of compressed sensing, two challenging problems are sparse representation and sparse recovery. The optimization formulation of these problems involves the minimization of ℓ_0–norm which is known as a NP–hard problem.

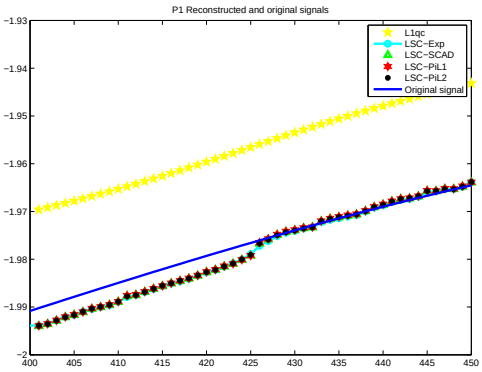
We have rigorously studied these problems with DC programming and DCA. Two problems are expressed in three models: linear constraint, least–square constraint and regularization least square. We investigated four approximations of ℓ_0–norm for each model which give birth to 12 nonconvex problems and solved them by DC algorithms. The comparison with 5 well-known algorithms (*FOCUSS, RWL1, SL0, $\ell 1eq$, $\ell 1qc$*) has showed the efficiency of our proposed algorithms.



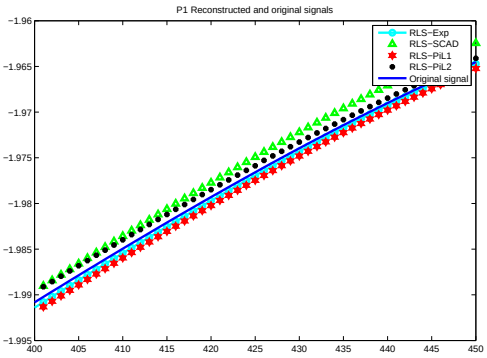
(a) Recovered signals



(b) Zoom of the recovered signals (in red circle) by using *Linear Constraint (LC)* model

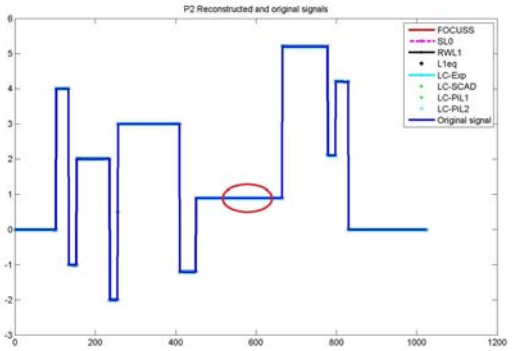


(c) Zoom of the recovered signals (in red circle) by using *Least Square Constrain (LSC)* model

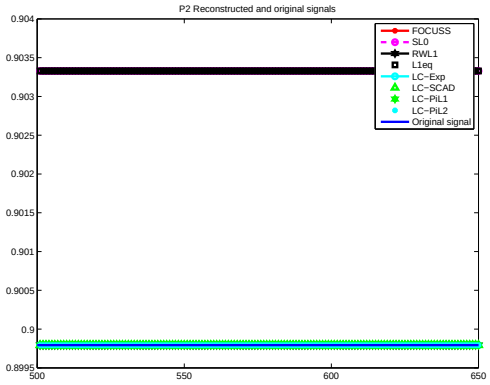


(d) Zoom of the recovered signals (in red circle) by using *Regularization Least Square (RLS)* model

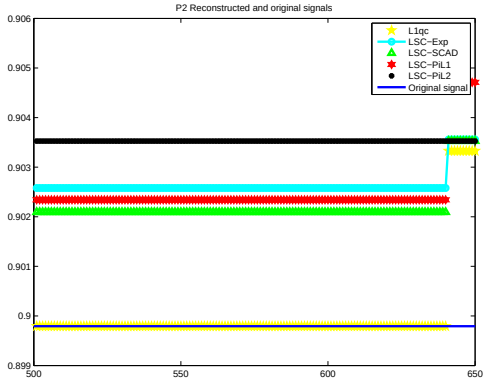
Figure 2.9: *Prob* 1. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc*



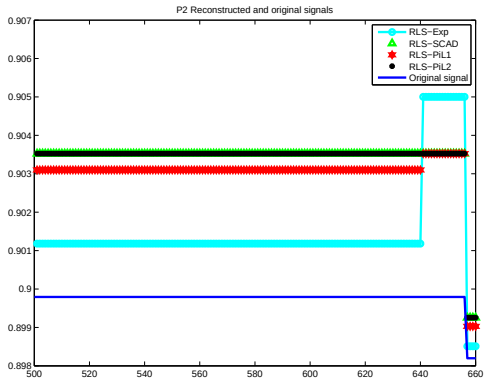
(a) Recovered signals



(b) Zoom of the recovered signals (in red circle) by using *Linear Constraint (LC)* model

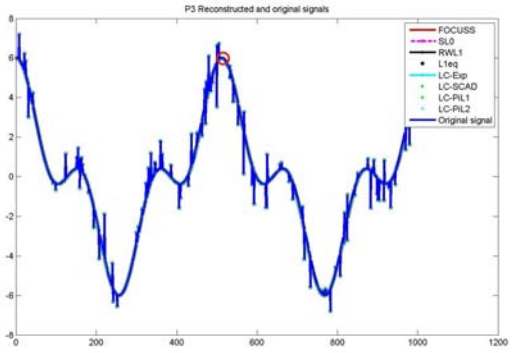


(c) Zoom of the recovered signals (in red circle) by using *Least Square Constrain (LSC)* model

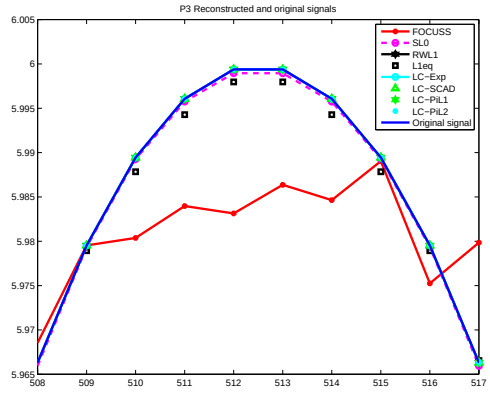


(d) Zoom of the recovered signals (in red circle) by using *Regularization Least Square (RLS)* model

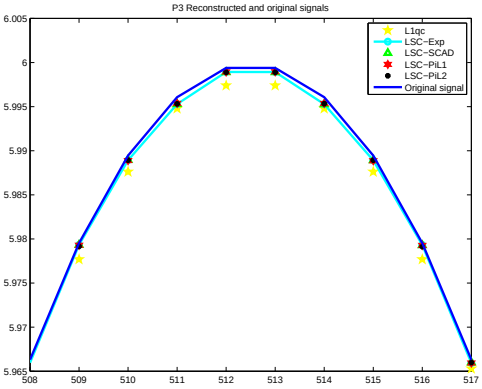
Figure 2.10: *Prob* 2. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc*



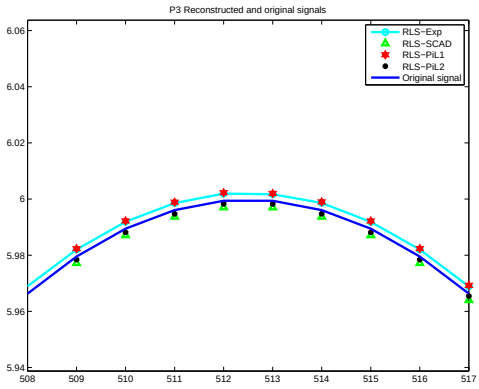
(a) Recovered signals



(b) Zoom of the recovered signals (in red circle) by using *Linear Constraint (LC)* model

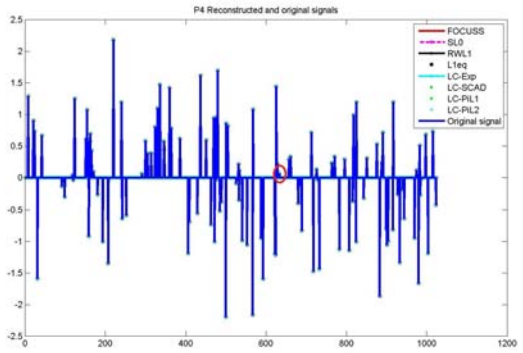


(c) Zoom of the recovered signals (in red circle) by using *Least Square Constrain (LSC)* model

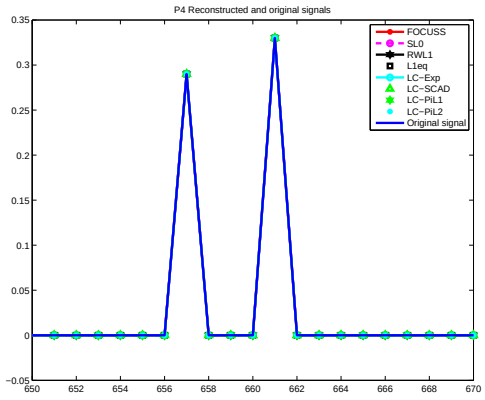


(d) Zoom of the recovered signals (in red circle) by using *Regularization Least Square (RLS)* model

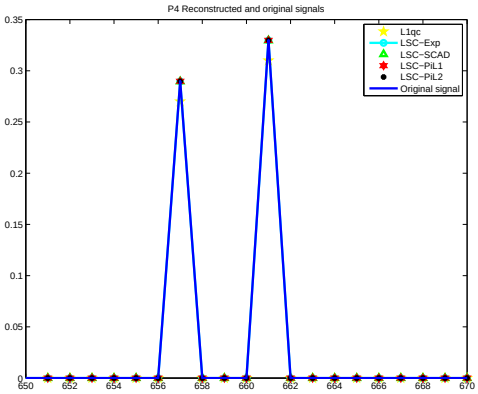
Figure 2.11: *Prob* 3. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc*



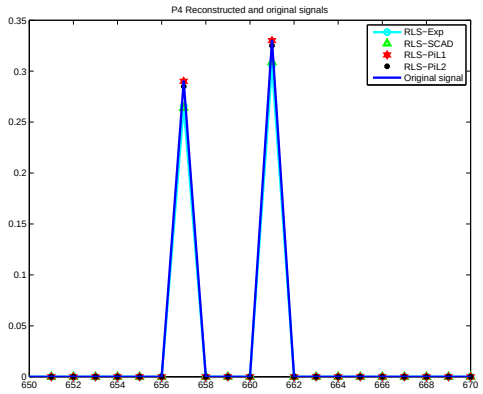
(a) Recovered signals



(b) Zoom of the recovered signals (in red circle) by using *Linear Constraint (LC)* model

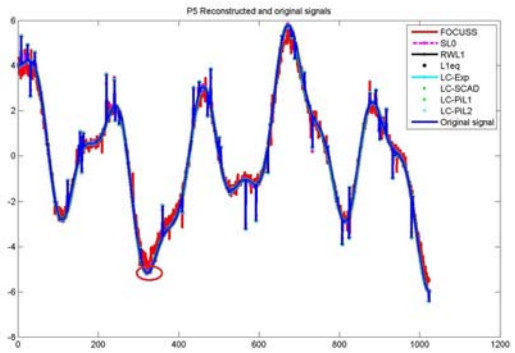


(c) Zoom of the recovered signals (in red circle) by using *Least Square Constrain (LSC)* model

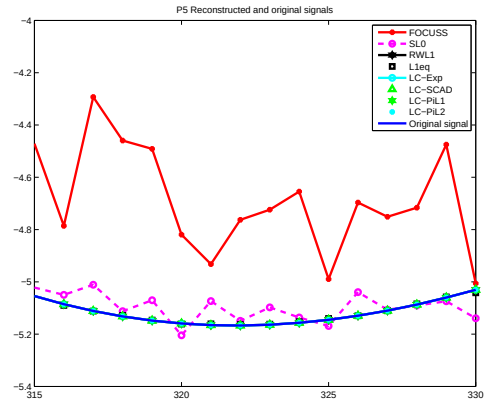


(d) Zoom of the recovered signals (in red circle) by using *Regularization Least Square (RLS)* model

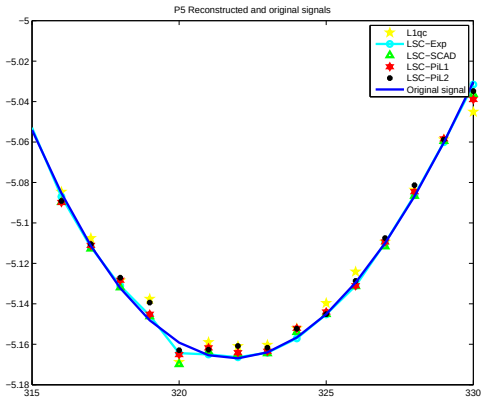
Figure 2.12: *Prob* 4. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc*



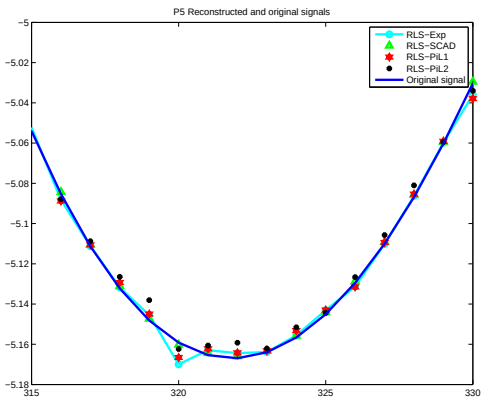
(a) Recovered signals



(b) Zoom of the recovered signals (in red circle) by using *Linear Constraint (LC)* model

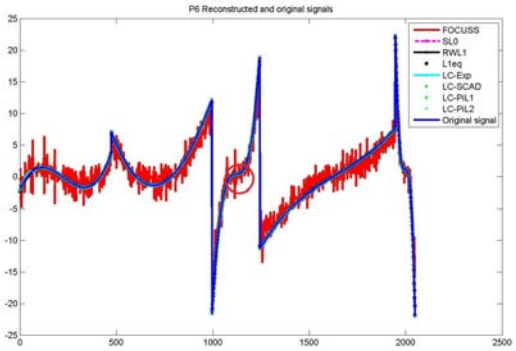


(c) Zoom of the recovered signals (in red circle) by using *Least Square Constrain (LSC)* model

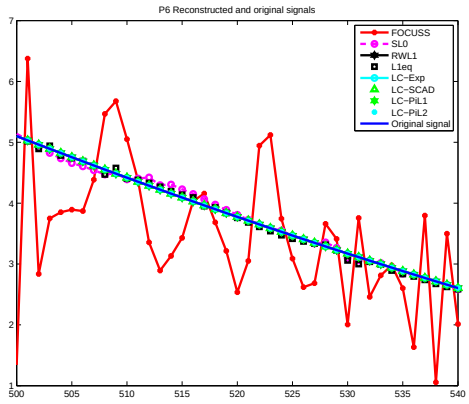


(d) Zoom of the recovered signals (in red circle) by using *Regularization Least Square (RLS)* model

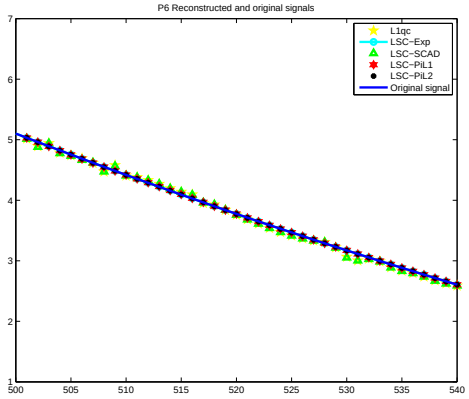
Figure 2.13: *Prob* 5. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc*



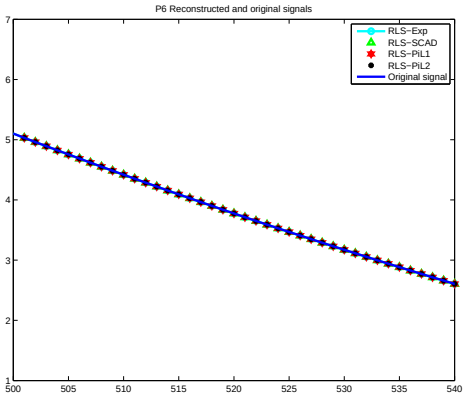
(a) Recovered signals



(b) Zoom of the recovered signals (in red circle) by using *Linear Constraint (LC)* model

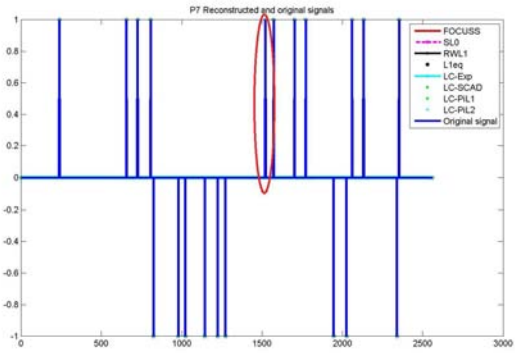


(c) Zoom of the recovered signals (in red circle) by using *Least Square Constrain (LSC)* model

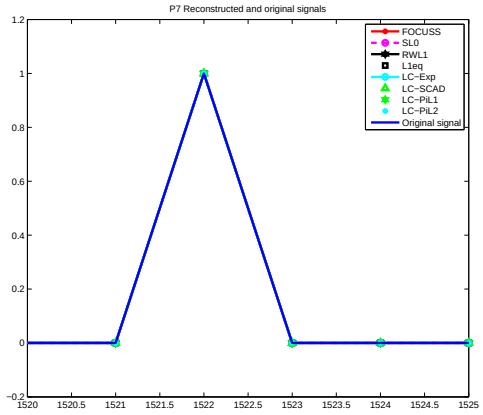


(d) Zoom of the recovered signals (in red circle) by using *Regularization Least Square (RLS)* model

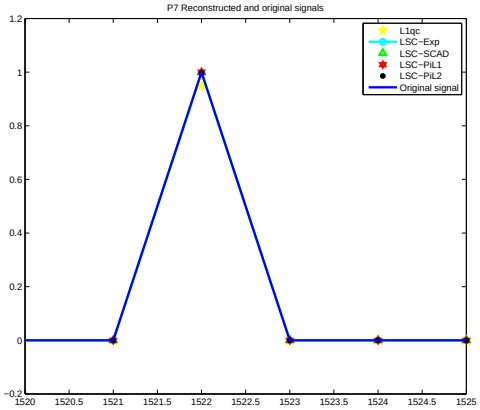
Figure 2.14: *Prob* 6. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, $\ell 1eq$, $\ell 1qc$*



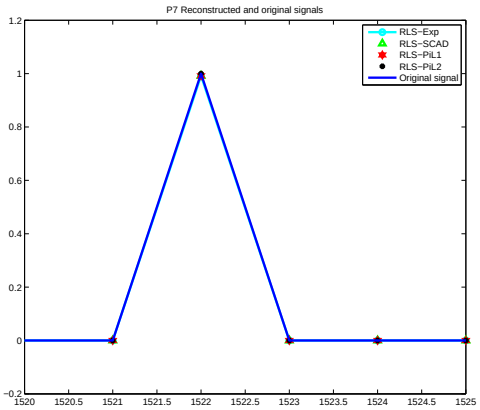
(a) Recovered signals



(b) Zoom of the recovered signals (in red circle) by using *Linear Constraint (LC)* model

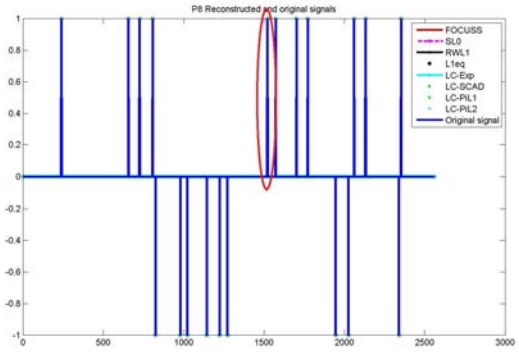


(c) Zoom of the recovered signals (in red circle) by using *Least Square Constrain (LSC)* model

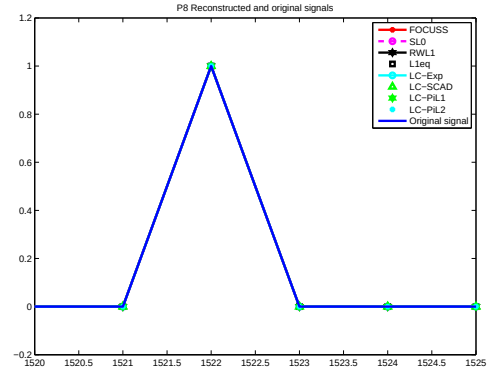


(d) Zoom of the recovered signals (in red circle) by using *Regularization Least Square (RLS)* model

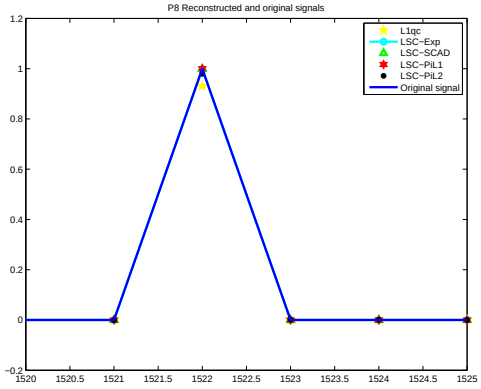
Figure 2.15: *Prob* 7. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc*



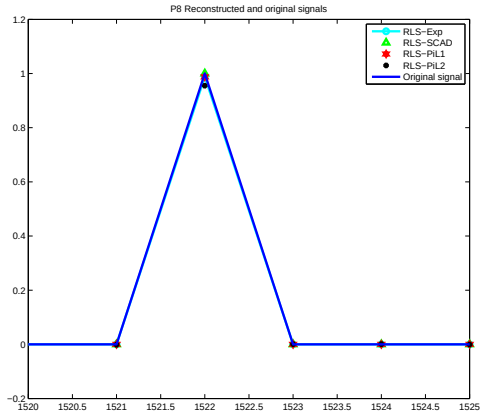
(a) Recovered signals



(b) Zoom of the recovered signals (in red circle) by using *Linear Constraint (LC)* model

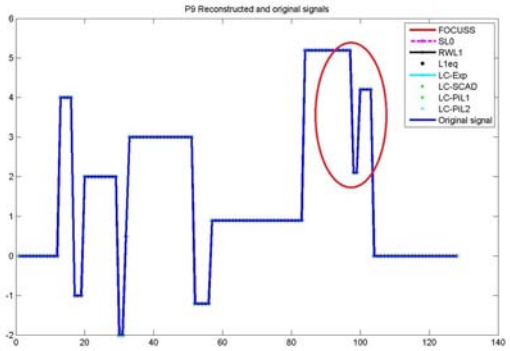


(c) Zoom of the recovered signals (in red circle) by using *Least Square Constrain (LSC)* model

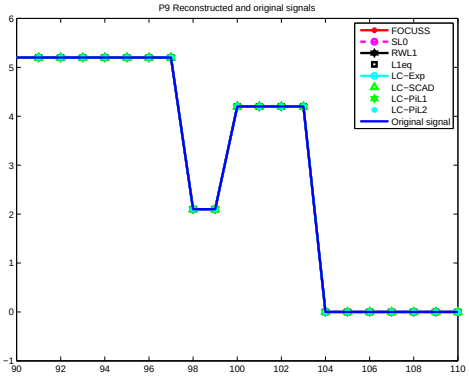


(d) Zoom of the recovered signals (in red circle) by using *Regularization Least Square (RLS)* model

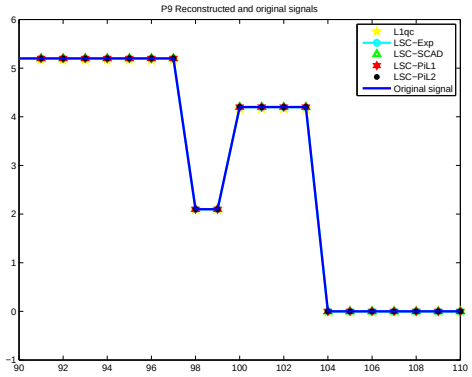
Figure 2.16: *Prob* 8. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc*



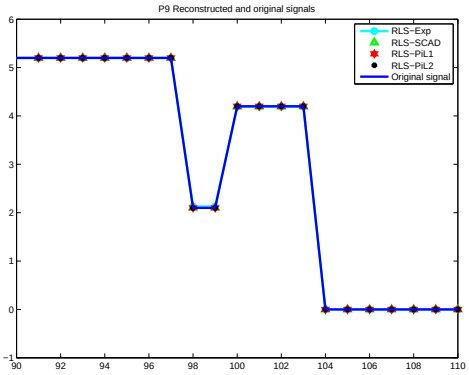
(a) Recovered signals



(b) Zoom of the recovered signals (in red circle) by using *Linear Constraint (LC)* model

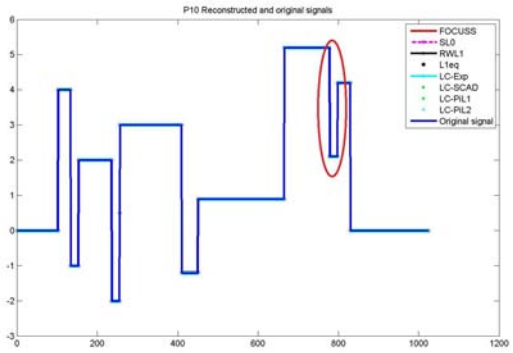


(c) Zoom of the recovered signals (in red circle) by using *Least Square Constrain (LSC)* model

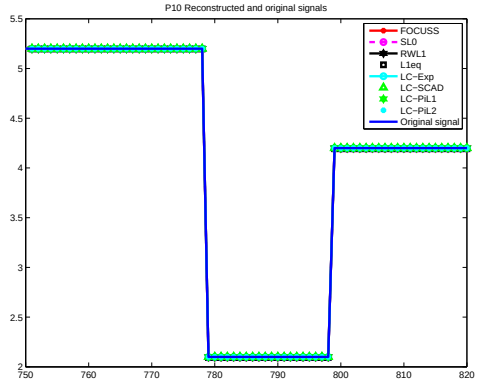


(d) Zoom of the recovered signals (in red circle) by using *Regularization Least Square (RLS)* model

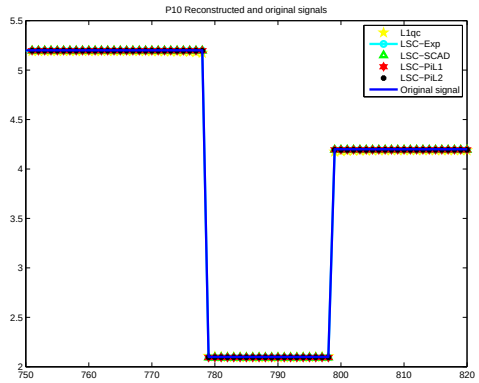
Figure 2.17: *Prob* 9. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc*



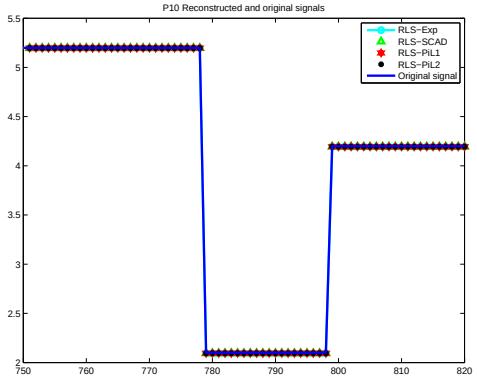
(a) Recovered signals



(b) Zoom of the recovered signals (in red circle) by using *Linear Constraint (LC)* model

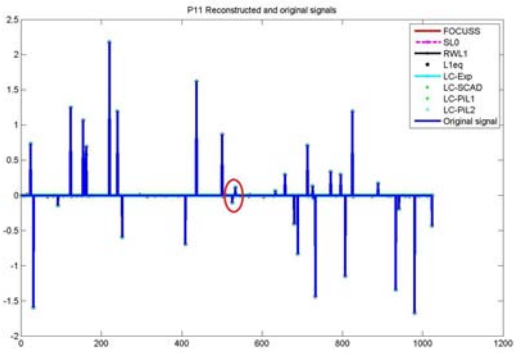


(c) Zoom of the recovered signals (in red circle) by using *Least Square Constrain (LSC)* model

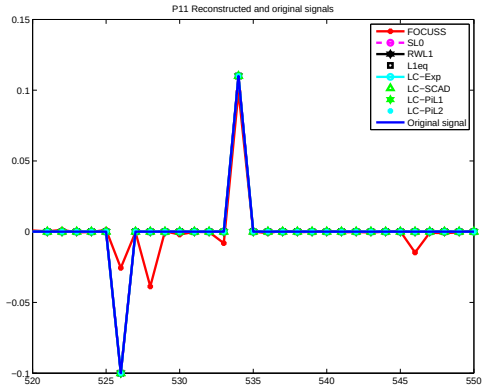


(d) Zoom of the recovered signals (in red circle) by using *Regularization Least Square (RLS)* model

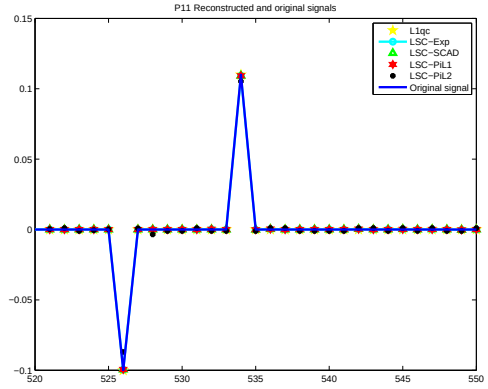
Figure 2.18: *Prob* 10. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc*



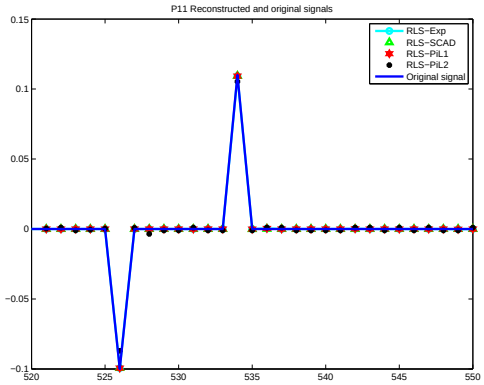
(a) Recovered signals



(b) Zoom of the recovered signals (in red circle) by using *Linear Constraint (LC)* model



(c) Zoom of the recovered signals (in red circle) by using *Least Square Constrain (LSC)* model



(d) Zoom of the recovered signals (in red circle) by using *Regularization Least Square (RLS)* model

Figure 2.19: *Prob* 11. Comparison of the recovered signals by using DCAs and 5 other algorithms: *FOCUSS, RWL1, SL0, ℓ1eq, ℓ1qc*

Chapter 3

Dictionary learning and application to image denoising

3.1 Introduction

The sparse representation of signals is attracting more and more researchers in recent years. It has proven to be extremely effective in many signal processing applications such as acquiring, representing and compressing high–dimensional signals.

In sparse signal representation, a basis or frame is referred to as a dictionary or an overcomplete dictionary, respectively, with the dictionary elements being called *atoms*.

It does not exist an universal dictionary that can represent every signal sparsely. So a dictionary is chosen based on the represented signals. There are two methods for building a dictionary: predefining a dictionary based on a mathematical model of the data (such as wavelets, contourlets, curvelets,...) (Mallat and Zhang [1993]) or learning a dictionary from the training set (Elad and Aharon [2006], Mairal et al. [2010], Olshausen and Field [1997]). It has been proved that the use of learned dictionaries instead of predefined dictionaries gives better results for many image processing tasks (Elad and Aharon [2006]).

The problem dictionary learning is stated as follows:

Given a set of L signals: $X = [x_1, \ldots, x_L], x_i \in \mathbb{R}^n$, finding a dictionary matrix D with k columns, such that every $x_i \in X$, can be represented as a sparse combination of columns of D. Denote the sparse signals as $W = [w_1, \ldots, w_L],\ w_i \in \mathbb{R}^k$, then the dictionary learning problem is to solve:

$$\min_{D \in\ \mathcal{C}, W} \frac{1}{2}\|X - DW\|_F^2 + \lambda \sum_{l=1}^{L} \|w_l\|_0, \quad \lambda > 0, \tag{3.1}$$

where $\mathcal{C} = \{D \in \mathbb{R}^{n\times k} : \|d_j\|_2 \leq 1\ \forall j = 1, \ldots, k\}$; $X \in \mathbb{R}^{n\times L}$; and $W \in \mathbb{R}^{k\times L}$; $n \ll k$; d_j denotes the j^{th} column of matrix D.

The first of (3.1) presents the error of sparse representation, while the remaining part describes the sparsity of the represent matrix. To prevent degeneracy, a normalization condition $||d_j||_2 \leq 1$ is usually placed on each column d_j of matrix D. λ is the non-negative parameter controlling the trade–off between data fitting and sparsity.

3.2 Related works

Most algorithms for dictionary learning (Aharon et al. [2006], Skretting and Engan [2010], Mairal et al. [2010]) iteratively alternate between two phases: sparse coding and dictionary updating. In the sparse coding phase, a sparse representation of signals is performed while the currently learned dictionary is fixed. In the dictionary updating phase, the learned dictionary is recomputed using the new sparse representation of signals.

In the first phase, the problem sparse coding finding one representation with the smallest number of atoms. Great efforts have been devoted to explore efficient and stable algorithms for solving this problem. Naturally, sparse coding leads to the ℓ_0–norm minimization problem:

$$\min_{W} \frac{1}{2}\|X - DW\|_F^2 + \lambda \sum_{l=1}^{L} \|w_l\|_0. \tag{3.2}$$

This problem is NP–hard (Mallat and Zhang [1993]). To deal with the fact that ℓ_0–norm is a discontinuous function, in several works, one approximates the ℓ_0–norm by a continuous, convex or nonconvex function. The previous chapter (Chapter 2) has presented these methods in greater detail.

In the second stage, the task is to learn dictionary from a set of data such that the error of sparse representation is minimum. It means that: finding D, an optimal solution of the problem:

$$\min_{D \in \mathcal{C}} \frac{1}{2}\|X - DW\|_F^2. \tag{3.3}$$

In the study of Olshausen and Field (Olshausen and Field [1997]), the authors used the gradient descent method for this problem. Engan et al. (Engan et al. [1999b,a]) and Aharon et al. (Aharon et al. [2006]) used *OMP* for sparse coding step, but the later investigated an algorithm for updating both the dictionary and its associated sparse coefficients simultaneously in updating step, while the former only updated the dictionary. We note that, the later study is one of the first studies on the powerfulness of dictionary learning in image denoising domain.

Another approach to update D using Newton method in a dual formulation is proposed by Lee et Seung (Lee et al. [2007]). In 2012, Wei Dai et al. (Dai et al. [2012]) introduced the *SimCO* algorithm (Simultaneous Codeword Optimization). This algorithm discovers the singular points, rather than local minima, which are the major bottleneck of dictionary update. To mitigate the problem caused by the singular points, regularized SimCO is proposed and then first and second order optimization have been used to solve this problem.

Based on maximum likelihood learning method, Kreutz-Delgado et al. (Kreutz-Delgado et al. [2003]) presented a maximum a posteriori (*MAP*) dictionary learning algorithm. This algorithm replaced the maximization of the likelihood function by the maximization of the posterior probability.

Engan et al. (Engan et al. [2007]) proposed the iterative least squares (*ILS*) method as a variant of *MOD*, which updates the dictionary after a batch of training vectors has been processed. In 2010, Skretting and Engan (Skretting and Engan [2010]) introduced a recursive least squares dictionary learning algorithm (*RLS-DLA*), which continuously update the dictionary as each training vector is being processed.

In Bolte et al. [2014], Bolte et al. introduced a proximal alternating linearized minimization (PALM) algorithm for solving nonconvex and nonsmooth minimization problems. This algorithm was applied to solve the nonnegative matrix factorization (NMF) problem, which has the form:

$$min\{\frac{1}{2}||A - XY||_F^2 : X \geq 0, Y \geq 0\}. \tag{3.4}$$

There are several studies in the field of online dictionary learning, such as: Mairal et al. (Mairal et al. [2010]), Rao et al. (Rao and Porikli [2012]) or Zhang et al. (Zhang et al. [2012]). Mairal et al. proposed

an online learning method, which is based on stochastic approximations. This method is suitable for a wide range of learning problems. Rao et al. introduced a new scheme to reduce and optimize dictionary size in an online setting by synthesizing new atoms from multiple previous ones. Zhang et al. presented an online method for Sparse Shape Composition problem. When new training shapes come, the method will update the existing one using a block coordinates descent approach instead of reconstructing the dictionary from the ground up.

Among the methods to deal with the sparse optimization problems, methods based on DC (Difference of Convex functions) programming and DCA (DC Algorithm) have appeared to be very efficient (Le Thi et al. [2008b], Le Thi et al. [2013c], Le Thi et al. [2014d], Le Thi et al. [2014b], ..). Motivated by these successes, we will investigate DC programming and DCA for solving the dictionary learning problem.

3.3 DC Programming and DCA for Dictionary learning

This section is organized as follows. First, we use the *PiL1* (or capped-ℓ_1) function, which was presented in the previous chapter (Chapter 2) and is proven in Le Thi et al. [2014d] as the best ℓ_0 approximation, to relax ℓ_0-norm for modeling sparsity in the dictionary learning problem. Second, we develop an algorithm based on DC programming and DCA to solve the new formulation of dictionary learning problem.

Throughout the section, the absolute matrix $|X|$ is defined as $[|X|]_{ij} = |X_{ij}|$ for all i, j. X_{IJ} indicates sub–matrix with row (resp. column) indices taken from I (resp. J). $X_{i:}$ (resp. $X_{:j}$) denotes the i^{th} row (resp. j^{th} column) of the matrix X. The notation $\circ$ (resp. $\frac{[.]}{[.]}$) denotes the component–wise product (resp. division) of matrices.

The general schema of this problem as follows:

Schema solution:

- Sparse coding stage: fix D, update W. This stage will leads to solve the problem:

$$W_{:l} \in \arg\min \left\{ \frac{1}{2}\|x_l - Dw\|^2 + \lambda\Phi(w) : w \in \mathbb{R}^k \right\} \quad \forall l = 1, \ldots, L. \tag{3.5}$$

- Dictionary updating stage: fix W, update D by solving the problem:

$$\min_{D \in \mathcal{C}} \frac{1}{2}\langle A, D^T D\rangle - \langle B, D\rangle, \tag{3.6}$$

  where $A = WW^T, B = XW^T$; and $\langle A, B\rangle$ denotes the inner product of two matrices A, B: $\langle A, B\rangle = \text{trace}(A^T B)$.

In the next part, we present DC formulation and DCA for solving the subproblem on each stage.

3.3.1 Sparse coding stage, update W

Recall that the PiL1 function has the form as:

$$\phi(t) = \min(1, \alpha|t|), \quad \alpha > 0, t \in \mathbb{R}. \tag{3.7}$$

Using PiL1 function, the problem of dictionary learning is expressed as a minimization problem:

$$F(D,W) = \frac{1}{2}\|X - DW\|_F^2 + \lambda \sum_{l=1}^{L} \Phi(w_l), \quad \lambda > 0, \tag{3.8}$$

where:

$$\Phi(w) = \sum_{i=1}^{k} \phi(w_{(i)}) \quad \text{with} \quad w = (w_{(1)}, \ldots, w_{(k)})^T. \tag{3.9}$$

(T denotes the transpose vector, and $w_{(i)}$ is i-th element of vector w.)

For convenience, we omit the subscript of x, the problem (3.5) becomes an optimization problem as follows:

$$\min_{w \in \mathbb{R}^k} \left\{ f_D(w) = \frac{1}{2}\|x - Dw\|^2 + \lambda \Phi(w) \right\} \tag{3.10}$$

where $D \in \mathbb{R}^{n \times k}$ and fixed, $x \in \mathbb{R}^n$.

From formulation (3.9), $\Phi(w)$ is a DC function with a DC decomposition given by:

$$\Phi(w) = k + \alpha\|w\|_1 - \sum_{i=1}^{k} \max(1, \alpha|w_i|).$$

So f_D is also a DC function and has a DC decomposition:

$$f_D(w) = g_D(w) - h_D(w), \tag{3.11}$$

where:

$$g_D(w) = \frac{1}{2}\|x - Dw\|^2 + \lambda\alpha\|w\|_1 + k\lambda,$$
$$h_D(w) = \lambda \sum_{j=1}^{k} \max(1, \alpha|w_i|).$$

Then DCA for solving problem (3.11) is simply as follows (for $l = 0, 1, 2, \ldots$):

– Calculate $y^l \in \partial h_D(w^l)$.
– Calculate $w^{l+1} \in \arg\min\{\frac{1}{2}\|x - Dw\|^2 + \lambda\alpha\|w\|_1 - \langle w, y^l \rangle : w \in \mathbb{R}^k\}$. (P_l).

Computation of $y^l \in \partial h_D(w^l)$ is explicitly given by:

$$y \in \partial h_D(w) \Leftrightarrow \begin{cases} y_j = 0 & \text{if } |w_j| < \frac{1}{\alpha} \\ y_j \in \text{sign}(w_j)[0, \lambda\alpha] & \text{if } |w_j| = \frac{1}{\alpha} \\ y_j = \text{sign}(w_j)\lambda\alpha & \text{otherwise,} \end{cases} \quad \forall j = 1, \ldots, k. \tag{3.12}$$

We will discuss on how to solve problem (P_l) below.

DCA for solving problem (P_l).

If we omit the subscript of y, then the problem (P_l) takes the form:

$$\min_{w \in \mathbb{R}^k} \overline{f}_D(w) := \frac{1}{2}\|x - Dw\|^2 + \lambda\alpha\|w\|_1 - \langle w, y \rangle. \tag{P_l}$$

Let $\rho \in \mathbb{R}^k_{++}$ s.t. $D^T D \preceq \text{diag}(\rho)$, then $\overline{f}_D$ has a DC decomposition $\overline{f}_D = \overline{g}_D - \overline{h}_D$, given by:

$$\overline{g}_D(w) = \sum_{j=1}^{k} \Big(\frac{1}{2}\rho_j w_j^2 + \lambda\alpha|w_j| - y_j w_j\Big),$$

$$\overline{h}_D(w) = \sum_{j=1}^{k} (\frac{1}{2}\rho_j w_j^2) - \frac{1}{2}\|x - Dw\|^2.$$

DCA for solving problem (P_l) consists of (for $t = 0, 1, 2, \ldots$):

- Compute $z^t = \nabla\overline{h}_D(w^t) = \text{diag}(\rho)w^t - D^T(Dw^t - x)$
- Compute:

$$w^{t+1} \in \arg\min\Big\{\sum_{j=1}^{k}\Big(\frac{1}{2}\rho_j(w_j)^2 + \lambda\alpha|w_j| - y_j w_j - z_j^t w_j\Big) : w \in \mathbb{R}^k\Big\}$$

$$\Leftrightarrow \quad w_j^{t+1} = \arg\min_{w_j} \frac{1}{2}\left(w_j - \frac{y_j + z_j^t}{\rho_j}\right) + \frac{\lambda\alpha}{\rho_j}|w_j| = \frac{S(y_j + z_j^t, \lambda\alpha)}{\rho_j} \quad \forall j = 1, \ldots, k,$$

where $S(u, \beta) = \text{sign}(u)(|u| - \beta)_+$ is the soft thresholding formula.

For simplicity, we rewrite the above updating rule in vector form as follows:

$$w^{t+1} = \frac{[S(y + z^t, \lambda\alpha)]}{[\rho]}, \tag{3.13}$$

where the operation S is component–wise, i.e. $S(a, b) = (S(a_i, b_i))_i$.

The following theorem gives a way to choose ρ.

Theorem 1 *Let $Q \in \mathbb{R}^{d\times d}$ be a symmetric matrix and $\rho \in \mathbb{R}^d$ be a vector given by $\rho = |Q|\mathbf{1}_{d\times 1}$, i.e. $\rho_i = \sum_{j=1}^{d} |Q_{ij}|$ for all $i = 1, \ldots, d$. Then we have:*

$$Q \preceq \textit{diag}(\rho).$$

Proof : For any $x \in \mathbb{R}^d$ we have:

$$\begin{aligned} x^T(\text{diag}(\rho) - Q)x &= \sum_{i=1}^{d}\rho_i x_i^2 - \sum_{i,j=1}^{d} Q_{ij}x_i x_j = \sum_{i,j=1}^{d}|Q_{ij}|x_i^2 - \sum_{i,j=1}^{d} Q_{ij}x_i x_j \\ &= \frac{1}{2}\sum_{i,j=1}^{d}|Q_{ij}|(x_i^2 + x_j^2) - \sum_{i,j=1}^{d} Q_{ij}x_i x_j \\ &\geq \frac{1}{2}\sum_{i,j=1}^{d}|Q_{ij}|(|x_i| - |x_j|)^2 \geq 0. \end{aligned}$$

This implies that $Q \preceq \text{diag}(\rho)$. □

According to Theorem 1, we can chose $\rho = |D^T D|\mathbf{1}_{k\times 1}$. However, we can do in a more effective way. Observe that if $w_j^t = 0$ and $|D_{:j}^T(Dw - x) - y_j| \leq \lambda\alpha$, we have $S(y_j + z_j^t, \lambda\alpha) = 0$ for any choice of ρ. Thus, the updating rule (3.13) makes no change on the components j^{th} of w^{t+1}.

Define:

$$I(w,y) = \{j = 1,\ldots,k : w_j \neq 0 \text{ or } |D_{:j}^T(Dw - x) - y_j| > \lambda\alpha\}, \quad w, y \in \mathbb{R}^k,$$

then at the iteration t^{th}, we only need to consider variables $\{w_j : j \in I\}$ (here $I = I(w^t, y)$ for short). Repeat the above procedure with w_I (resp. $D_{:I}$ and y_I) replacing w (resp. D and y), we compute:

$$z_I^t = \text{diag}(\rho_I)w_I^t - D_{:I}^T(D_{:I}w_I^t - x_I)$$

and:

$$w_I^{t+1} = \frac{[S(y_I + z_I^t, \lambda\alpha)]}{[\rho_I]}, \quad w_j^{t+1} = 0, \quad \forall j \notin I,$$

where $\rho_I = |D_{:I}^T D_{:I}|\mathbf{1}_{|I|\times 1}$ and $\rho_j = 0\ \forall j \notin I$.

These are equivalent to compute:

$$\omega = \frac{[p]}{[|D^T D|p]},$$

where $p \in \mathbb{R}^k$, that is defined by: $p_j = 1$ if $j \in I$ and $p_j = 0$, otherwise.

Then we compute:

$$z^t = w^t - (D^T(Dw^t - x) - y) \circ \omega, \tag{3.14}$$

$$w^{t+1} = S(z^t, \lambda\alpha\omega). \tag{3.15}$$

Proposition 3.1 *Under the updating rule (3.14)–(3.15), the function $\overline{f}_D$ is decreasing. Moreover, if w^t is not a stationary point (also global solution since problem (P_l) is convex) of problem (P_l) then $\overline{f}_D(w^{t+1}) < \overline{f}_D(w^t)$.*

Proof : Since $\overline{f}_D$ is convex, w is a solution of problem (P_l) if and only if:

$$0 \in \partial\bar{f}_D(w) \Leftrightarrow \begin{cases} |D_{:j}^T(Dw - x) - y_j| \leq \lambda\alpha & \text{if } w_j = 0, \\ D_{:j}^T(Dw - x) - y_j = -\text{sign}(w_j) & \text{if } w_j \neq 0, \end{cases} \forall j = 1,\ldots,k. \tag{3.16}$$

Thus, the variables $\{w_j : j \notin I(w,y)\}$ are already satisfying condition (3.16). By restricting problem (P_l) to variables $\{w_j : j \in I(w,y)\}$ the assertions of this proposition are consequences of general convergence properties of DCA. □

Note that, in the context of dictionary learning, w is expected to be very sparse. This implies that very few components of w need to be updated (corresponding to $\omega_j \neq 0$). To exploit this fact for solving problem (P_l), we will not calculate ω after each iteration. Instead, we only compute ω from the beginning and keep using it later on. This means that we do not actually solve problem (P_l). However, we will show that the convergence of the algorithm for solving problem (3.10) is still guaranteed.

We are now in a position to describe the DCA for solving problem (3.10).

DC algorithm for the sparse coding stage:

Initialization: Initialize $w^0 \in \mathbb{R}^k$, $T > 0$ (maximum number of inner–iterations), $\epsilon > 0$ (stopping tolerance), $l \leftarrow 0$

Repeat

Compute $y^l \in \partial h(w^l)$ by:

$$y_i^l = \begin{cases} 0 & \text{if } |w_i^l| \le \frac{1}{\alpha} \\ \text{sign}(w_i^l)\lambda\alpha & \text{otherwise,} \end{cases} \quad \forall i = 1, \dots, k.$$

Compute p^l by:

$$p_j^l = \begin{cases} 1 & \text{if } j \in I(w^l, y^l), \\ 0 & \text{otherwise,} \end{cases} \quad \forall j = 1, \dots, k.$$

Compute $\omega^l = \frac{[p^l]}{[|D^T D|p^l]}$, $w^{(l,0)} = w^l$,

For t = 1,. . . ,T:

– Compute $z^t = w^{(l,t-1)} - (D^T(Dw^{(l,t-1)} - x) - y^l) \circ \omega^l$.

– Compute $w^{(l,t)} = S(z^t, \lambda\alpha\omega^l)$.

If $\|w^{(l,t-1)} - w^{(l,t)}\| < \epsilon$

Break;

End If

End For

Set $w^{l+1} = w^{(l,t)}$.

Set $l \leftarrow l + 1$.

Until $\|w^{l+1} - w^l\| < \epsilon$.

Before going to the result concerning convergence of this algorithm, we describe characteristics of critical point of problem (3.10). We have:

$$y \in \partial g_D(w) \Leftrightarrow \begin{cases} y_j - D_{:j}^T(Dw - x) = \text{sign}(w_j)\lambda\alpha & \text{if } w_j \neq 0, \\ y_j - D_{:j}^T(Dw - x) \in [-\lambda\alpha, \lambda\alpha] & \text{if } w_j = 0, \end{cases} \quad \forall j = 1, \dots, k$$

$$y \in \partial h_D(w) \Leftrightarrow \begin{cases} y_j = 0 & \text{if } |w_j| < \frac{1}{\alpha}, \\ y_j \in \text{sign}(w_j)[0, \lambda\alpha] & \text{if } |w_j| = \frac{1}{\alpha}, \\ y_j = \text{sign}(w_j)\lambda\alpha & \text{if } |w_j| > \frac{1}{\alpha}. \end{cases} \quad \forall j = 1, \dots, k$$

Therefore, w is a critical point of (3.10) (i. e. $\partial g_D(w) \cap \partial h_D(w) \neq \emptyset$) if and only if:

$$\begin{cases} D_{:j}^T(Dw - x) \in [-\lambda\alpha, \lambda\alpha] & \text{if } w_j = 0, \\ D_{:j}^T(Dw - x) = -\text{sign}(w_j)\lambda\alpha & \text{if } |w_j| \in (0, \frac{1}{\alpha}), \\ D_{:j}^T(Dw - x) \in -\text{sign}(w_j)[0, \lambda\alpha] & \text{if } |w_j| = \frac{1}{\alpha}, \\ D_{:j}^T(Dw - x) = 0 & \text{if if } |w_j| > \frac{1}{\alpha}. \end{cases} \quad \forall j = 1, \dots, k \tag{3.17}$$

Theorem 2 *Suppose that $\{w^l\}$ is the sequence generated by Algorithm (**DC algorithm for the sparse coding stage**). Then $\{f_D(w^l)\}$ is a decreasing sequence and any limit point of the sequence $\{w^l\}$ is a critical point of problem (3.10).*

Proof : For any $l = 0, 1, 2, \dots$, we have:

$$f_D(w) \le \bar{f}_D(w) + C, \quad \forall w \in \mathbb{R}^k,$$

where $C = k\lambda - h_D(w^l) + \langle y^l, w^l \rangle$, and the equality holds if $w = w^l$. Thus, by Proposition 3.1:

$$f_D(w^{l+1}) \le \bar{f}_D(w^{l+1}) + C \le \bar{f}_D(w^{(l,1)}) + C \le \bar{f}_D(w^{(l,0)}) + C = f_D(w^l).$$

The first assertion is proved. Moreover, if w^l is not a critical point of problem (3.10), then $y^l \notin \partial g_D(w^l)$. This also means that $x^l \equiv x^{(l,0)}$ is not a critical point of problem (P_l) (not satisfying condition (3.16)) and that $I(w^l) \neq \emptyset$. By Proposition 3.1, $\bar{f}_D(w^{(l,1)}) < \bar{f}_D(w^{(l,0)})$, and consequently $f_D(w^{l+1}) < f_D(w^l)$. This implies that if $f_D(w^{l+1}) = f_D(w^l)$ then w^l is a critical point of problem (3.10) and algorithm for the sparse coding stage terminates at the l^{th} iteration.

Assume that w^* is an arbitrary limit point of the sequence $\{w^l\}_{l=0}^{\infty}$. Consider any sub-sequence $\{w^l\}_{l\in\mathcal{R}}$ with $\mathcal{R} \subseteq \{0, 1, 2, \ldots\}$ converging to w^*. Then we have:

$$f_D(w^*) = \lim_{l\in\mathcal{R}, l\to+\infty} f_D(w^l) = \inf_{l=0,1,2,\ldots} f_D(w^l) \geq 0. \tag{3.18}$$

Note that $\{y^l\}$ (resp.$\{p^l\}$ and $\{\omega^l\}$) generated by **DC algorithm for the sparse coding stage** has finite value. Thus, by passing to a subsequence, if necessary, we can assume that for any $l \in \mathcal{R}$, $y^l = y^*$, $p^l = p^*$ and $\omega^l = \omega^*$, for some $y^* \in \{0, \lambda\alpha, -\lambda\alpha\}^k$, $p^* \in \{0, 1\}^k$ and $\omega^* = \frac{[p^*]}{[|D^T D|p^*]}$. Moreover, we also assume that for any $l \in \mathcal{R}$, computing w^{l+1} from w^l (loop **for** in this algorithm) takes the same number of inner iterations $t^* \in \{1, \ldots, T\}$.

Consider the function $\psi : \mathbb{R}^k \to \mathbb{R}^k$ defined by:

$$\psi(w) = S(w - (D^T(Dw - x) - y^*) \circ \omega^*, \lambda\alpha\omega^*), \quad w \in \mathbb{R}^k.$$

We have ψ and $\Psi = \psi \circ \cdots \circ \psi$ (t^* times) are continuous functions, and:

$$w^{l+1} = \Psi(w^l), \quad \forall l \in \mathcal{R}.$$

This implies that $\{w^{l+1}\}_{l\in\mathcal{R}}$ converges to $\Psi(w^*)$ and $f_D(\Psi(w^*)) = f_D(w^*)$.
Moreover, since $\{w^l\}_{l\in\mathcal{R}}$ converges to w^*, there is an l_0 such that for any $l \in \mathcal{R}$ and $l \geq l_0$,

$$\begin{aligned}
&I(w^*, y^*) \subseteq I(w^l, y^*) = \{j : p_j^* = 1\},\\
&\left\{j : |w_j^*| < \frac{1}{\alpha}\right\} \subseteq \left\{j : |w_j^l| < \frac{1}{\alpha}\right\} \subseteq \left\{j : y_j^* = 0\right\},\\
&\left\{j : w_j^* > \frac{1}{\alpha}\right\} \subseteq \left\{j : w_j^l > \frac{1}{\alpha}\right\} = \left\{j : y_j^* = \lambda\alpha\right\},\\
&\left\{j : w_j^* < -\frac{1}{\alpha}\right\} \subseteq \left\{j : w_j^l < -\frac{1}{\alpha}\right\} = \left\{j : y_j^* = -\lambda\alpha\right\},\\
&y_j^* \in \{0, \lambda\alpha\} \text{ if } w_j^* = \frac{1}{\alpha}, \quad y_j^* \in \{0, -\lambda\alpha\} \text{ if } w_j^* = -\frac{1}{\alpha}.
\end{aligned}$$

Therefore, $y^* \in \partial h_D(w^*)$. By the same arguments as ones at the beginning of this proof, we have w^* is a critical point of problem (3.10). □

3.3.2 Dictionary updating: update D

For updating D we solve the optimization of the form:

$$\min_{D\in\mathcal{C}} f_W(D) := \frac{1}{2}\langle A, D^T D\rangle - \langle B, D\rangle, \tag{3.19}$$

where $A = WW^T, B = XW^T$.

Let $\gamma = |A|\mathbf{1}_{k\times 1}$, we can decompose f_W as:

$$f_W = g_W - h_W, \tag{3.20}$$

where g_W and h_W are given by:

$$\begin{aligned} g_W(D) &= \frac{1}{2}\sum_{j=1}^{k}\gamma_j\|D_{:j}\|^2, \\ h_W(D) &= \frac{1}{2}\sum_{j=1}^{k}\gamma_j\|D_{:j}\|^2 - \left(\frac{1}{2}\langle A, D^T D\rangle - \langle B, D\rangle\right). \end{aligned}$$

It is clearly that g_W is convex and h_W is also a convex function by Theorem 1 and the fact that:

$$h_W(D) = \sum_{i=1}^{n}\left[\frac{1}{2}\sum_{j=1}^{k}\gamma_j D_{ij}^2 - \left(\frac{1}{2}D_{i:}AD_{i:}^T - \langle B_{i:}, D_{i:}\rangle\right)\right].$$

DCA applied for the problem (3.19) with DC decomposition (3.20) consists of two steps:

- Compute $\bar{D}^{(t)} = \nabla h_W(D^{(t)}) = \Gamma \circ D^{(t)} - (D^{(t)}A - B)$, where $\Gamma \in \mathbb{R}^{n\times k}$ is the matrix defined by $\Gamma_{i:} = \gamma$, $i = 1, \ldots, n$.
- Compute

$$\begin{aligned} D^{(t+1)} &= \arg\min\left\{g_W(D) - \langle \bar{D}^{(t)}, D\rangle : D \in \mathcal{C}\right\} \\ &= \arg\min\left\{\sum_{j=1}^{k}\left(\frac{1}{2}\gamma_j\|D_{:j}\|^2 - \langle \bar{D}_{:j}^{(t)}, D_{:j}\rangle\right) : D \in \mathcal{C}\right\} \\ &= \arg\min\left\{\sum_{j=1}^{k}\frac{1}{2}\left\|D_{:j} - \frac{1}{\gamma_j}\bar{D}_{:j}^{(t)}\right\|^2 : D \in \mathcal{C}\right\} \end{aligned}$$

$$\Leftrightarrow D_{:j}^{(t+1)} = \operatorname*{Proj}_{\|D_{:j}\|\le 1}\frac{\bar{D}_{:j}^{(t)}}{\gamma_j} = \frac{\bar{D}_{:j}^{(t)}}{\max\{\gamma_j, \|\bar{D}_{:j}^{(t)}\|\}}, \quad \forall j = 1, \ldots, k. \tag{3.21}$$

We summarize this procedure in the following algorithm.

DC algorithm for the dictionary updating stage:

Initialization:Initial matrix $D^{(0)} \in \mathcal{C}$, $t \leftarrow 0$

Repeat

Compute $\overline{D}^{(t)} = \Gamma \circ D^{(t)} - (D^{(t)}A - B)$.

Compute $D^{(t+1)}$ by (3.21).

Set $t \leftarrow t + 1$.

Until $\|D^{(t+1)} - D^{(t)}\| < \epsilon$.

Since the problem (3.19) is convex, general convergence of DCA implies that:

Theorem 3 *Any limit point D^* of the sequence $\{D^{(t)}\}$ generated by above algorithm is a global solution of problem (3.19).*

3.4 Application to image denoising

In this section, we are interested in the image denoising problem. During the image acquisition, due to the effect of the acquisition device as well as of the environment, the observed image will be affected by noise. Denoting by x the true image, the observed image y may be modeled as:

$$y = x + \eta,$$

where η denotes the additive noise variable.

In Aharon et al. [2006], the authors presented a K-SVD based algorithm for denoising of gray images with additive homogeneous white Gaussian noise. In our work, we construct our algorithm based on the framework of this work. Here we will use DCA to solve two subproblems "sparse representation" and "dictionary learning" as mentioned in section 3.3.1 and section 3.3.2 instead of OMP and K-SVD.

Assume that all patches with size $\sqrt{n} \times \sqrt{n}$ are extracted from the original image x_{ij}, which admit a sparse representation. The denoising problem using sparse decomposition technique per each patch leads to the following minimization problem:

$$\min_{D \in \mathcal{C}, W, x} \beta \|x - y\|^2 + \lambda \sum_{ij} \|w_{ij}\|_0 + \sum_{ij} \|Dw_{ij} - R_{ij}x\|^2. \tag{3.22}$$

In this formulation, $D \in \mathbb{R}^{n \times L}$ is the dictionary used to represent the patches in the recovered image; $[i, j]$ is the position of the patch in the image (representing its top-left corner). The vectors $w_{ij} \in \mathbb{R}^L$ are the sparse representation for the $[i, j]$–th patch in x using the dictionary D. The operator R_{ij} is a binary $n \times N$ matrix, which extracts the square $\sqrt{n} \times \sqrt{n}$ patch of coordinate $[i, j]$ from the image written as a column vector. The first term in (3.22) is the likelihood force that demands the proximity between the measured image y and its denoised version x. The second term provides the sparsest representation and the third term ensures the consistency of the decomposition.

For solving this problem, firstly, we must learn a dictionary D from several patches (also known as a training set). These patches are taken from observed image with an overlap of 1 pixel. Each patch with size $\sqrt{n} \times \sqrt{n}$ will be converted to a vector $x_i \in \mathbb{R}^n$. We have a training set X consist of L signal x_i. This leads to the dictionary learning problem (3.1):

$$\min_{D \in \mathcal{C}, W} \frac{1}{2} \|X - DW\|^2 + \lambda \sum_{l=1}^{L} \|w_l\|_0, \quad \lambda > 0.$$

This problem will be solved by a DCA based algorithm with two stages: sparse coding and dictionary updating stage, that have been presented above.

Secondly, the sparse representations of all the patches on image are computed by solving the minimization problem:

$$\min_{w_{ij}} \|x_{ij} - Dw_{ij}\|^2 + \lambda \|w_{ij}\|_0. \tag{3.23}$$

We will use algorithm introduced in section 3.3.1 to solve this problem.

Finally, from (3.22), given all w_{ij} and D we can calculate x. It leads to solve

$$\min_{x} \beta \|x - y\|^2 + \sum_{ij} ||Dw_{ij} - R_{ij}x||_2^2. \tag{3.24}$$

This is a quadratic problem and it can be solved directly, which has the solution of the form:

$$x = (\beta \mathbf{I} + \sum_{ij} R_{ij}^T)^{-1} (\beta y + \sum_{ij} R_{ij}^T D w_{ij}). \tag{3.25}$$

Test Protocol

In this section, we present the results achieved by applying our methods on 55 gray scale images. We compare the efficiency of our algorithm with K–SVD, a standard algorithm, which is introduced by Michal Aharon et al. in 2006 (Aharon et al. [2006]) and proximal alternating linearized minimization (PALM) (Bolte et al. [2014]). The code of K–SVD and the images are taken from SMALLbox framework (Sparse Representation and Dictionary Learning evaluation toolbox) (Damnjanovic et al. [2010]). The tested noise, white Gaussian noise with a standard deviation σ, has been added to each image. The code of PALM is written on Matlab based on the algorithm of NMF with some modifications.

We use the PSNR criterion (Peak signal–to–noise ratio) and MSE (mean squared error) to evaluate the obtained denoising results. The best denoising is achieved when PSNR is large, while MSE is small.

The PSNR is defined by:

$$PSNR = 10.\log\frac{MAX_I^2}{\sqrt{MSE}}. \tag{3.26}$$

Here, MAX_I is the maximum possible pixel value of the image, which is 255 for an 8–bits image.

Given an image I with size $M \times N$ and its noisy approximation J, the MSE is given by:

$$MSE = \frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}[I(i,j) - J(i,j)]^2. \tag{3.27}$$

From the image, we take 40.000 patches (which is also the size of training set), each patch has size of $\sqrt{n}\times\sqrt{n}$ pixels with an overlap of 1 pixel. In our tests, $\sqrt{n}$ has the integer values from 5 to 15. The matrix dictionary D has size of $n \times 256$, and the initial of D is chosen as an overcomplete DCT dictionary (Discrete Cosine Transforms).

All algorithms were implemented in the Matlab R2007a, and performed on a PC Intel i5 CPU650, 3.2 GHz of 4GB RAM.

We stop DCA-based algorithm with the tolerance $\epsilon = 10^{-4}$. The parameter λ is chosen as follows: $\lambda \in [600, \ldots, 800]$.

Experiments

In our experiments, the cases below will be considered:

1. Comparing the performance of DCA, K–SVD and PALM on 55 images with the same tested noise levels.
2. The efficiency of DCA with different values of λ.
3. The efficiency of DCA with different sizes of patches on different types of images.

The Table 3.1 shows the results of DCA, K–SVD and PALM when the tested noise σ is set to 20.

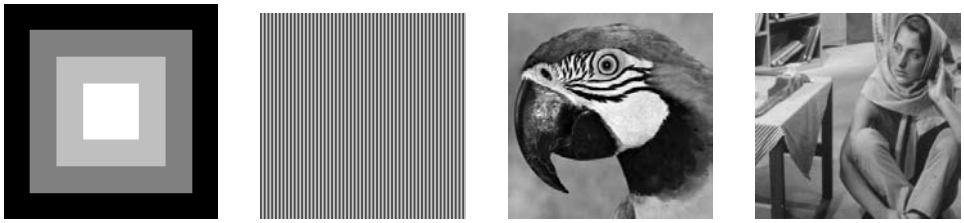
In the Table 3.1, the second column indicates the PSNR values of observed images (noisy images). The third, the forth and the last column correspond to the PSNR values of denoised images by K–SVD, DCA and PALM algorithm respectively.

From the numerical results, we observed that our algorithm is better than K–SVD on 54/55 images and both outperform PALM. These results are encouraging, which suggest that we can use DC programming and DCA for dictionary learning problem in other domains of image analysis: compress sensing, image denoising, etc. The figures from 3.4 to 3.23 present the denoised image results by K–SVD, DCA and PALM of 20/55 test images.

The second test case concerns the effect of the parameter λ. In sparse coding phase, the larger λ is, the smaller ℓ_0-norm will be. When λ is large enough, in our experience, $\lambda \geq 1000$, the vectors w_{ij} have the

Table 3.1: The denoising image and PSNR (dB) of DCA v.s K-SVD and PALM

Img	**PSNR noisy image**	**PSNR–KSVD**	**PSNR–DCA**	**PSNR–PALM**
1	22.10	30.84	**30.92**	30.01
2	22.13	30.17	**30.43**	29.82
3	22.12	30.69	**30.94**	30.39
4	22.11	30.22	**30.56**	29.82
5	22.12	30.61	**30.92**	30.22
6	22.12	29.24	**29.43**	28.95
7	22.10	27.80	**27.85**	27.10
8	22.10	30.11	**30.42**	29.84
9	22.11	28.57	**28.83**	28.41
10	22.11	32.14	**32.22**	31.80
11	22.10	**28.74**	28.60	27.80
12	22.10	29.18	**29.38**	28.95
13	22.10	30.20	**30.35**	29.97
14	22.10	25.81	**25.99**	25.55
15	22.09	27.30	**27.52**	27.15
16	22.11	25.65	**25.70**	25.52
17	22.10	27.29	**27.54**	27.07
18	22.11	26.30	**26.45**	26.12
19	22.10	29.14	**29.34**	28.80
20	22.10	27.77	**28.05**	27.54
21	22.11	29.91	**30.14**	29.61
22	22.09	28.21	**28.33**	28.05
23	22.14	33.52	**33.69**	33.05
24	22.12	31.05	**31.34**	30.77
25	22.11	33.25	**33.45**	32.63
26	22.11	31.03	**31.33**	30.77
27	22.11	32.97	**33.14**	32.78
28	22.11	30.91	**31.16**	30.69
29	22.11	26.19	**26.33**	26.07
30	22.11	28.68	**28.96**	28.46
31	22.12	25.18	**25.23**	25.03
32	22.11	27.92	**28.11**	27.77
33	22.11	29.05	**29.23**	28.81
34	22.12	27.98	**28.25**	27.73
35	22.11	30.38	**30.57**	30.02
36	22.10	28.46	**28.79**	28.16
37	22.11	28.35	**28.62**	28.08
38	22.12	30.46	**30.79**	30.21
39	22.09	32.08	**32.14**	31.83
40	22.10	32.01	**32.26**	31.66
41	22.11	30.02	**30.24**	29.62
42	22.10	29.92	**30.17**	29.41
43	22.10	30.80	**30.95**	30.27
44	22.13	23.58	**23.68**	23.54
45	22.10	29.84	**29.95**	29.55
46	22.09	27.02	**27.21**	26.87
47	22.09	26.34	**26.54**	26.22
48	22.11	31.21	**31.28**	30.64
49	22.08	33.18	**33.42**	33.06
50	22.13	32.36	**32.55**	32.14
51	22.12	32.22	**32.39**	32.07
52	22.10	39.33	**40.20**	39.73
53	22.11	33.25	**33.59**	33.09
54	22.10	30.97	**31.01**	30.51
55	22.09	29.48	**29.69**	28.97



(a) Box (b) Stripes (c) Parrot (d) Barbara

Figure 3.1: Sample from the tested images.

sparsity in the range of 90%. Figure 3.2 presents the denoising results on the images "Box", "Stripes" and "Parrot". Three considered images have different levels of "simple". (Simple here means that the image has the regions flat, less the detail areas.) For the simple images, the greater λ will give the better solutions. In practice, we choose λ in interval [1300, 1600].

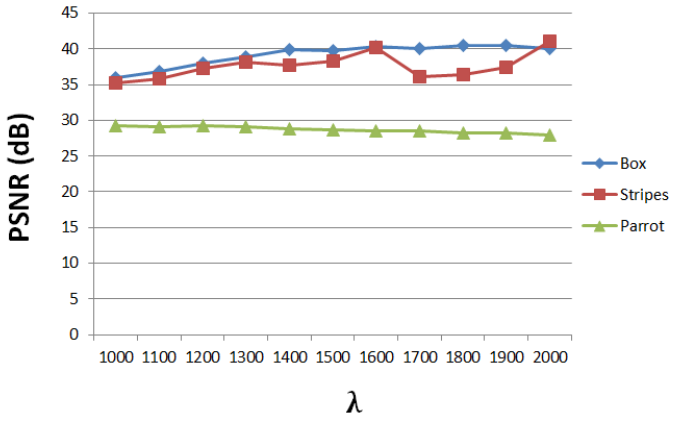


Figure 3.2: The performance of DCA with different values of the parameters λ.

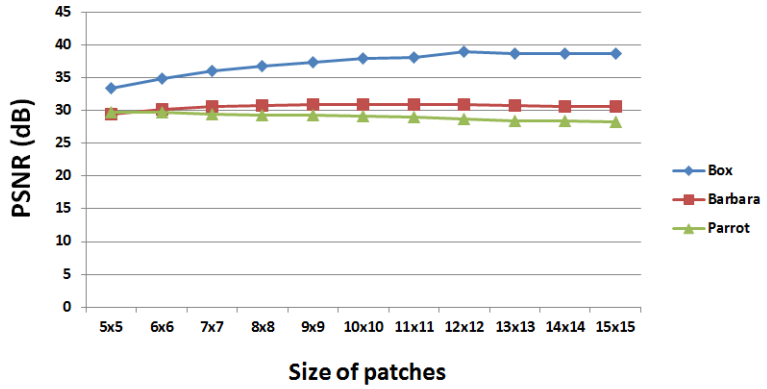


Figure 3.3: The dependence of the performance of DCA with different sizes of patches and different types of images.

In the last case, we tested with different sizes of patches on different types of image. In all experiments, the

results showed that the small patches are better for recovering the type of images which have more details, complex areas; whereas the large patches are better for the type of images which have more flat areas. For example, the image "Box" can be denoised better when the patch size increases, but the image "Parrot" and "Barbara" have the better results when the patch size decreases. It can be seen in Figure 3.3.

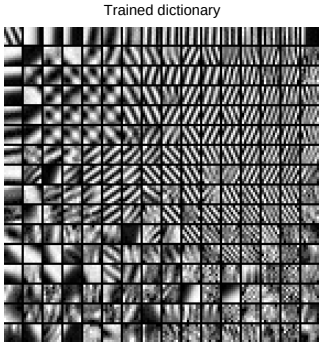
3.5 Conclusion

In this chapter, we have studied the DC programming and DCA for dictionary learning problem and applied it on the denoising image problem. The dictionary learning processes have been done with the training set including the overlapping patches taken from the noisy images. The efficiency of DCA has been approved experimentally on the set of gray images and the results show that our method is encourage. The experiments also suggested the way improve the result of denoising when working with the different type of images. The success on denoising image encourage us to apply DC and DCA on different problems on image processing and image analysis.

(a) Original image

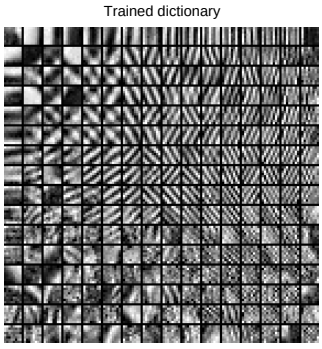
(b) Noisy image, PSNR = 22.11 dB

(c) Denoised image by KSVD, PSNR = 30.84 dB



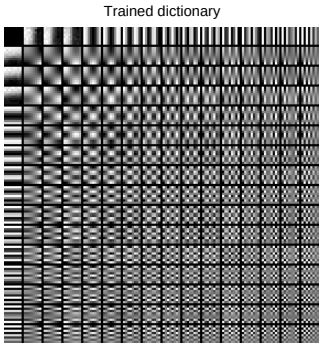
(d) Trained Dictionary by KSVD

(e) Denoised image by DCA, PSNR = 31.06 dB



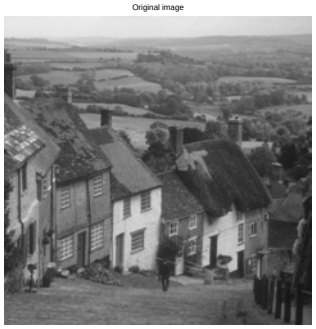
(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 30.01 dB

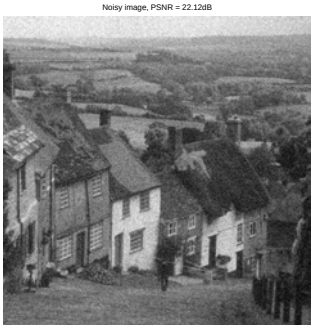


(h) Trained Dictionary by PALM

Figure 3.4: Image *1*.

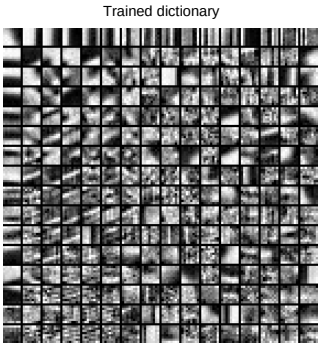


(a) Original image



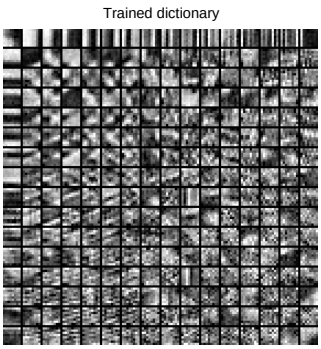
(b) Noisy image, PSNR = 22.12 dB

(c) Denoised image by KSVD, PSNR = 30.17 dB



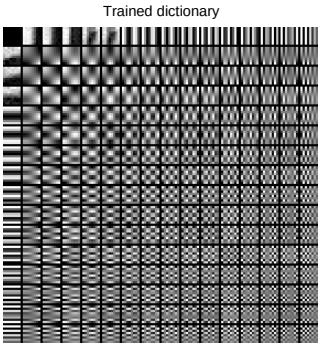
(d) Trained Dictionary by KSVD

(e) Denoised image by DCA, PSNR = 30.44 dB



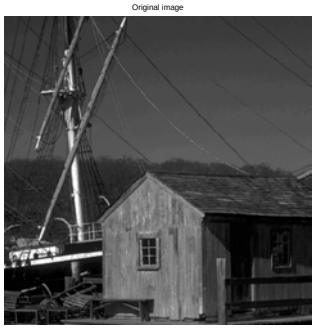
(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 29.82 dB

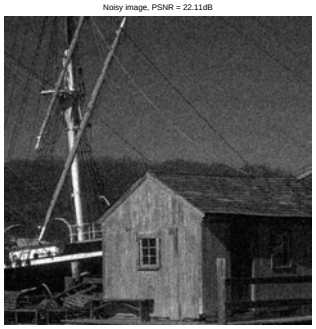


(h) Trained Dictionary by PALM

Figure 3.5: Image *2*.

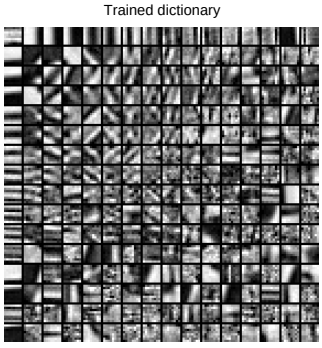


(a) Original image



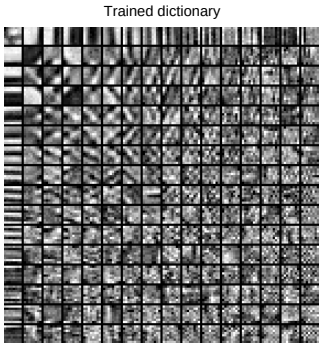
(b) Noisy image, PSNR = 22.11 dB

(c) Denoised image by KSVD, PSNR = 30.69 dB



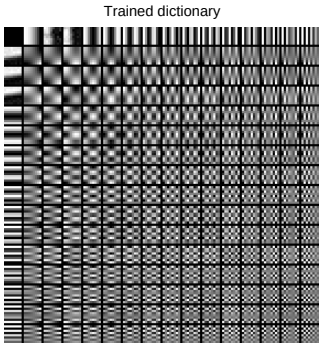
(d) Trained Dictionary by KSVD

(e) Denoised image by DCA, PSNR = 30.92 dB



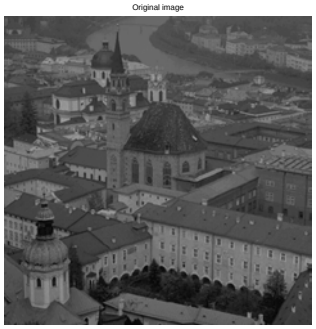
(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 30.39 dB

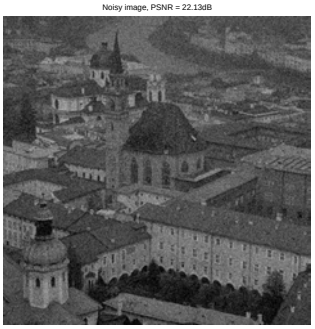


(h) Trained Dictionary by PALM

Figure 3.6: Image *3*.

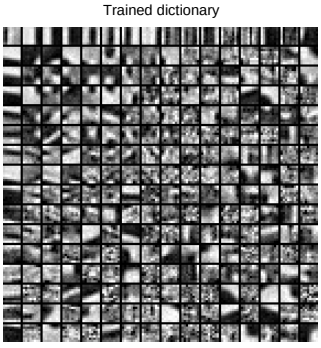


(a) Original image

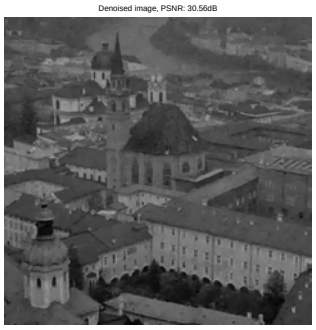


(b) Noisy image, PSNR = 22.13 dB

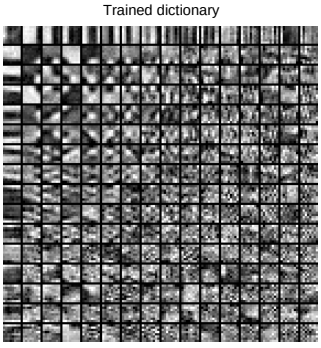
(c) Denoised image by KSVD, PSNR = 30.22 dB



(d) Trained Dictionary by KSVD

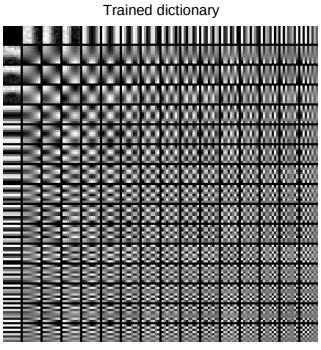


(e) Denoised image by DCA, PSNR = 30.56 dB



(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 29.82 dB



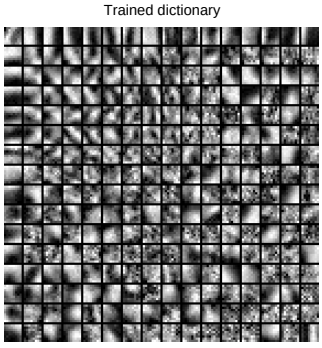
(h) Trained Dictionary by PALM

Figure 3.7: Image *4*.

(a) Original image

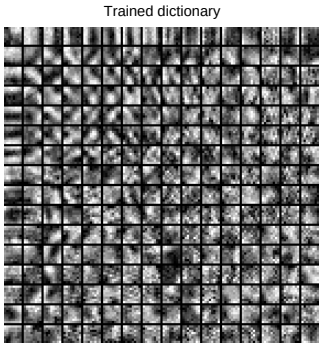
(b) Noisy image, PSNR = 22.12 dB

(c) Denoised image by KSVD, PSNR = 30.61 dB



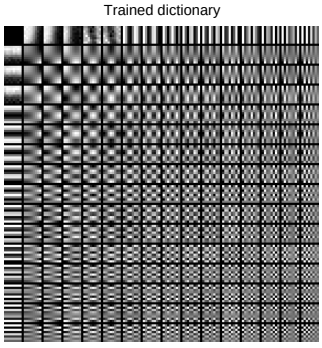
(d) Trained Dictionary by KSVD

(e) Denoised image by DCA, PSNR = 30.97 dB



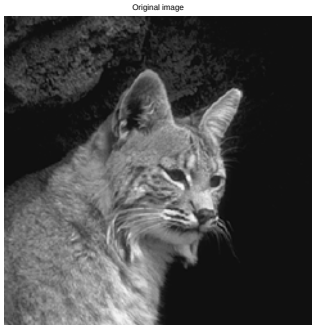
(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 30.22 dB



(h) Trained Dictionary by PALM

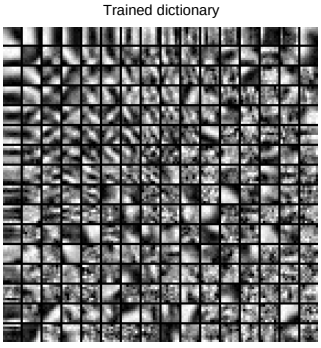
Figure 3.8: Image *5*.



(a) Original image

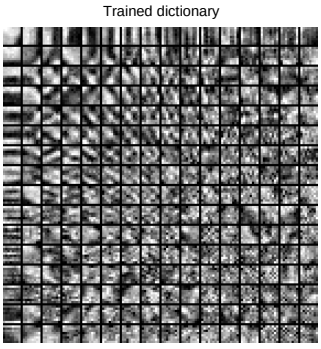
(b) Noisy image, PSNR = 22.12 dB

(c) Denoised image by KSVD, PSNR = 30.11 dB



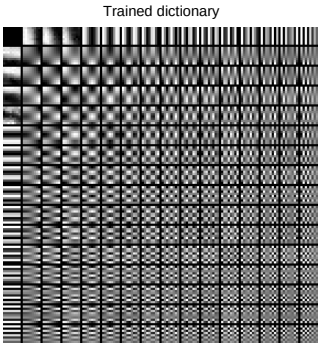
(d) Trained Dictionary by KSVD

(e) Denoised image by DCA, PSNR = 30.38 dB



(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 29.84 dB



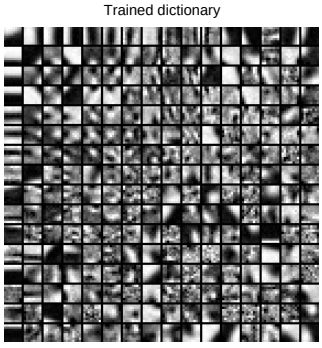
(h) Trained Dictionary by PALM

Figure 3.9: Image *6*.

(a) Original image

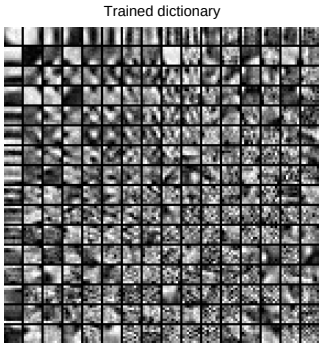
(b) Noisy image, PSNR = 22.12 dB

(c) Denoised image by KSVD, PSNR = 32.14 dB



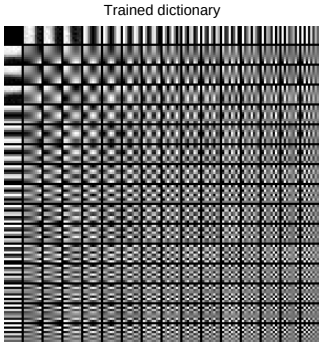
(d) Trained Dictionary by KSVD

(e) Denoised image by DCA, PSNR = 32.29 dB



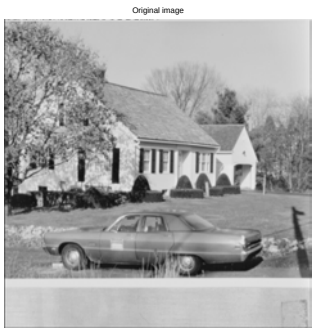
(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 31.80 dB

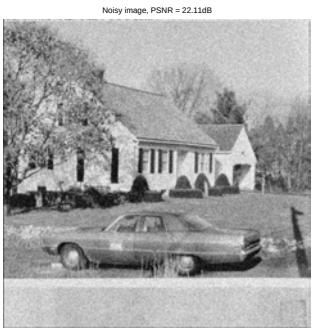


(h) Trained Dictionary by PALM

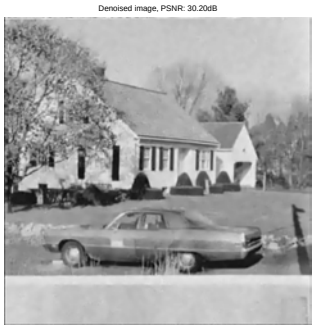
Figure 3.10: Image 7.



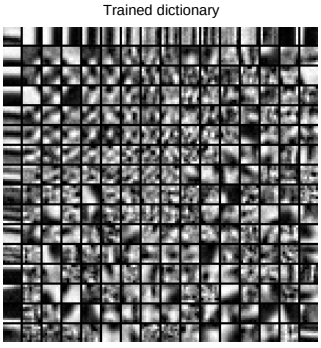
(a) Original image



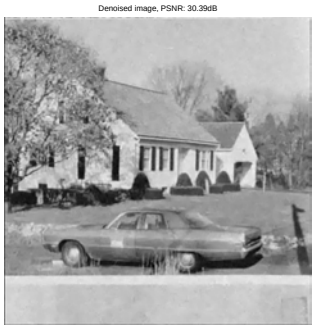
(b) Noisy image, PSNR = 22.11dB



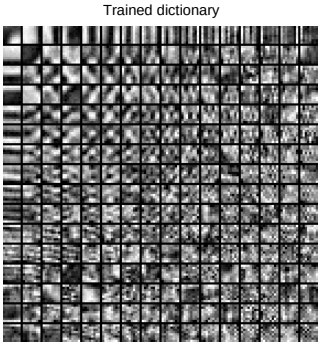
(c) Denoised image by KSVD, PSNR = 30.20dB



(d) Trained Dictionary by KSVD

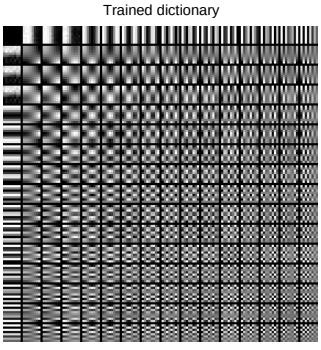


(e) Denoised image by DCA, PSNR = 30.39dB



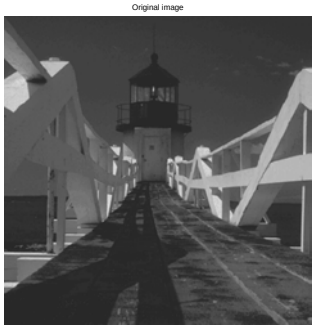
(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 29.97 dB

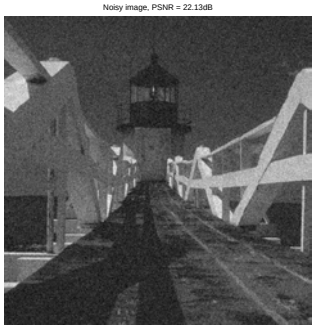


(h) Trained Dictionary by PALM

Figure 3.11: Image *8*.

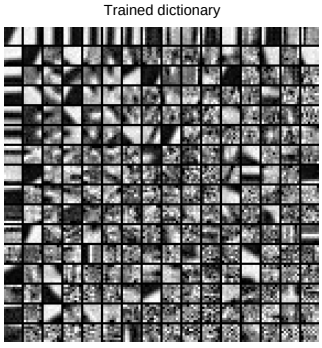


(a) Original image



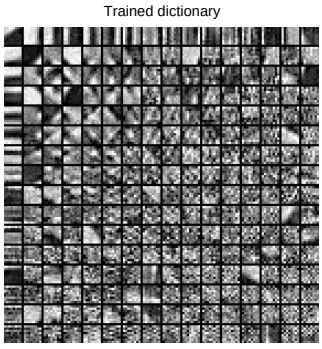
(b) Noisy image, PSNR = 22.13dB

(c) Denoised image by KSVD, PSNR = 33.52dB



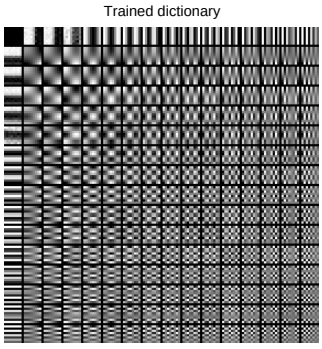
(d) Trained Dictionary by KSVD

(e) Denoised image by DCA, PSNR = 33.77dB



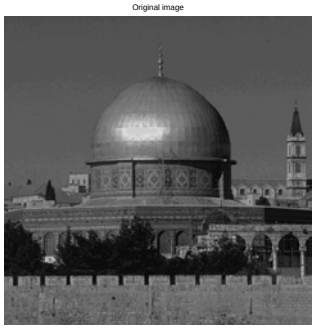
(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 33.05 dB

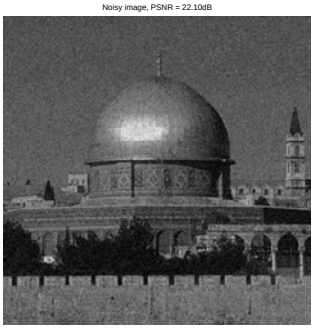


(h) Trained Dictionary by PALM

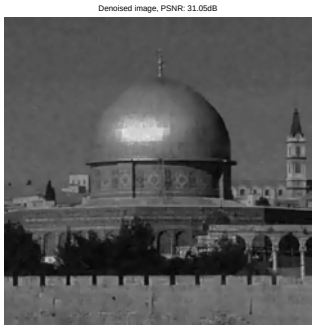
Figure 3.12: Image *9*.



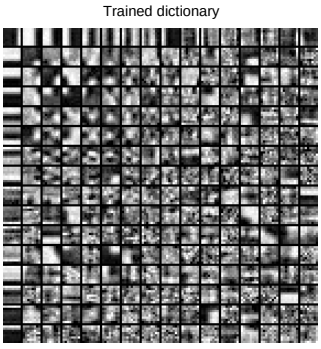
(a) Original image



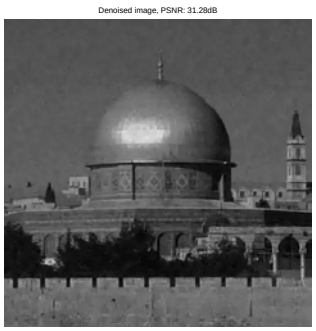
(b) Noisy image, PSNR = 22.10dB



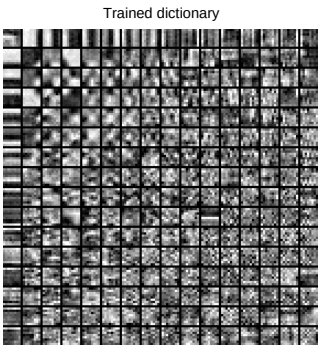
(c) Denoised image by KSVD, PSNR = 31.05dB



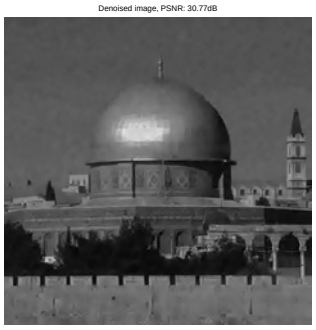
(d) Trained Dictionary by KSVD



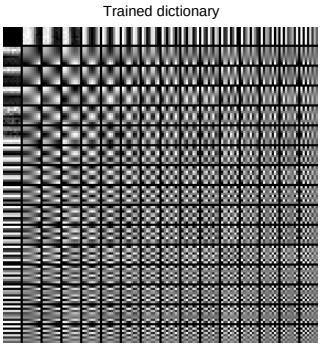
(e) Denoised image by DCA, PSNR = 31.28dB



(f) Trained Dictionary by DCA

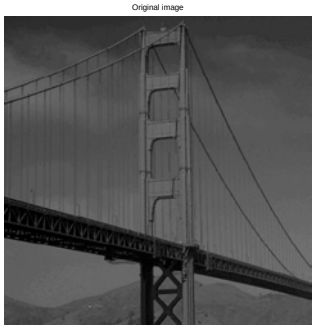


(g) Denoised image by PALM, PSNR = 30.77 dB

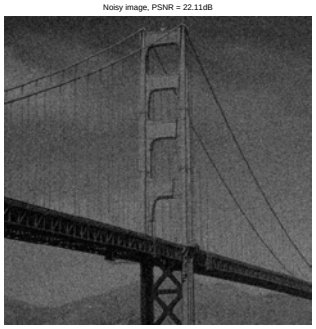


(h) Trained Dictionary by PALM

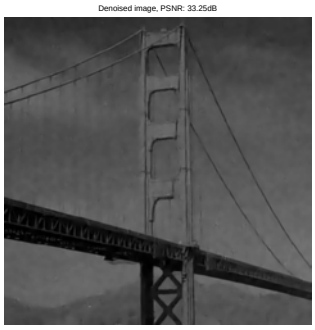
Figure 3.13: Image *10*.



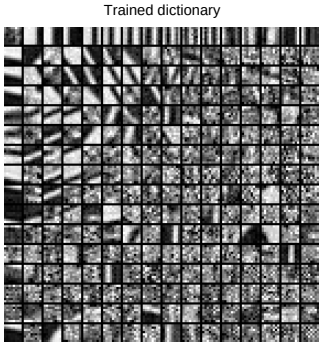
(a) Original image



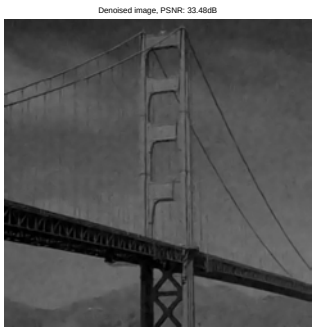
(b) Noisy image, PSNR = 22.11dB



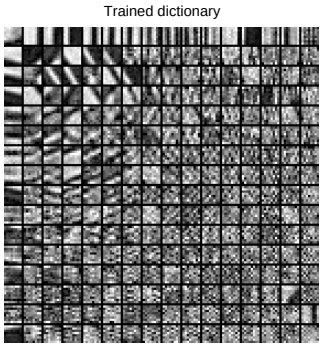
(c) Denoised image by KSVD, PSNR = 33.25dB



(d) Trained Dictionary by KSVD

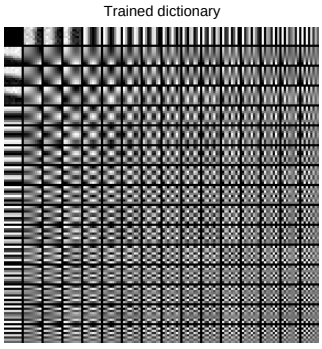


(e) Denoised image by DCA, PSNR = 33.48dB



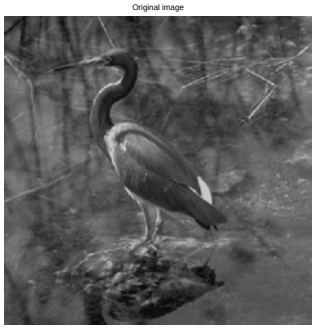
(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 32.63 dB

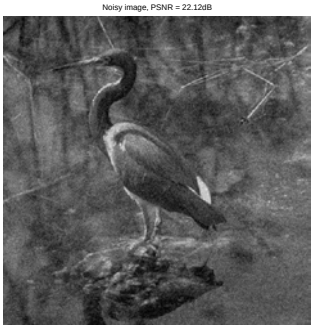


(h) Trained Dictionary by PALM

Figure 3.14: Image *11*.

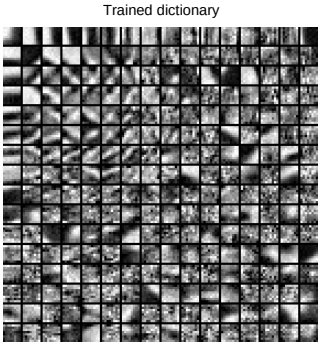


(a) Original image



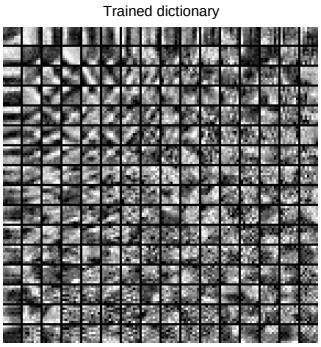
(b) Noisy image, PSNR = 22.12dB

(c) Denoised image by KSVD, PSNR = 31.03dB



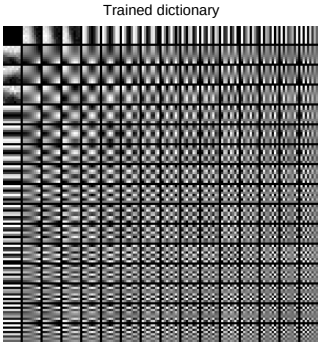
(d) Trained Dictionary by KSVD

(e) Denoised image by DCA, PSNR = 31.40dB



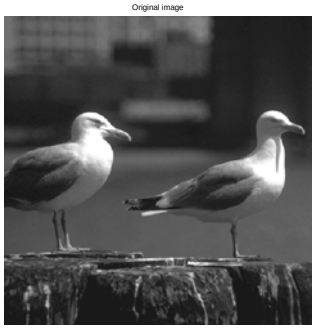
(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 30.77 dB

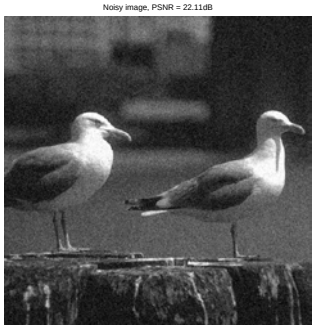


(h) Trained Dictionary by PALM

Figure 3.15: Image *12*.

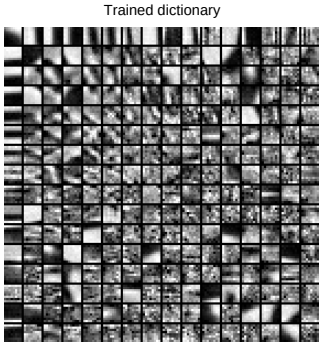


(a) Original image



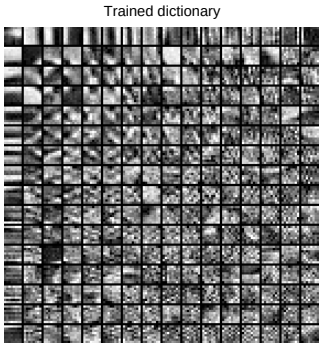
(b) Noisy image, PSNR = 22.11dB

(c) Denoised image by KSVD, PSNR = 32.97dB



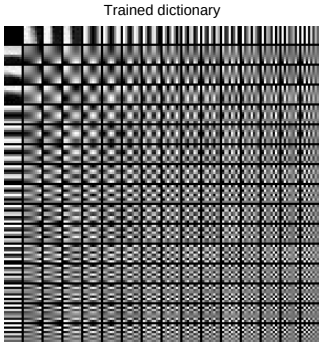
(d) Trained Dictionary by KSVD

(e) Denoised image by DCA, PSNR = 33.02dB



(f) Trained Dictionary by DCA

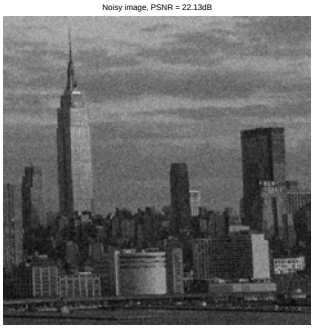
(g) Denoised image by PALM, PSNR = 32.78 dB



(h) Trained Dictionary by PALM

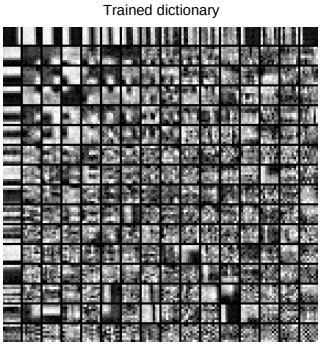
Figure 3.16: Image *13*.

(a) Original image

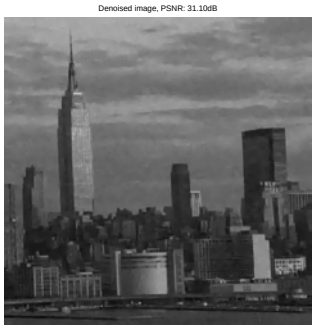


(b) Noisy image, PSNR = 22.13dB

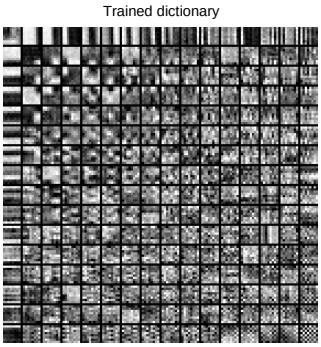
(c) Denoised image by KSVD, PSNR = 30.91dB



(d) Trained Dictionary by KSVD

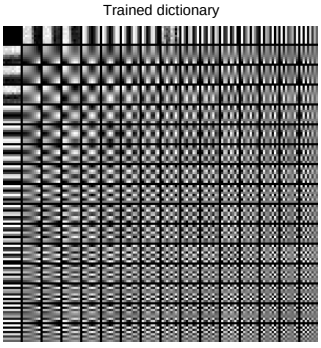


(e) Denoised image by DCA, PSNR = 31.10dB



(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 30.69 dB



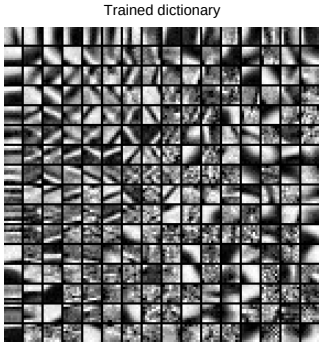
(h) Trained Dictionary by PALM

Figure 3.17: Image *14*.

(a) Original image

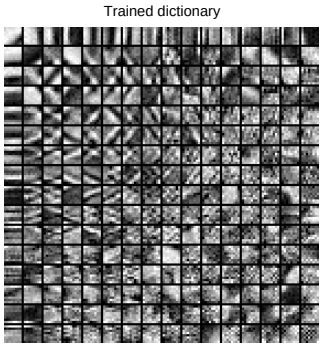
(b) Noisy image, PSNR = 22.10dB

(c) Denoised image by KSVD, PSNR = 30.38dB



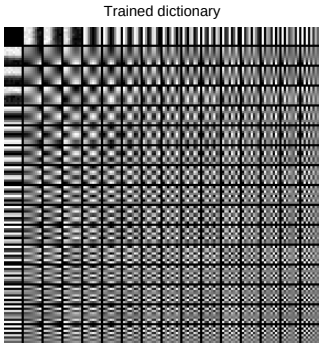
(d) Trained Dictionary by KSVD

(e) Denoised image by DCA, PSNR = 30.45dB



(f) Trained Dictionary by DCA

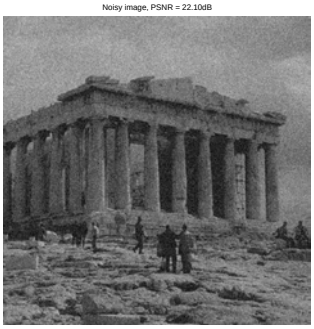
(g) Denoised image by PALM, PSNR = 30.02 dB



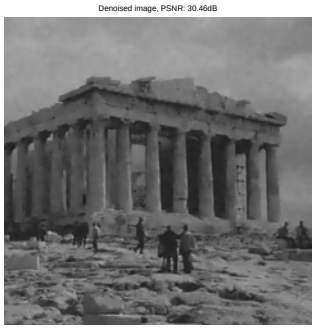
(h) Trained Dictionary by PALM

Figure 3.18: Image *15*.

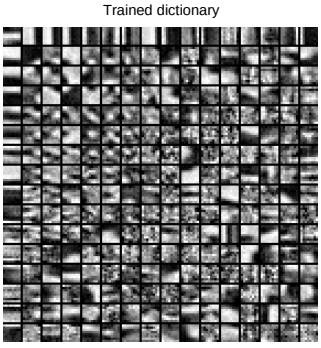
(a) Original image



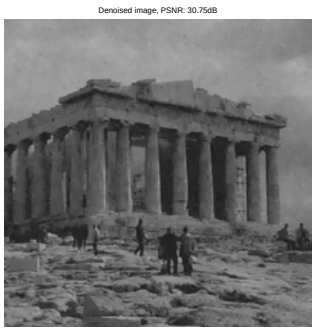
(b) Noisy image, PSNR = 22.10dB



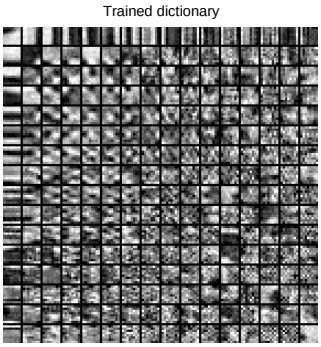
(c) Denoised image by KSVD, PSNR = 30.46dB



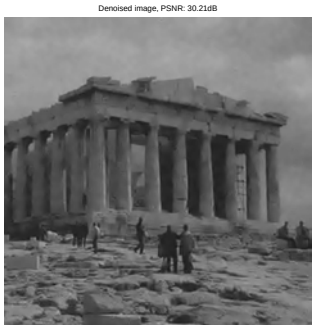
(d) Trained Dictionary by KSVD



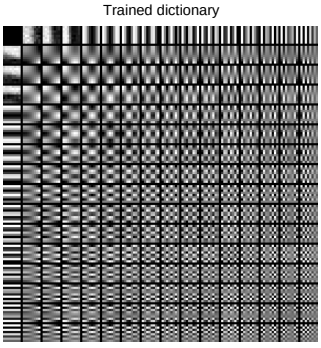
(e) Denoised image by DCA, PSNR = 30.75dB



(f) Trained Dictionary by DCA

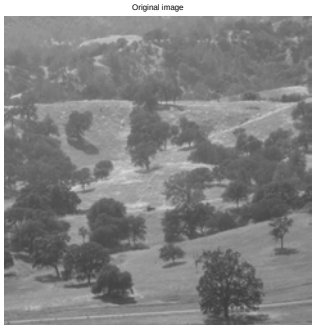


(g) Denoised image by PALM, PSNR = 30.21 dB

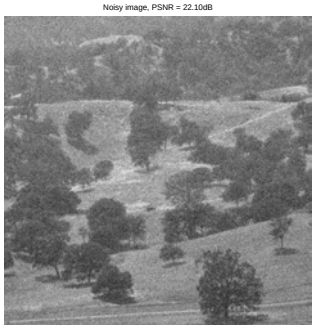


(h) Trained Dictionary by PALM

Figure 3.19: Image *16*.

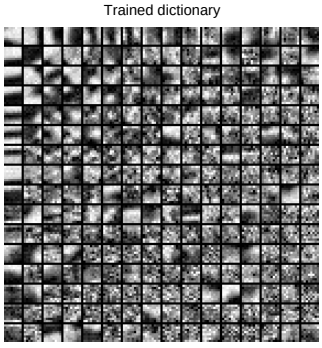


(a) Original image



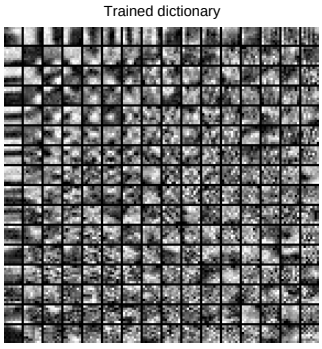
(b) Noisy image, PSNR = 22.10dB

(c) Denoised image by KSVD, PSNR = 32.01dB



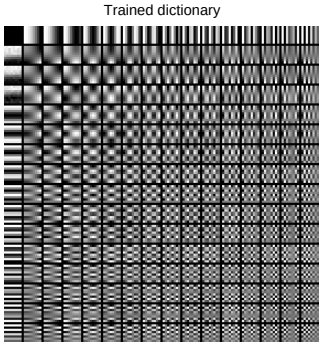
(d) Trained Dictionary by KSVD

(e) Denoised image by DCA, PSNR = 32.38dB



(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 31.66 dB



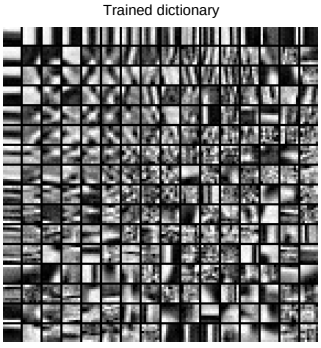
(h) Trained Dictionary by PALM

Figure 3.20: Image *17*.

(a) Original image

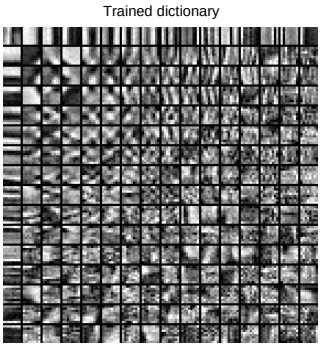
(b) Noisy image, PSNR = 22.10dB

(c) Denoised image by KSVD, PSNR = 30.02dB



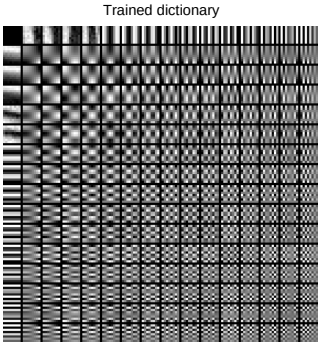
(d) Trained Dictionary by KSVD

(e) Denoised image by DCA, PSNR = 30.21dB



(f) Trained Dictionary by DCA

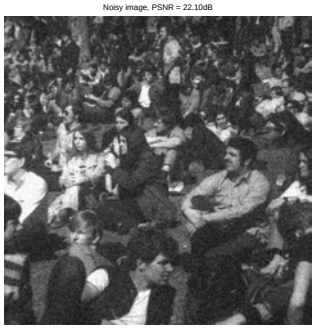
(g) Denoised image by PALM, PSNR = 29.62 dB



(h) Trained Dictionary by PALM

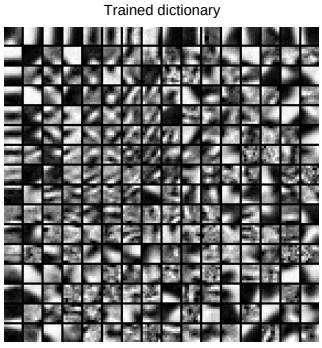
Figure 3.21: Image *18*.

(a) Original image



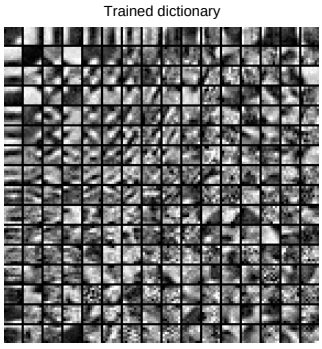
(b) Noisy image, PSNR = 22.10dB

(c) Denoised image by KSVD, PSNR = 30.80dB



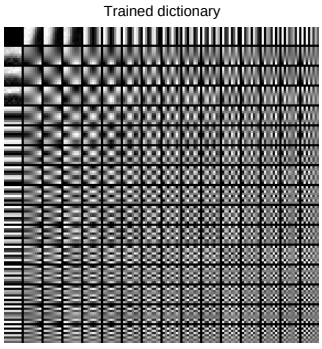
(d) Trained Dictionary by KSVD

(e) Denoised image by DCA, PSNR = 30.99dB



(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 30.27 dB

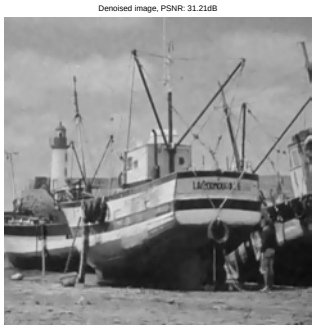


(h) Trained Dictionary by PALM

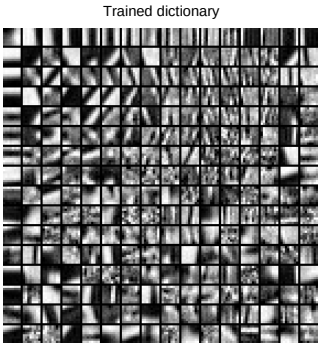
Figure 3.22: Image *19*.

(a) Original image

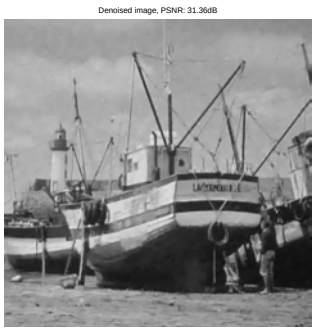
(b) Noisy image, PSNR = 22.13dB



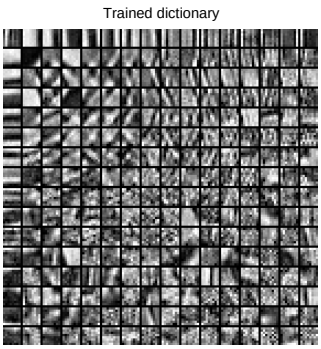
(c) Denoised image by KSVD, PSNR = 31.21dB



(d) Trained Dictionary by KSVD

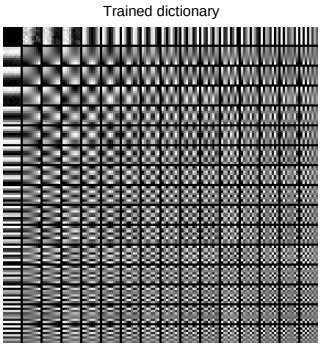


(e) Denoised image by DCA, PSNR = 31.36dB



(f) Trained Dictionary by DCA

(g) Denoised image by PALM, PSNR = 30.64 dB



(h) Trained Dictionary by PALM

Figure 3.23: Image *20*.

Chapter 4

Image segmentation and application to automated cell counting

4.1 Image segmentation

4.1.1 Introduction

Image segmentation is a significant processing step in many image, video and computer vision applications. It is a critical step towards content analysis and image understanding. The aim of image segmentation is to partition an image into a set non–overlapped, consistent regions with respect to some characteristics such as colors/gray values or textures.

Image segmentation is an important research field and many segmentation methods have been proposed in the literature. For a more complete review on image segmentation methods, the reader is referred to (Haralick and Shapiro [1985], Pal and Pal [1993], Skarbek and Koschan [1994b], Verge Llahi [2005]) and the references therein.

We can classify image segmentation methods into four categories (Verge Llahi [2005]): methods based on pixels, on areas, on contours and on physical model for image formation.

Pixel based methods, which consist in regrouping in different regions the pixels contained in an image, are the simplest approaches for image segmentation. There exists some algorithms for these methods, for example: thresholding, K–Means, FCM (Bezdek [1981], Skarbek and Koschan [1994a], Siang Tan and Mat Isa [2011],...) and so on.

1. A part of this chapter is published under the titles:
[1]. Le Hoai Minh, Nguyen Thi Bich Thuy, Ta Minh Thuy, Le Thi Hoai An, *Image Segmentation via Feature Weighted Fuzzy Clustering by a DCA based algorithm*, Advanced Computational Methods for Knowledge Engineering, Studies in Computational Intelligence. Volume 479, Springer, ISSN: 1860–949X (Print) 1860–9503 (Online), pp. 53–63 (2013). The 1-*th* International Conference on Computer Science, Applied Mathematics and Applications (ICCSAMA 2013), Warsaw, Poland, 9–10 May, 2013.
[2]. Le Thi Hoai An, Le Minh Tam, Nguyen Thi Bich Thuy, *A novel approach to automated cell counting based on a Difference of Convex functions Algorithm (DCA)*, Computational Collective Intelligence. Technologies and Applications, Lecture Notes in Computer Science Volume 8083, pp 336–345, 2013. The 5-*th* International Conference on Computational Collective Intelligence, Technologies and Applications (ICCCI 2013), Craiova, Romania, 11–13 September, 2013.

Areas based methods work on the assumption that the pixels within one area are similar. First, the pixel will compare with its neighborhoods, if it is similar it belong to the cluster of its neighborhoods. The algorithms in these methods used to Region–growing (Seeded, Unseeded region growing) (Taylor and Lewis [1992], Brand [1993],...); Graph theoretical or Split & Merge techniques (Markov Random Field) (Huang et al. [1992], Cheng et al. [2002], Panjwani and Healey [1993],...).

Another approach is based on detecting the contour of image. This approach can be divided into two categories: Local techniques (Chapron [1992], Cumani [1989],...) and Global techniques (Geman and Geman [1984], Li [2009],...).

Segmentation based on physical models perform the partition via the study of the process of light reflection and image formation (Healey [1992], Campadelli et al. [1997], Wolff [1994]).

In our work, we focused on pixel based methods. There are three main classes of algorithms in this domain:
- Histogram-based technique: firstly, a histogram is computed from all of the pixels in the image. Then, image pixels are classified as belonging to one of those classes thus formed by using the peaks and valleys in the histogram.
- Clustering techniques: pixels are grouped, using a hard clustering method, by means of their color values/textures.
- Fuzzy clustering techniques: instead of using hard clustering, fuzzy clustering is used for pixel classification task. A popular choice is the Fuzzy C-Means algorithm (Bezdek [1981]).

Fuzzy C-Means (FCM) clustering, introduced by Bezdek in 1981 (Bezdek [1981]), is most widely used fuzzy clustering method. The FCM problem is formulated as a non convex optimization problem for which only heuristic algorithms are available before the work of Le Thi et al. in 2007 (Le Thi et al. [2007c]). In this work, the authors reformulated FCM model as DC (Difference of Convex function) programs and then developed three DCA (DC Algorithm) schemes to solve the three resulting DC programs. The numerical results on several real data sets show that the proposed DCA is an efficient approach for fuzzy clustering in large data sets of high dimension and it is superior to the FCM algorithm in both running-time and quality of solutions. Later, in Le Thi et al. [2008a], the authors have successfully applied the DCA based algorithm for FCM in noisy image segmentation problems.

Another hand, usually in classification, the distance measure involves all attributes of the data set. It is applicable if most attributes are important to every cluster. However, the performance of clustering algorithms can be significantly degraded if many irrelevant attributes are used. In the literature, various approaches have been proposed to address this problem. The first strategy is feature selection that finds irrelevant features and removes them from the feature set before constructing a classifier. Feature weighting is an extension of the feature selection where the features are assigned continuous weights.

Relevant features correspond to high weight values, whereas weight values close to zero represent irrelevant features. Clustering using weighted dissimilarity measures attracts more and more attention (see Chan et al. [2004], Frigui and Nasui [2004]). In Frigui and Nasui [2004], the authors investigated the FCM problem using weighted features for image segmentation.

The problem FCM using features weighting can be stated as follows. Let $\mathcal{X} := \{x_1, x_2, ..., x_n\}$ be a data set of n entities with m attributes and the known number of clusters k ($2 \leq k \leq n$). Denote by Λ a $k \times m$ matrix defined as $\Lambda = (\lambda_{l,i})$ where $\lambda_{l,i}$ defines the relevance of i-th feature to the cluster C_l. $W = (w_{j,l}) \in \mathbb{R}^{n \times k}$ with $j = 1, \ldots, n$ and $l = 1, \ldots, k$ called the *fuzzy partition* matrix in which each element $w_{j,l}$ indicates the membership degree of each point x_j in the cluster C_l (the probability that a point x_j belongs to the cluster C_l). $Z = (z_{l,i}) \in \mathbb{R}^{n \times k}$ with $l = 1, \ldots, k$ and $i = 1, \ldots, m$ present k centres of k clusters.

The task is to regrouping the set $\mathcal{X}$ into k clusters in order to minimize the sum of squared distances from the entities to the centroid of their cluster. The dissimilarity measure is defined by m weighted attributes. Then a straightforward formulation of the clustering using weighted dissimilarity measures is (μ, β are exponents

greater than 1):

$$\begin{cases} \min F(W,Z,\Lambda) := \sum\limits_{l=1}^{k}\sum\limits_{j=1}^{n}\sum\limits_{i=1}^{m} w_{jl}^{\mu}\lambda_{li}^{\beta}(z_{li}-x_{ji})^2 \\ s.t: \sum\limits_{l=1}^{k} w_{jl} = 1, j = 1..n, \\ \quad \sum\limits_{i=1}^{m} \lambda_{li} = 1, l = 1..k, \\ \quad w_{jl} \in [0,1], j = 1..n, l = 1..k, \\ \quad \lambda_{li} \in [0,1], l = 1..k, i = 1..m. \end{cases} \tag{4.1}$$

Problem (4.1) is difficult due to the nonconvexity of the objective function. Moreover, in real applications this is a very large scale problem (high dimension and large data set, i.e. n and m are very large), that is why global optimization approaches such as Branch & Bound, Cutting plane algorithms etc. cannot be used. In Frigui and Nasui [2004], the authors proposed a FCM type algorithm, called **SCAD** (Simultaneous Clustering and Attribute Discrimination), to solve the problem (4.1). At first, **SCAD** fixes Z, Λ and finds W to minimize $F(W,.,.)$. Then W, Λ are fixed for finding Z minimizing $F(.,Z,.)$. Finally, Λ is obtained by minimizing $F(.,.,\Lambda)$ with W and Z fixed. The process is repeated until no more improvement in the objective function can be made.

For hard clustering, there exists a number of algorithms that work with features weighting. In Chan et al. [2004], the authors presented **WF–KM** algorithm. At first, **WF-KM** fixes two variables, and then finding the last variable to minimize the objective function $F(.,.,.)$. In Jing et al. [2007], the authors proposed a variance by adding the entropy of dimensions, namely $\gamma\sum_{j=1}^{m}\lambda_{l,i}\log\lambda_{l,i}$, to objective function. By modifying the objective function, the algorithm can avoid the problem of identifying clusters by few dimensions in sparse data. In another work Huang et al. [2005], the authors considered the problem where the matrix of weights Λ becomes a vector $\bar{\Lambda}$. More precisely, $\bar{\Lambda}_j$ defines the relevance of i-th feature to all cluster C_l $(l = 1..k)$.

We investigate in this work, for solving the problem (4.1) that works with feature weighted variables $\lambda_{l,i}$, an efficient nonconvex programming approach based on DC Programming and DCA. This work differs from previous work that uses the advantages of feature weighting, as well as the fuzyness and the power of DCA.

4.1.2 A DC formulation of the problem (4.1)

In the problem (4.1), the variables W and Λ are a priori bounded. One can also find a constraint for bounding the variable Z. Indeed, let $\alpha_i := \min_{j=1,\ldots,n} x_{j,i}$, $\gamma_i := \max_{j=1,\ldots,n} x_{j,i}$. Hence $z_l \in \mathcal{T}_l := \Pi_{i=1}^{m}[\alpha_i, \gamma_i]$ for all $l = 1, ..., k$, and $Z \in \mathcal{T} := \Pi_{l=1}^{k}\mathcal{T}_l$.

Let Δ_l (resp. $\mathcal{C}_j$) be the $(m-1)$-simplex in $\mathbb{R}^m$(resp. $(k-1)$-simplex in $\mathbb{R}^k$), for each $l \in \{1, ..., k\}$ (resp. for each $j \in \{1, ..., n\}$), defined by:

$$\Delta_l := \left\{ \Lambda_l := (\lambda_{l,i})_l \in [0,1]^m : \sum_{i=1}^{m} \lambda_{l,i} = 1 \right\};$$
$$\mathcal{C}_j := \left\{ W_j := (w_{j,l})_j \in [0,1]^k : \sum_{l=1}^{k} w_{j,l} = 1 \right\},$$

and $\mathcal{C} := \Pi_{j=1}^{n}\mathcal{C}_j, \mathcal{T} := \Pi_{l=1}^{k}\mathcal{T}_l, \Delta := \Pi_{l=1}^{k}\Delta_l$.

The problem (4.1) can be rewritten as:

$$\min\{F(W,Z,\Lambda) : (W,Z,\Lambda) \in (\mathcal{C}\times\mathcal{T}\times\Delta)\}. \tag{4.2}$$

Our DC decomposition of F is based on the following result.

Proposition 4.1 *There exists $\rho > 0$ such that the function*

$$h(u,v,y) := \frac{\rho}{2}\left(u^2+v^2+y^2\right) - u^{\mu}y^{\beta}(v-a)^2$$

is convex on $(u,v,y) \in [0,1]\times[\alpha,\gamma]\times[0,1]$.

Proof: Let us consider the function $f : \mathbb{R}\times\mathbb{R}\to\mathbb{R}$ defined by:

$$f(u,v,y) = u^{\mu}y^{\beta}(v-a)^2. \tag{4.3}$$

The Hessian of f, denoted $J(u,v,y)$, is given by:

$$J(u,v,y) = \begin{pmatrix} \mu(\mu-1)u^{\mu-2}y^{\beta}(v-a)^2 & 2\mu u^{\mu-1}y^{\beta}(v-a) & \mu u^{\mu-1}\beta y^{\beta-1}(v-a)^2 \\ 2\mu u^{\mu-1}y^{\beta}(v-a) & 2u^{\mu}y^{\beta} & 2\beta u^{\mu}y^{\beta-1}(v-a) \\ \beta y^{\beta-1}\mu u^{\mu-1}(v-a)^2 & 2\beta u^{\mu}y^{\beta-1}(v-a) & \beta(\beta-1)u^{\mu}y^{\beta-2}(v-a)^2 \end{pmatrix}. \tag{4.4}$$

For all $(u,v,y)\in[0,1]\times[\alpha,\gamma]\times[0,1]$, the ℓ_1-norm of $J(u,v,y)$ ($||J(u,v,y)||_1$) is computed as follows:

$$\max\Big\{ \begin{array}{l} |\mu(\mu-1)u^{\mu-2}y^{\beta}(v-a)^2| + |2\mu u^{\mu-1}y^{\beta}(v-a)| + |\mu u^{\mu-1}\beta y^{\beta-1}(v-a)^2|; \\ |2\mu u^{\mu-1}y^{\beta}(v-a)| + |2u^{\mu}y^{\beta}| + |2\beta u^{\mu}y^{\beta-1}(v-a)|; \\ |\beta y^{\beta-1}\mu u^{\mu-1}(v-a)^2| + |2\beta u^{\mu}y^{\beta-1}(v-a)| + |\beta(\beta-1)u^{\mu}y^{\beta-2}(v-a)^2|\Big\}. \end{array} \tag{4.5}$$

For all $(u,v,y) : u\in[0,1], v\in[\alpha,\gamma], y\in[0,1], \mu\geq 2, \beta\geq 2$ we have

$$\begin{aligned} ||J(u,v,y)||_1 \leq \rho := \max\{&\mu(\mu-1)\delta^2 + 2\mu\delta + \beta\mu\delta^2; 2\mu\delta + 2 + 2\beta\delta; \\ &\beta\mu\delta^2 + 2\beta\delta + \beta(\beta-1)\delta^2\}, \end{aligned} \tag{4.6}$$

where $\delta = \gamma - \alpha$.

It leads to $\rho - ||J(u,v,y)||_1 \geq 0$. As a consequence, $\nabla^2 h(u,v,y)$ is a positive semidefine matrix, where $h(u,v,y)$ is defined as:

$$h(u,v,y) = \frac{\rho}{2}\left(u^2+v^2+y^2\right) - u^{\mu}y^{\beta}(v-a)^2. \tag{4.7}$$

Hence, $h(u,v,y)$ is convex on $\{u\in[0,1], v\in[\alpha,\gamma], y\in[0,1]\}$ with ρ as large as in (4.6). ■

Using the above proposition, for $u\leftarrow w_{jl}, v\leftarrow z_{li}, y\leftarrow\lambda_{li}$, the function

$$h_{lij}(w_{jl}, z_{li}, \lambda_{li}) = \frac{\rho}{2}\left(w_{jl}^2 + z_{li}^2 + \lambda_{li}^2\right) - w_{jl}^{\mu}\lambda_{li}^{\beta}(z_{li}-x_{ji})^2 \tag{4.8}$$

is convex on $([0,1]\times[\alpha_i,\gamma_i]\times[0,1]\}$.

As a consequence, the function $H(W,Z,\Lambda)$ defined by:

$$H(W,Z,\Lambda) := \sum_{l=1}^{k}\sum_{j=1}^{n}\sum_{i=1}^{m}\left[\frac{\rho}{2}\left(w_{jl}^2 + z_{li}^2 + \lambda_{li}^2\right) - w_{jl}^{\mu}\lambda_{li}^{\beta}(z_{li}-x_{ji})^2\right] \tag{4.9}$$

is convex on $(\mathcal{C}\times\mathcal{T}\times\Delta)$.

Finally, we can express F as follows:

$$F(W,Z,\Lambda) := G(W,Z,\Lambda) - H(W,Z,\Lambda), \tag{4.10}$$

where

$$G(W,Z,\Lambda) := \frac{\rho}{2}\sum_{l=1}^{k}\sum_{j=1}^{n}\sum_{i=1}^{m}\left(w_{jl}^2 + z_{li}^2 + \lambda_{li}^2\right);$$

and $H(W,Z,\Lambda)$ as (4.9) are clearly convex functions. Therefore, we get the following DC formulation of (4.2):

$$\min\left\{F(W,Z,\Lambda) := G(W,Z,\Lambda) - H(W,Z,\Lambda) : (W,Z,\Lambda) \in (\mathcal{C}\times\mathcal{T}\times\Delta)\right\}. \tag{4.11}$$

4.1.3 DCA applied to (4.11)

For designing a DCA applied to (4.11), we first need to compute $(\bar{W}^r, \bar{Z}^r, \bar{\Lambda}^r) \in \partial H(W^r, Z^r, \Lambda^r)$ and then solve the convex program

$$\min\left\{\frac{\rho}{2}\sum_{l=1}^{k}\sum_{j=1}^{n}\sum_{i=1}^{m}\left(w_{jl}^2 + z_{li}^2 + \lambda_{li}^2\right) - \langle(W,Z,\Lambda),(\bar{W}^r,\bar{Z}^r,\bar{\Lambda}^r)\rangle : (W,Z,\Lambda) \in (\mathcal{C}\times\mathcal{T}\times\Delta)\right\}. \tag{4.12}$$

The function H is differentiable and its gradient at the point (W^r, Z^r, Λ^r) is given by:

$$\begin{aligned}
\bar{W}^r &= \nabla_W H(W,Z,\Lambda) = \left(m\rho w_{jl} - \sum_{i=1}^{m}\mu w_{jl}^{\mu-1}\lambda_{li}^{\beta}(z_{li}-x_{ji})^2\right)_{j=1..n}^{l=1..k},\\
\bar{Z}^r &= \nabla_Z H(W,Z,\Lambda) = \left(n\rho z_{li} - \sum_{j=1}^{n} 2w_{jl}^{\mu}\lambda_{li}^{\beta}(z_{li}-x_{ji})\right)_{l=1..k}^{i=1..m},\\
\bar{\Lambda}^r &= \nabla_\Lambda H(W,Z,\Lambda) = \left(n\rho\lambda_{li} - \sum_{j=1}^{n}\beta w_{jl}^{\mu}\lambda_{li}^{\beta-1}(z_{li}-x_{ji})^2\right)_{l=1..k}.
\end{aligned} \tag{4.13}$$

The solution of the auxiliary problem (4.12) is explicitly computed as (Proj stands for the projection)

$$\begin{aligned}
(W^{r+1})_j &= \mathrm{Proj}_{\mathcal{C}_j}\left(\tfrac{1}{m\rho}(\bar{W}^r)_j\right)\ \ j=1,...n;\\
(Z^{r+1})_{li} &= \mathrm{Proj}_{[\alpha_i,\gamma_i]}\left(\tfrac{1}{n\rho}(\bar{Z}^r)_{li}\right)\ \ l=1,..,k, i=1,...m;\\
(\Lambda^{r+1})_l &= \mathrm{Proj}_{\Delta_l}\left(\tfrac{1}{n\rho}(\bar{\Lambda}^r)_l\right)\ \ l=1,...k.
\end{aligned} \tag{4.14}$$

Finally, DCA applied on (4.11) can be described as follows.

DCA-SI: DCA applied to (4.11)

- **Initialization:** Choose W^0, Z^0 and Λ^0. Let $\epsilon > 0$ be sufficiently small, $r = 0$.
- **Repeat**
  - Compute $(\bar{W}^r, \bar{Z}^r, \bar{\Lambda}^r)$ via (4.13).
  - Compute $(W^{r+1}, Z^{r+1}, \Lambda^{r+1})$ via (4.14).
  - $r = r + 1$
- **Until** $\|(W^{r+1}, Z^{r+1}, \Lambda^{r+1}) - (W^r, Z^r, \Lambda^r)\| \le \epsilon$ or $|F(W^{r+1}, Z^{r+1}, \Lambda^{r+1}) - F(W^r, Z^r, \Lambda^r)| \le \epsilon$.

Theorem 4 *(Convergence properties of DCA-SI)*

(i) ***DCA-SI*** *generates a sequence* $\{W^r, Z^r, \Lambda^r\}$ *such that the sequence* $\{F(W^r, Z^r, \Lambda^r)\}$ *is monotonously decreasing.*
(ii) ***DCA-SI*** *has a linear convergence.*
(iii) The sequence $\{W^r, Z^r, \Lambda^r\}$ *generated by* ***DCA-SI*** *converges to a critical point of* $F = G - H$.

Proof: (i) - (iii) are direct consequences of the convergence properties of general DC programs.

4.1.4 Finding a good starting point of DCA

Finding a good starting point is an important question while designing DCA schemes. The research of such a point depends on the structure of the problem being considered and can be done by, for example, a heuristic procedure. Generally speaking a good starting point for DCA must not be a local minimizer, because DCA is stationary from such a point. As proposed in (Le Thi et al. [2008a]), we use an alternative SCAD - DCA-SI procedure for (4.11) which is described as follows.

SCAD - DCA-SI procedure
- **Initialization:** Choose randomly W^0, Z^0 and Λ^0. Let $maxiter > 0$ be a given integer. Set $s = 0$.
- **Repeat**
  - Perform one iteration of **SCAD** from (W^s, Z^s, Λ^s).
  - Perform one iteration of **DCA-SI** from the solution given by **SCAD** to obtain $(W^{s+1}, Z^{s+1}, \Lambda^{s+1})$.
  - $s = s + 1$.
- **Until** $s = maxiter$.

In our experiments, we use $maxiter = 2$.

4.1.5 Computational experiments

4.1.5.1 Protocol testing

We compare the performance of our method (**DCA–SI** with alternative procedure for finding the starting point) with three methods: **SCAD** (Frigui and Nasui [2004]) - an algorithm working with fuzzy weighted features, **DCAFCM** (Le Thi et al. [2007c]) and **FCM** (Bezdek [1981]) - two algorithms dealing with fuzzy model.

The parameters of each algorithm are chosen as follows: $\mu \in [1.1, \ldots, 4.0]$ for both FCM and DCAFCM, $\mu, \beta \in [1.1, \ldots, 4.0]$ for SCAD, and $\mu, \beta \in [2.0, \ldots, 4.0]$ for DCA–SI. DCA based algorithms (DCA–SI and DCAFCM) are stopped with the tolerance $\epsilon = 10^{-4}$.

As in Frigui and Nasui [2004], each pixel is mapped to an 8–dimensional feature vector consisting of three colors, three texture features and the two coordinates of pixels. The three color features are L^*a^*b coordinates of the color image. The three texture features (polarity, anisotropy and contrast (cf. Belongie et al. [1998], Frigui and Nasui [2004]) are computed as follows. The image $I(x, y)$ is convolved with Gaussian smoothing kernels $G_\delta(x, y)$ of several scales δ: $M_\delta(x, y) = G_\delta(x, y) \otimes (\Delta I(x, y))(\Delta I(x, y))^t$.

- The polarity is defined by $p = |E_+ - E_-|/(E_+ - E_-)$, where E_+ and E_- represent, respectively, the number of gradient vectors in the matrix Gauss kernels $G_\delta(x, y)$ of scale δ at the pixel (x, y) on the positive and negative sides of the dominant orientation. For each pixel, an optimal scale value is selected such that it corresponds to the value where polarity stabilizes with respect to scale.
- The anisotropy is computed by $a = 1 - \lambda_2/\lambda_1$, where λ_1, λ_2 are the eigenvalues of $M_\delta(x, y)$ at the selected scale.
- The texture contrast is defined as $c = 2(\sqrt{\lambda_1 + \lambda_2})^3$.

4.1.5.2 Testing on the labeled images

We perform image segmentation on 7 images taken from *Berkeley segmentation dataset (BSD)* (Martin et al. [2001]). These images consist of the real labels of pixels.

We using the PWCO index (percentage of well classified objects) to evaluate the segmentation results obtained by four algorithms. The Table 4.1 presents the results of all experiments and the Figure 4.1 describes a chart of the results.

We consider now three examples: image 113044, image 12003 and image 134052.

We would like to partition the image 113044 (Figure 4.3) into two clusters: "horses" and "grasses". Comparing the methods using feature weights (DCA–SI and SCAD) with the methods without feature weights (DCAFCM and FCM) we observe that the first approach is clearly better. DCA–SI and SCAD detect clearly two part: "horses" and "grasses", while DCAFCM and FCM detect incorrectly some pixel "grasses" (approximate 6.5% with DCAFCM and 12.5% with FCM). Comparing the effectiveness of DCA based algorithms with two algorithms: SCAD and FCM, we see that DCA–SI is better than SCAD with the accuracy 97.49% in comparing with 96.98% ; DCAFCM is better than FCM with the accuracy 93.55% in comparing with 87.50%.

The Image 12003 (Figure 4.4) is a picture of a starfish. This image consists of 3 clusters: the starfish, the coral (on the upper left and on the right side) and the moss. The clusters of this image are not separate, the coral seems transparent with the moss background, so segmentation task of this image is not easy. All of four methods can not detect well the coral class, while the starfish is separated from two remain classes. The best accuracy is 90.41% (DCA–SI) and the worst is 51.42% (FCM).

The leopard image (Image 134052) in the Figure 4.5, has 2 clusters: leopard and background. We see that DCAFCM and FCM detect some background pixels belonging to leopard and a part of body leopard as background. This result again shows that using features weighting are better. The accuracy of DCA–SI as well as SCAD is: 96.61%, while the accuracy of DCAFCM and FCM is 80.61%.

Table 4.1: The results of PWCO(%) of 4 methods

Image	Size	No. Classes	DCA–SI	SCAD	DCAFCM	FCM
113044	321×481	2	**97.49**	96.98	93.55	87.50
124084	321×481	3	**93.92**	82.86	75.19	74.35
113016	321×481	2	**97.06**	96.76	86.54	92.68
12003	321×481	3	**90.41**	71.05	64.56	51.42
134052	321×481	2	96.60	**96.61**	80.61	80.61
35070	321×481	3	**83.06**	80.44	75.70	65.69
157032	321×481	6	**76.42**	68.12	68.71	42.08

The Table 4.1 (resp. Table 4.2) shows the PWCO values (resp. the CPU time values) of four algorithms on 7 testing images. The name and the size of images are defined in *Berkeley segmentation dataset* (Martin et al. [2001]). The last four columns are the PWCO values (resp. CPU time values) of each algorithm. The Figure 4.1 (resp. Figure 4.2) presents the chart of PWCO values (resp. CPU time values).

From these tables, we observe that:

i) Comparing the fuzzy clustering algorithms with and without features weighting, say DCA–SI and SCAD versus DCAFCM and FCM:
DCA–SI gives better PWCO than DCAFCM and FCM on all data sets (7/7 data sets); the gain of DCA–SI goes up to 25.9% (in comparing with DCAFCM) and 39% (in comparing with FCM) in the

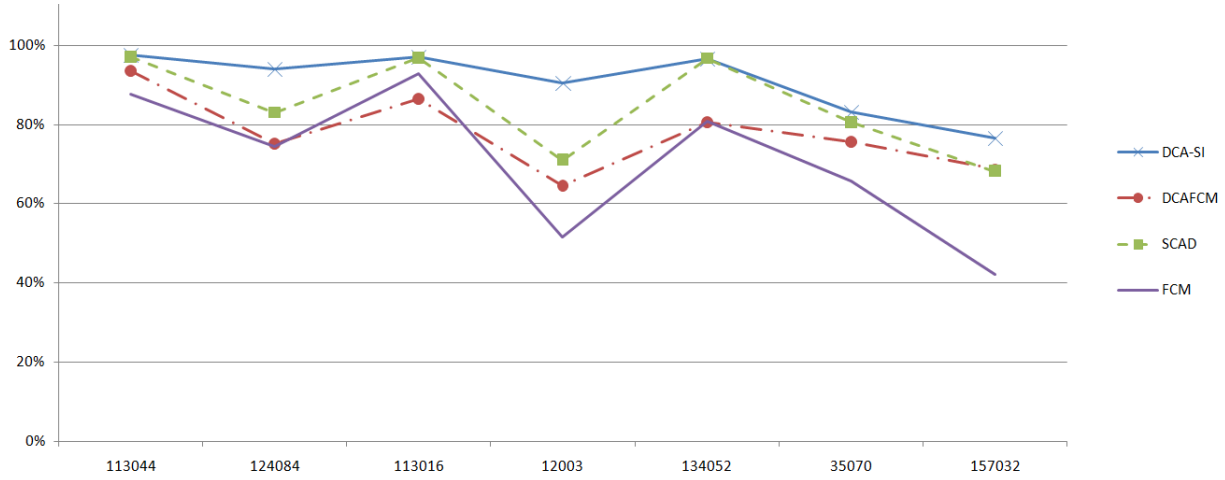


Figure 4.1: Accuracy.

Table 4.2: CPU Time running in seconds

Image	**Size**	**No. Classes**	**DCA–SI**	**SCAD**	**DCAFCM**	**FCM**
113044	321×481	2	78.35	42.32	24.10	9.22
124084	321×481	3	94.31	79.63	32.63	13.64
113016	321×481	2	86.36	44.13	21.90	9.08
12003	321×481	3	108.37	253.22	32.74	14.04
134052	321×481	2	19.49	38.42	16.73	12.43
35070	321×481	3	98.64	112.82	32.85	13.51
157032	321×481	6	164.35	623.07	59.20	27.27

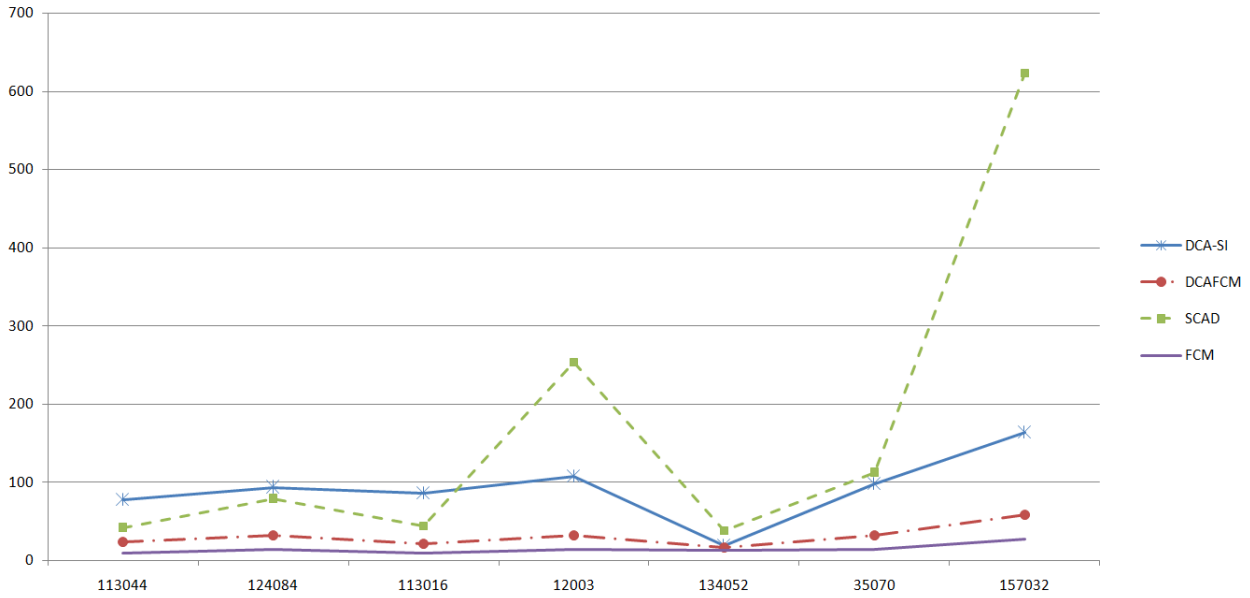


Figure 4.2: CPU Time running in seconds.

starfish image (Figure 4.4);
SCAD is better than FCM on all cases. SCAD is worse than DCAFCM in a case of Image 157032 (68.12% with 68.71%) and better than DCAFCM in the remaining cases;

ii) DCA–SI gives better PWCO than SCAD with a big gain on Image 12003 (19.4%) and Image 124084 (11.1%). The PWCO values of DCA–SI are greater than the PWCO values of SCAD on 6/7 data sets, while SCAD is slightly better than DCA–SI on Image 134052 (96.61% versus 96.60%).

iii) FCM is fastest algorithm and SCAD is slowest.

4.1.5.3 Testing on the unlabeled images

In this section, we also use the images taken from *Berkeley Segmentation Dataset* (BSD) (Martin et al. [2001]) (Image 196027, 35049, 41004 and 238011). However, these images do not include the real label of pixels. In addition, we perform segmentation with another synthetic image, which include some different types of shapes (same as in Busin et al. [2005]), and peppers image, which is taken from the *USC-SIPI Image Database* (SIPI Image Database).

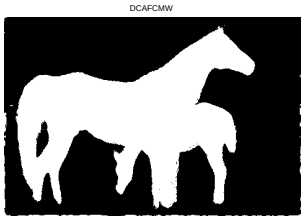
We use an evaluation function proposed by Borsotti et al. (Borsotti et al. [1998]) in 1998, that is one of standard criterion, to evaluate the quality of segmentation. The target of image segmentation is partitioning the domain–independent of the image into a set of regions which are visually distinct and uniform with respect to some properties, such as grey level, texture or color,... The quality solutions will be evaluated based on both the values of Borsotti function and the visual segmentation results.

The Borsotti function is given by the equation:

$$Q(I) = \frac{1}{10000(N \times M)} \sqrt{k} \sum_{i=1}^{k} [\frac{e_i^2}{1 + \log A_i} + (\frac{R(A_i)}{A_i})^2] \tag{4.15}$$

where I is the segmented image, $N \times M$ is the size of image, and k is the number of regions of the segmented image, while A_i and e_i are, respectively, the area and the average color error of the i–th region; e_i is defined

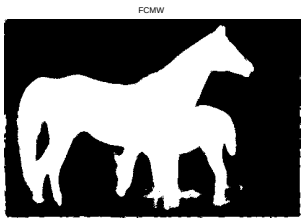
(a) Original image



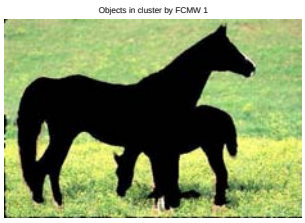
(a) DCA–SI

(b) Class 1

(c) Class 2

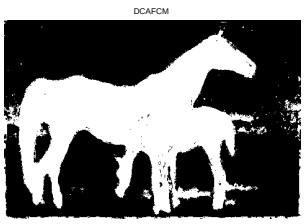


(a) SCAD

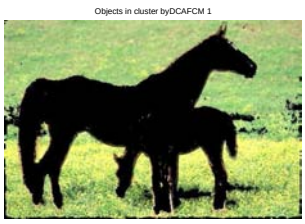


(b) Class 1

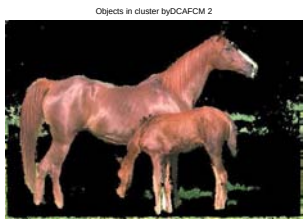
(c) Class 2



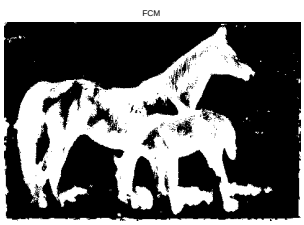
(a) DCAFCM



(b) Class 1

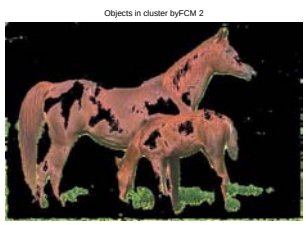


(c) Class 2



(a) FCM

(b) Class 1



(c) Class 2

Figure 4.3: Image 113044

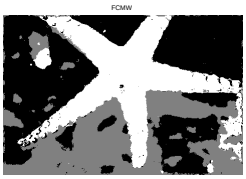
(a) Original image

(a) DCA–SI

(b) Class 1

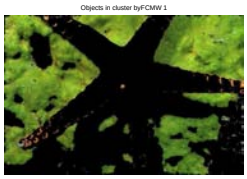
(c) Class 2

(d) Class 3

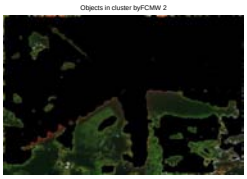


(a) SCAD

(b) Class 1



(c) Class 2



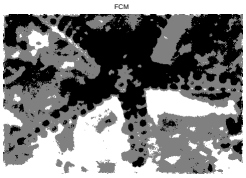
(d) Class 3

(a) DCAFCM

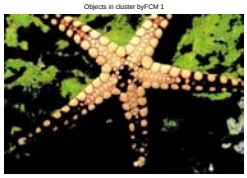
(b) Class 1

(c) Class 2

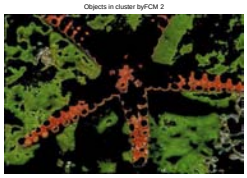
(d) Class 3



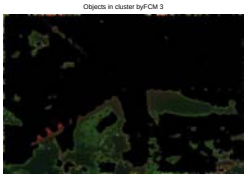
(a) FCM



(b) Class 1



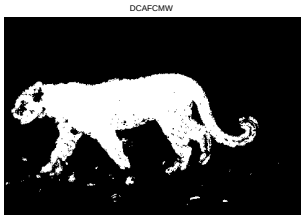
(c) Class 2



(d) Class 3

Figure 4.4: Image 12003

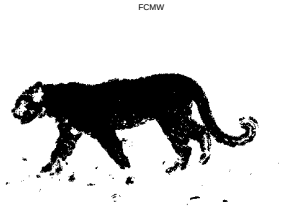
(a) Original image



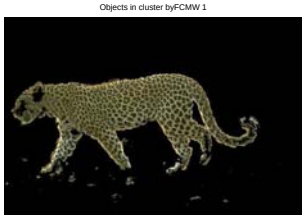
(a) DCA–SI

(b) Class 1

(c) Class 2

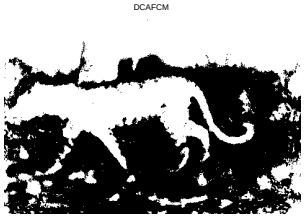


(a) SCAD



(b) Class 1

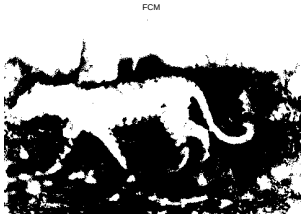
(c) Class 2



(a) DCAFCM

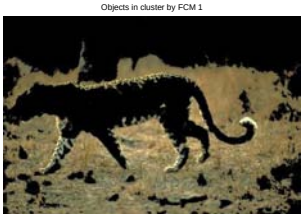
(b) Class 1

(c) Class 2



(a) FCM

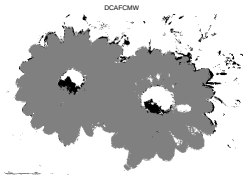
(b) Class 1



(c) Class 2

Figure 4.5: Image 134052

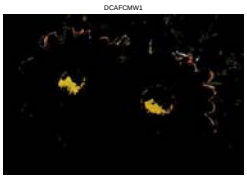
(a) Original image



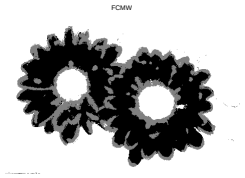
(a) DCA–SI

(b) Class 1

(c) Class 2

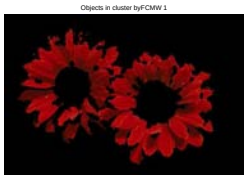


(d) Class 3

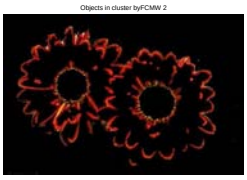


(a) SCAD

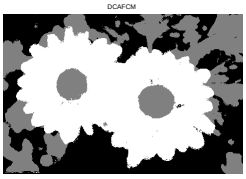
(b) Class 1



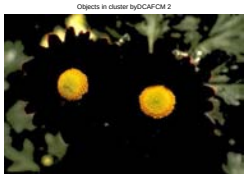
(c) Class 2



(d) Class 3

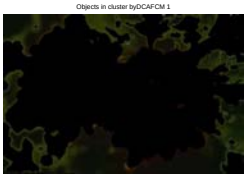


(a) DCAFCM

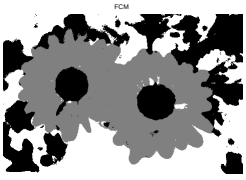


(b) Class 1

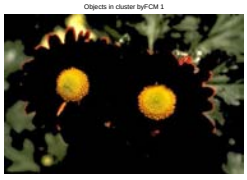
(c) Class 2



(d) Class 3

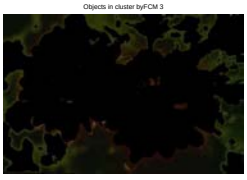


(a) FCM



(b) Class 1

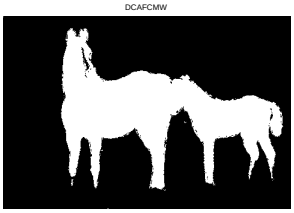
(c) Class 2



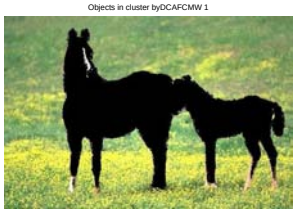
(d) Class 3

Figure 4.6: Image 124084

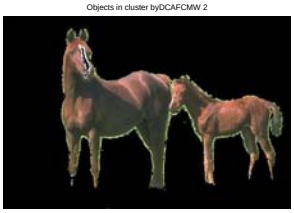
(a) Original image



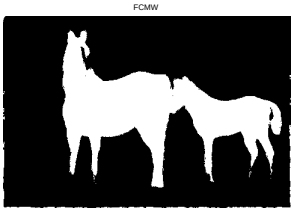
(a) DCA–SI



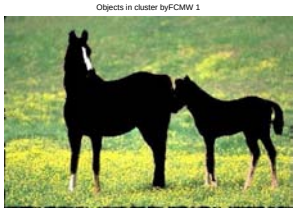
(b) Class 1



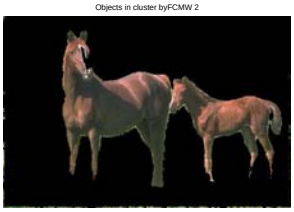
(c) Class 2



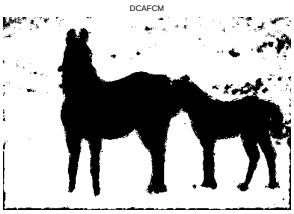
(a) SCAD



(b) Class 1

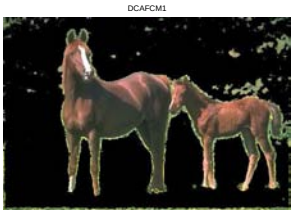


(c) Class 2

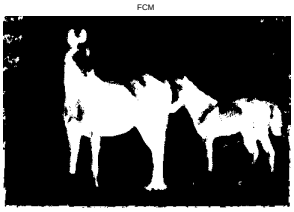


(a) DCAFCM

(b) Class 1

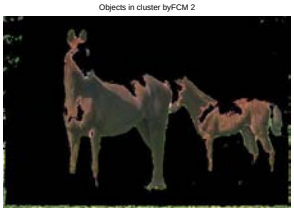


(c) Class 2



(a) FCM

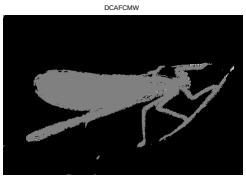
(b) Class 1



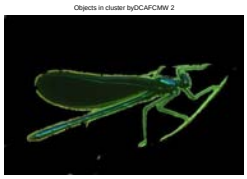
(c) Class 2

Figure 4.7: Image 113016

(a) Original image

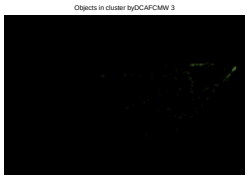


(a) DCA–SI

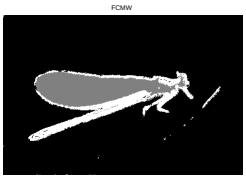


(b) Class 1

(c) Class 2



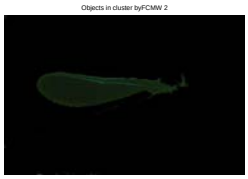
(d) Class 3



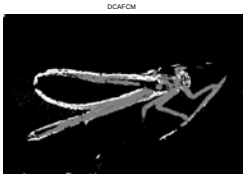
(a) SCAD

(b) Class 1

(c) Class 2



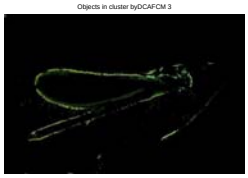
(d) Class 3



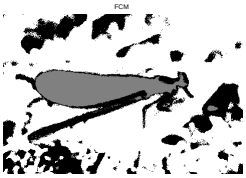
(a) DCAFCM

(b) Class 1

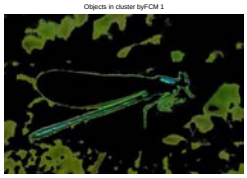
(c) Class 2



(d) Class 3

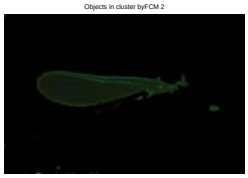


(a) FCM



(b) Class 1

(c) Class 2



(d) Class 3

Figure 4.8: Image 35070

(a) Origin image

(a) DCA–SI (b) Class 1 (c) Class 2 (d) Class 3 (e) Class 4 (f) Class 5 (g) Class 6

(a) SCAD (b) Class 1 (c) Class 2 (d) Class 3 (e) Class 4 (f) Class 5 (g) Class 6

(a) DCAFCM (b) Class 1 (c) Class 2 (d) Class 3 (e) Class 4 (f) Class 5 (g) Class 6

(a) FCM (b) Class 1 (c) Class 2 (d) Class 3 (e) Class 4 (f) Class 5 (g) Class 6

Figure 4.9: Image 157032

Table 4.3: The value of Borsotti function $Q(I)$

Problem	Size	No. Classes	DCA-SI	SCAD	DCAFCM	FCM
peppers	512×512	2	**185.395**	219.551	193.267	235.231
shapes	256×256	5	**288.836**	303.837	308.988	396.308
196027	321×481	3	**20.992**	24.762	32.531	35.782
35049	321×481	3	**9.202**	10.638	29.102	25.983
41004	321×481	3	**52.995**	68.782	95.280	112.409
238011	321×481	3	**25.102**	31.001	31.677	43.576

as the sum of the Euclidean distances between the color vectors of the pixels of region i and the color vector attributed to region i in the segmented image, $R(A_i)$ represents the number of regions having an area equal A_i. Good segmentation is distinguished by small value of $Q(I)$.

The Table 4.3 shows the Borsotti function $Q(I)$ results of four algorithms on the tested images. We now consider in detail the results of segmented images.

In the figure 4.10, we would like to separate the image into two regions: the red peppers and the green peppers. We observed that DCA–SI can detect the green bell pepper better than SCAD and DCAFCM and also FCM. The results of DCAFCM is quite similar to the results of DCA–SI in both the segmentation image and the Q value (185.395 in compare with 193.267). SCAD has some mistakes in the green bell pepper. In FCM image result, only the boundaries of each peppers were detected, that is worst case.

Figure 4.11 contains 5 regions based on their shapes and colors: the background, the green circle, the orange square, the yellow square, and the pink region. The first row shows that the segmentation obtained by DCA–SI is quite good. DCA–SI detects well 3 out of 5 regions (the pink region, the green circle, and the background). However, DCA-SI puts the yellow square and the orange square into one segment, since the two colors are close together, the last segment is the edge of circle. Concerning SCAD and DCAFCM, that are mistake in separate two squares: yellow and orange, and some points in the edges of squares. FCM has another mistake when separate the background into two regions. The gain of DCA–SI (the best case) and FCM (the worst case) up to 107.472.

The figure 4.12 (Image 196027) consists of 3 classes: the parrot, the wooden stakes, and the background. Algorithm FCM detects that the parrot is the first class, but it lost the beak. In addition, the wooden stakes class and the background are bad. In the images segmentation result of DCAFCM algorithm, the parrot is detected better, but it is not clearly in the part tail. The class wooden stakes and the background have some mistakes in the stakes and around the head of parrot. In the result of DCA–SI, the class background confused with some spots in the stakes, while SCAD'result is more mistakes, which include a part of the parrot's tail. It proves that our method is more efficient than others.

The image 35049 (figure 4.13) also include 3 classes: the butterfly, the flower, and the background. DCA-SI and SCAD detect quite good two classes: the butterfly and the flower; but SCAD has some mistakes when it detects the wing of butterfly. DCAFCM and FCM have confused when clustering the flower into butterfly class but FCM has mixed background class and flower class. It shows that using feature weighting allows us to improve greatly the quality of segmentation.

The image 41004 (figure 4.14) contains 3 classes: the deer, the wheat field, and the forest. DCA-SI and SCAD detected well the classes of wheat field but they had some mistakes in some parts of the deer and the forest. DCA-SI gave the less mistake result than SCAD while separate the deer and the forest. DCAFCM and FCM have confused more seriously when clustering the deer into forest class but FCM also had the mistake when detected the wheat field class. In this image, the running time of DCA–SI is the best, it only takes 15 seconds (while SCAD takes 91.166 seconds, and DCAFCM takes 37.341 seconds).

Table 4.4: Performance time (seconds)

Problem	Size	No. Classes	DCA-SI	SCAD	DCAFCM	FCM
peppers	512×512	2	68.881	54.64	49.505	24.195
shapes	256×256	5	14.96	61.17	13.25	14.92
196027	321×481	3	59.01	40.62	16.95	21.17
35049	321×481	3	52.048	80.268	19.821	9.354
41004	321×481	3	15.107	91.166	37.341	28.089
238011	321×481	3	9.098	52.584	36.554	25.141

From three cases (images: 196027, 35049, 41004), we observe the superiority of algorithms using weighted features: DCA–SI and SCAD are better than DCAFCM and FCM in term of Q values.

The image 238011 (figure 4.15) has 3 classes: the lunar, the sky and the trees. Only DCA–SI could detect well the three classes. SCAD and DCAFCM are the same mistake when segmenting the lunar into class sky, while they divided the edges of the trees and the sky in one class, and the rest in the third class. FCM had the worst result when detected only the trees class and segmented the sky into two classes. This result again shows that using DCA approach and feature weighting allowed us to improve the performance segmentation.

From the results, we can conclude that:

i) In term of Borsotti function as well as the segmentation image, DCA applied to the model using feature weighting gives the best results and the model using weighted features is more efficient than the model without weighted features.

ii) In comparing two algorithms without the weighted features, we see that DCAFCM algorithm is better than FCM on 5/6 cases in term of Q values.

iii) The running time of FCM is less expensive in average: 20.478 seconds, while SCAD is most time consuming: 63.408 seconds.

4.1.6 Conclusion

In this section, we have studied DC programming and DCA for image segmentation via weighted feature fuzzy clustering. The optimization model has been formulated as a DC program. The resulting problem, then, is solved by a DCA based algorithm. It fortunately turns out that, at each iteration, the projections of points onto a simplex and/or a rectangle are given in the explicit form. In our experiments, we have compared 4 algorithms: DCA–SI, SCAD, DCAFCM and FCM. The results show the efficiency of weighted feature measures, which allows to improve the results of segmentation. Furthermore, computational experiments show the superiority of DCA–SI with respect to the other algorithms. We are convinced that our approach is promising for weighted feature fuzzy clustering.

(a) Original image

(a) DCA–SI

(b) Class 1

(c) Class 2

(a) SCAD

(b) Class 1

(c) Class 2

(a) DCAFCM

(b) Class 1

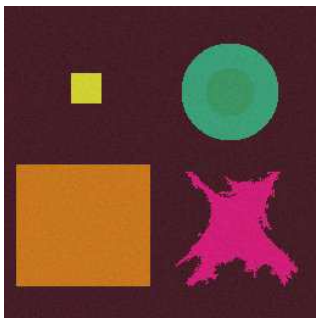
(c) Class 2

(a) FCM

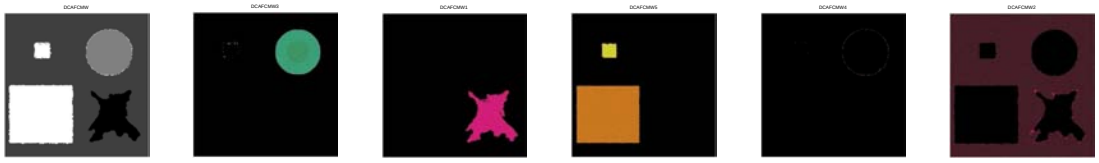
(b) Class 1

(c) Class 2

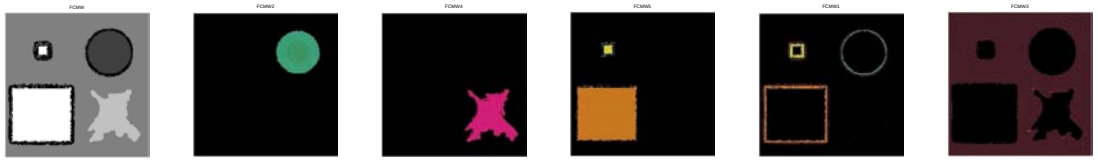
Figure 4.10: Image Peppers



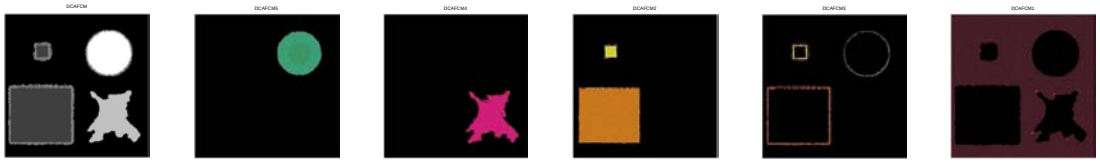
(a) Original image



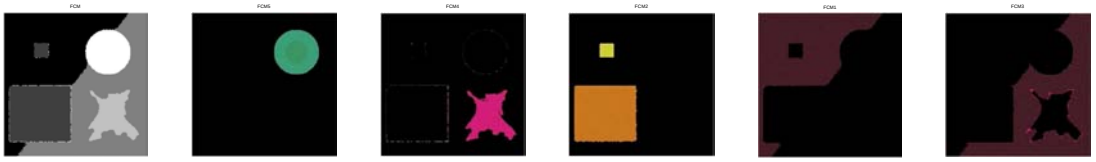
(a) DCA–SI (b) Class 1 (c) Class 2 (d) Class 3 (e) Class 4 (f) Class 5



(a) SCAD (b) Class 1 (c) Class 2 (d) Class 3 (e) Class 4 (f) Class 5



(a) DCAFCM (b) Class 1 (c) Class 2 (d) Class 3 (e) Class 4 (f) Class 5



(a) FCM (b) Class 1 (c) Class 2 (d) Class 3 (e) Class 4 (f) Class 5

Figure 4.11: Image Shapes

(a) Original image

(a) DCA–SI (b) Class 1 (c) Class 2 (d) Class 3

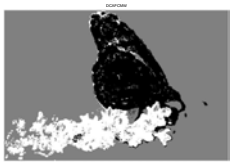
(a) SCAD (b) Class 1 (c) Class 2 (d) Class 3

(a) DCAFCM (b) Class 1 (c) Class 2 (d) Class 3

(a) FCM (b) Class 1 (c) Class 2 (d) Class 3

Figure 4.12: Image 196027

(a) Original image

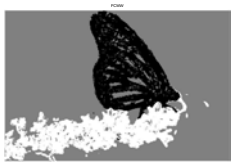


(a) DCA–SI

(b) Class 1

(c) Class 2

(d) Class 3

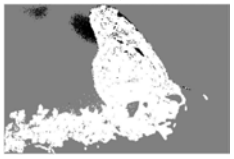


(a) SCAD

(b) Class 1

(c) Class 2

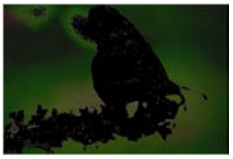
(d) Class 3



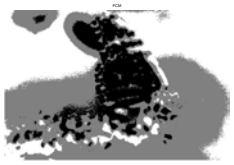
(a) DCAFCM

(b) Class 1

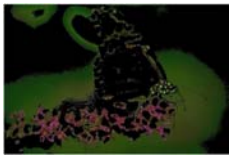
(c) Class 2



(d) Class 3

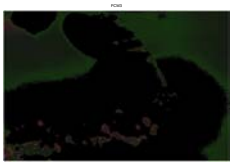


(a) FCM



(b) Class 1

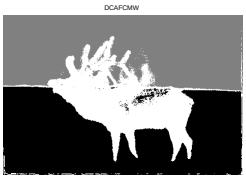
(c) Class 2



(d) Class 3

Figure 4.13: Image 35049

(a) Original image



(a) DCA–SI

(b) Class 1

(c) Class 2

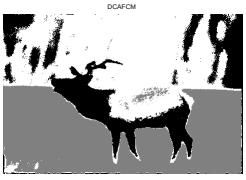
(d) Class 3

(a) SCAD

(b) Class 1

(c) Class 2

(d) Class 3

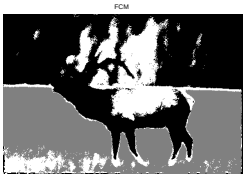


(a) DCAFCM

(b) Class 1

(c) Class 2

(d) Class 3



(a) FCM

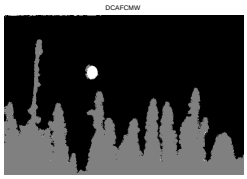
(b) Class 1

(c) Class 2

(d) Class 3

Figure 4.14: Image 41004

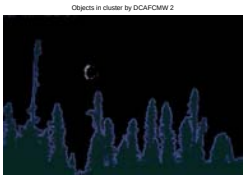
(a) Original image



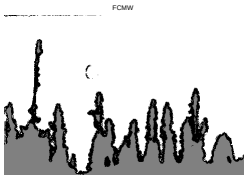
(a) DCA–SI

(b) Class 1

(c) Class 2

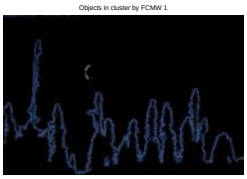


(d) Class 3

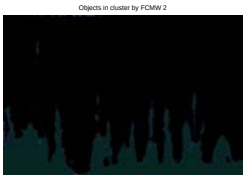


(a) SCAD

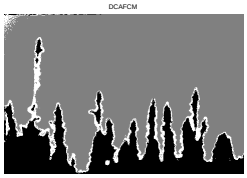
(b) Class 1



(c) Class 2

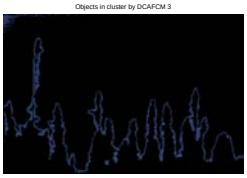


(d) Class 3

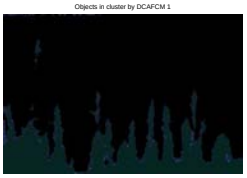


(a) DCAFCM

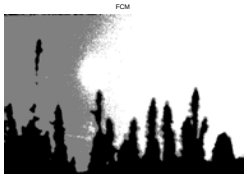
(b) Class 1



(c) Class 2

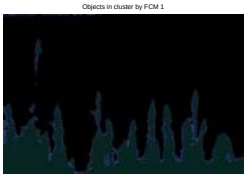


(d) Class 3

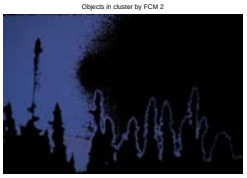


(a) FCM

(b) Class 1



(c) Class 2



(d) Class 3

Figure 4.15: Image 238011

4.2 Application to Cell counting problem

Cytological analysis, specially cell counting, is an important element in the diagnosis of many diseases. Cell segmentation, the major phase of cell counting procedure, basically performed by intensity thresholding, feature detection, morphological filtering, region accumulation and deformable model fitting. We present in this section an automatic method for cell counting with segmentation based on DCA–SI algorithm. This new application of our method can give promising results compared to the traditional manual analysis despite the very high cell density.

4.2.1 Introduction

Cytological analysis is an important element in the diagnosis of several diseases. The traditional method for an expert to achieve the differential counting is very tedious and time–consuming. Counting should be automated but it can become a complicated process.

Some examples of common techniques used in cell segmentation are thresholding (Liao and Deng [2002]), cell modeling (Liao and Deng [2002]), filtering, mathematical morphology (Anoraganingrum [1999]), watershed clustering (Jiang et al. [2003]) and fuzzy sets (Theera-Umpon [2005]). Each algorithm is ultimately a combination of segmentation methods for adaptation to cell types. The development of an efficient segmentation algorithm, the main step of cell counting, constitutes a challenger for researchers in this domain.

In this section, we will apply our algorithm introduced in previous section 4.1 for segmentation cell images. Then we perform mathematical morphology operations for counting the number of cells. Results of cell counting of SCAD and our method are compared subsequently with the manual analysis, considered as the reference.

4.2.2 Morphological Operations

Mathematical morphology was first introduced by Georges Matheron and Jean Serra (Matheron [1974], Serra [1983]). The basic operations of mathematical morphology are the dilation, erosion, closing and opening. The function of dilation is increasing the image while erosion makes it lower. Closing operation helps to close the inner hole region and eliminate the bays along the border area and opening is used to gets rid of small fragments, protruding regions near its borders. Based on these operations, various morphological operations were developed.

After segmentation phase, cells almost are defined in the binary image, but some noncellular particles were also present. The binary image is thus further processed to remove the objects that do not correspond to the cells of interest by applying some morphological operations: region filling to obtaining solid particles, filtering to removing small artifacts, etc.

Showed in **Fig.4.16** are the results of some morphological operations. **Fig.4.16(b)**, the ImFill(BW1, 'holes') function in Matlab was used to fill holes in the input image. In **Fig.4.16(c)**, BwAreaOpen(bw,p) helps to remove from a binary image all connected objects that have fewer than p pixels. The value of p is chosen large enough so that it can eliminate the wrong objects but still remains the correct cells.

In the final binary image there were still overlapping or touching cells. A further procedure using watershed was thus applied to separate these cells before counting. The idea of watershed comes from an example is finding a line which separates the U.S.A. into two regions. A drop of water falling on one side of this line flows down until it reaches the Atlantic Ocean, whereas a drop falling on the other side flows down to the Pacific Ocean. As we shall see in further detail later, this line constitutes a typical example of a watershed line. The two regions it separates are called the catchment basins of the Atlantic and the Pacific Oceans,

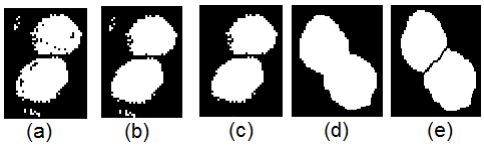


Figure 4.16: Some morphological operations: (a): after segmentation, (b): fill holes, (c): remove small objects, (d): overlapping cells, (e): separated cells by watershed

respectively. The two Oceans are the minima associated with these catchment basins (Vincent and Soille [1991]).

Fig.4.16(d) is the image of overlapping cells and **Fig.4.16(e)** is an example result after using watershed operation.

4.2.3 Computational experiments

Our algorithm were implemented in the Visual C++ 2008 combined with Matlab R2007a, and were performed on a PC Intel i5 CPU650, 3.2 GHz of 4GB RAM. Images for experiment are: Oligodendrocyte cells (one type of brain cells) (image **I1**), Oligodendrogliomas (Medical [home page]) (image **I2**) and Mouse liver cells (Nanotechweb [home page]) (image **I3**).

First, we perform clustering with three algorithms: **DCA-SI**, **SCAD** (Frigui and Nasui [2004]) and thresholding. The number of clusters is 2. Then, we apply some morphography operations: fill holes, remove small objects, overlapping cells and separate cells for counting the number of cells.

Fig.4.17 shows the comparative results of three algorithms: **DCA-SI**, **SCAD** and thresholding. In image **I1**, segmentation results of all three algorithms were good as cells were very clear and separable from the background. However in image **I2**, thresholding can not detect the cells with the blood as cells' colour and environment were close. Consequently, some mistakes were found in the segmentation result in row 4. while **DCA-SI** can eliminate the blood out (row 2). In **SCAD** result (row 3) it was difficult to count in the following phase because of existence of some blur region. In image **I3**, we obtained cells with much holes and fragments as the thresholding method separated cells based on pixels' intensity. In this case, **DCA-SI** and **SCAD** which used more information of a pixel (texture and position feature) gave the cells more solid and clear. Subsequently, cells segmented by **DCA-SI** were more sharpen and smooth than **SCAD**.

The counting results of three algorithms on three images are reported in the **Table 4.5** below.

Table 4.5: Cell counting results

Image	Size	DCA-SI	SCAD	Threshold	Manual
I1	1360×1024	434	477	434	430
I2	800 ×533	248	476	473	295
I3	400 ×292	126	133	80	123

This table showed that **DCA-SI** which has better results on segmentation phase can give more exact results in the counting phase.

Fig.4.18 are the counting results on images **I1**, **I2**, **I3** using **DCA-SI** segmentation method combined with some morphological operations above.

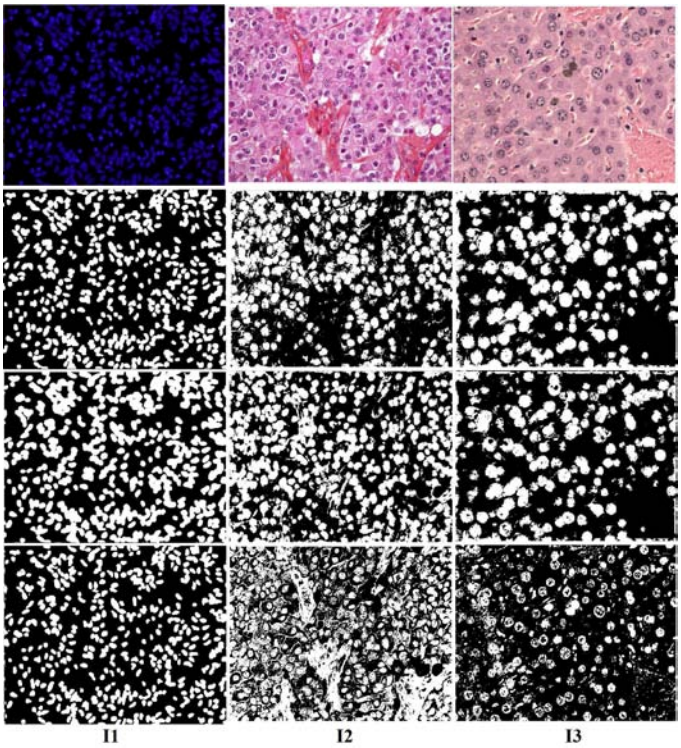


Figure 4.17: The rows show respectively: Original images, images segmented by **DCA-SI**, **SCAD**, thresholding

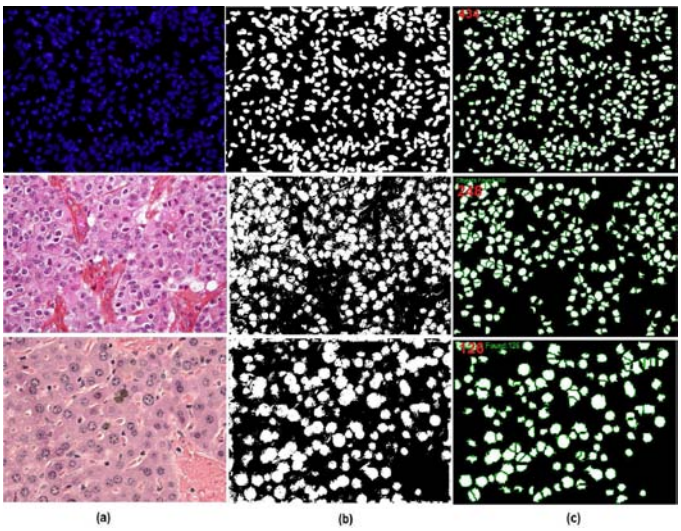


Figure 4.18: Original(a), Segmented by **DCA-SI** (b) and Count result(c) images

4.2.4 Conclusion

In this section, we have presented an automatic segmentation technique for microscope cell images which is an important step in cell counting problem. The proposed segmentation technique, based on Feature

Weighted Fuzzy Clustering via DCA, was evaluated by comparing with two others methods frequently used in cell counting problem. In our experiments, the counting results given by DCA are better than those of SCAD and thresholding. DCA method appears to be an effective segmentation technique in complex histological images and could be applied in other domains of medical image.

Conclusion and Future works

The signal/image processing problems had been appeared for a long time, but they are always a challenging research area. The problems relate with not only the size of data but also the complexity of processing.

In this thesis, we have focused on three problems: compressed sensing, dictionary learning and image denoising, image segmentation. We developed DC programming and DCA for solving some classes of these problems.

By the assumption that one signal/image can be represented by only a few non-zero coefficients in a suitable basis or dictionary, and one signal/image can recovery from a few measurements, this results opened a new direction, it is different with traditional ways, for the tasks of signal processing/image processing: acquisition, compression and storage, transmission, denoising,... It leads to two problems: 1) finding the sparse representation of a signal; 2) recovering a signal from the given measurements. In this thesis, we have been successfully applied DC and DCA to solves these problems.

The two first problems, sparse representation and sparse recovery lead to a NP–hard problem, non convex and discontinuous. A common approach for solving these problems is approximating the ℓ_0–norm by ℓ_1–norm and the problem is converted to a convex, continuous optimization problem. But, in many cases, some conditions may be not satisfied. Some experiments show that the non convex approach performs better than that based on the employment of the ℓ_1–norm. We have used four nonconvex approximations of the ℓ_0–norm applied to three models: linear constraint, least–square constraint and regularization least square models. The numerical results have shown the effective, stability and the power of DC programming and DCA.

For the dictionary learning problem, we presented a DCA based algorithm with two stages. In the first stage, sparse coding, we used one of the approximations above, and we developed a DC algorithm for solving the convex subproblem. We also proposed a DCA for second problem in the dictionary updating stage. The learned dictionary has been used on the problem of image denoising. The training data was a set of image patches, which were extracted from the noisy image. A comparison with the standard algorithm K–SVD has been performed and the results showed the effectiveness of our algorithm.

The final issue addressed in this thesis is the image segmentation problem. We proposed a DCA approach, where the problem was recast as a DC program. Then, a DC algorithm has been developed to solve the resulting problem. It fortunately turns out that the corresponding DCA consists in computing, at each iteration, the projection of points onto a simplex and/or a rectangle, that all are given in the explicit form. We tested our algorithms on two types of image: labeled images and unlabeled images.

We applied our approach to an application in medicine, cell counting problem. This is an important task in the diagnosis of many diseases, but automatic counting is not easy. By combining with some morphological operations, the given results are very promising.

Concerning the future works, we plan to develop new models for our problems. The studies of the DC decomposition as well as the strategies of initial points in the DC algorithms are still open issues.

From the sparsity of signal/image, we can investigate our approaches in other domains, such as: biomedical

imaging, genomic signal processing, ...

Based on the advantages of the learning dictionary, we intend develop DC programming and DCA for online dictionary learning processing or execute other methods to solving the convex subproblems.

The segmentation image problem without a prior knowledge of the segment number is still difficult task. We can study a schema that combines our method with other approaches to detect automatically the segments numbers and perform segmentation at the same time.

In the future, we intend to apply the DC Programming and DCA in other domains such as image compressing, inpainting, face recognition, ... We believe that DCA is an innovative approach for signal / image processing, as well as for nonconvex programming, non smooth and/or large–scale problems.

References

M. Aharon, M. Elad, and A. Bruckstein. K–svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, Nov 2006. ISSN 1053-587X.

Dwi Anoraganingrum. Cell segmentation with median filter and mathematical morphology operation. In *Proc Intl Conf on Image Anal and Proc*, pages 1043–1046, 1999.

H. Attouch, J. Bolte, P. Redont, and A. Soubeyran. Proximal alternating minimization and projection methods for nonconvex problems. an approach based on the kurdyka-lojasiewicz inequality. *Mathematics of Operations Research*, 35, no. 2:438–457, 2010.

A. Auslender. *Optimisation: méthodes numériques*. Maîtrise de mathématiques et applications fondamentales. Masson, 1976. ISBN 978-2225429002.

H. H. Bauschke and P.L. Combettes. Convex analysis and monotone operator theory in hilbert spaces. *Springer*, 2011.

Serge Belongie, Chad Carson, Hayit Greenspan, and Jitendra Malik. Color- and texture-based image segmentation using em and its application to content-based image retrieval. In *Proceedings of the Sixth International Conference on Computer Vision*, ICCV '98, pages 675–, Washington, DC, USA, 1998. IEEE Computer Society. ISBN 81-7319-221-9.

James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 1981. ISBN 0306406713.

J. Bolte, S. Sabach, and M. Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Mathematical Programming Series A*, (146)(1-2):459–494, 2014.

M. Borsotti, P. Campadelli, and R. Schettini. Quantitative evaluation of color image segmentation results. *Pattern Recognition Letters*, 19(8):741–747, 1998.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1): 1–122, 2011.

Paul S. Bradley and Olvi L. Mangasarian. Feature selection via concave minimization and support vector machines. In *ICML*, pages 82–90, 1998.

Matthew Brand. A short note on local region growing by pseudophysical simulation. In *Conference on Computer Vision and Pattern Recognition, CVPR 1993, 15-17 June, 1993, New York, NY, USA*, pages 782–783, 1993.

L. Busin, N. Vandenbroucke, and L. Macaire. Color spaces and image segmentation. *Advances in Imaging and Electron Physics*, 151:65–168, 2005.

P. Campadelli, D. Medici, and R. Schettini. Color image segmentation using hopfield networks. *Image and Vision Computing*, 15(3):161 – 166, 1997. ISSN 0262-8856.

E. J. Candes and P. A. Randall. Highly robust error correction by convex programming. *IEEE Transactions on Information Theory*, 54(7):2829–2840, July 2008.

E. J. Candes and T. Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51 (12):4203–4215, December 2005. ISSN 0018-9448.

E. J. Candes and T. Tao. The Dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics*, 35(6):2313–2351, December 2007. ISSN 0090-5364.

E. J. Candes, J. K. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inf. Theor.*, 52(2):489–509, February 2006a. ISSN 0018-9448.

E. J. Candes, J. K. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Comm. Pure Appl. Math.*, 59(8):1207–1223, August 2006b. ISSN 0010-3640.

E. J. Candes, M. B. Wakin, and S. P. Boyd. Enhancing sparsity by reweighted L1 minimization. *Journal of Fourier Analysis and Applications*, 14(5):877–905, 2008.

E.J. Candes and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, Dec 2006. ISSN 0018-9448.

Elaine Y. Chan, Wai-Ki Ching, Michael K. Ng, and Joshua Zhexue Huang. An optimization algorithm for clustering using weighted dissimilarity measures. *Pattern Recognition*, 37(5):943–952, 2004.

M. Chapron. A new chromatic edge detector used for color image segmentation. In *In Proc. 11th Int. Conf. on Pattern Recognition*, volume 3, pages 311–314, 1992.

R. Chartrand. Exact reconstruction of sparse signals via nonconvex minimization. *IEEE Signal Process.Lett.*, 14(10):707–710, 2007.

R. Chartrand and W. Yin. Iteratively reweighted algorithms for compressive sensing. *IEEE international conference on acoustics, speech, and signal processing*, 2008.

S. S. Chen, D. L Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.

H. D. Cheng, X. H. Jiang, and Jingli Wang. Color image segmentation based on homogram thresholding and region merging. *Pattern Recognition*, 35:373–393, 2002.

O. Cheng Soon and H. A. Le Thi. Learning sparse classifiers with difference of convex functions algorithms. *Optimization Methods and Software*, 28(4):830–854, 2013.

P. Combettes and J. Pesquet. Proximal splitting methods in signal processing. *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, 20, NO. 3,:185–212, 2011.

Resources Compressive Sensing (home page). Resources compressive sensing. http://www.compressedsensing.com/. Accessed on March 2014.

Aldo Cumani. Edge detection in multispectral images. In *Computer Vision, Graphic and Image Processing: Graphical Models and Image Processing*, volume 53, pages 40–51, 1989.

W. Dai and O. Milenkovic. Subspace pursuit for compres sive sensing: Closing the gap between performance and complexity. *IEEE Trans.Info. Theory*, vol. 55, no. 5:2230–2249, May 2009.

Wei Dai, Tao Xu, and Wenwu Wang. Simultaneous codeword optimization (simco) for dictionary update and learning. *IEEE Transactions on Signal Processing*, 60(12):6340–6353, 2012.

Ivan Damnjanovic, Matthew E. P. Davies, and Mark D. Plumbley. Smallbox – an evaluation framework for sparse representations and dictionary learning algorithms. http://code.soundsoftware.ac.uk/projects/smallbox/, 2010. Proc. LVA/ICA'10, pp. 418–425. Accessed on March 2014.

George Dantzig. *Linear programming and extensions.* Princeton University Press, August 1963. ISBN 0691059136.

I. Daubechies, R. DeVore, M. Fornasier, and C. Güntük. Iteratively reweighted least squares minimization for sparse recovery. *Commun. Pure Appl. Math.*, 63:1–38, 2010.

D. L. Donoho. For most large underdetermined systems of linear equations the minimal ℓ_1-norm solution is also the sparsest solution. *Commun. Pure Appl. Math.*, 59:797–829, 2006a.

D. L. Donoho. Compressed sensing. *IEEE Trans. Inf. Theor.*, 52(4):1289–1306, April 2006b. ISSN 0018-9448.

D. L. Donoho and M. Elad. Optimally sparse representation in general (non-orthogonal) dictionaries via ell1 minimization. In *Proc. Natl Acad. Sci. USA 100 2197–202*, volume 100, pages 2197–2202, 2003.

D. L. Donoho and X. Huo. Uncertainty principles and ideal atomic decomposition. *IEEE Transactions on Information Theory*, 47(7):2845–2862, 2001.

D. L. Donoho and B. F. Logan. Signal recovery and the large sieve. *SIAM Journal on Applied Mathematics*, 52(2):577–591, April 1992. ISSN 0036-1399 (print), 1095-712X (electronic).

D.L. Donoho, Y. Tsaig, I. Drori, and J.-L. Starck. Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit. http://www-stat.stanford.edu/donoho/reports.html, 2006.

M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *Trans. Img. Proc.*, 15(12):3736–3745, December 2006. ISSN 1057-7149.

K. Engan, B. D. , Rao, and K. Kreutz-Delgado. Frame design using focuss with method of optimal directions (mod). *Proc. Norwegian Signal Processing Symposium*, pages 65–69, 1999a.

K. Engan, S. O. Aase, and H. J. Husoy. Method of optimal directions for frame design. In *Proceedings of the Acoustics, Speech, and Signal Processing, IEEE International Conference - Volume 05*, ICASSP '99, pages 2443–2446, Washington, DC, USA, 1999b. IEEE Computer Society. ISBN 0-7803-5041-3.

K Engan, K. Skretting, and J. H. Husoy. Family of iterative ls-based dictionary learning algorithms, ils-dla, for sparse signal representation. *Digital Signal Processing*, 17(1):32–49, 2007. ISSN 1051-2004.

E. Esser, Y. Lou, and J. Xin. A method for finding structured sparse solutions to non-negative least squares problems with applications. *SIAM J. Imaging Sciences*, 6(4):2010–2046, 2013.

Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties, 2001.

Alhussein Fawzi, Mike Davies, and Pascal Frossard. Dictionary learning for fast classification based on soft-thresholding. *arXiv:1402.1973v2 [cs.CV] 2Oct2014*, 2014.

S. Foucart and M. Lai. Sparsest solutions of underdetermined linear systems via ℓ_q-minimization for $0 < q \leq 1$. *Appl. Comput. Harmon. Anal.*, 26:395–407, 2009.

Hichem Frigui and Olfa Nasui. Unsupervised learning of prototypes and attribute weights. *Pattern Recognition*, 37(3):567–581, 2004.

W. J. Fu. Penalized regressions: The bridge versus the lasso. *Journal of Computational and Graphical Statistics*, 7:397–416, 1998.

Gilles Gasso, Alain Rakotomamonjy, and Stéphane Canu. Recovering sparse signals with a certain family of nonconvex penalties and DC programming. *IEEE Transactions on Signal Processing*, 57:4686–4698, 2009.

Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 6(6):721–741, November 1984. ISSN 0162-8828.

T. Goldstein and S. Osher. The split bregman method for ℓ_1-regularized problems. *SIAM Journal on Imaging Sciences*, 2(2):323–343, 2009.

I.F. Gorodnitsky and B.D. Rao. Sparse signal reconstructions from limited data using focuss: A re-weighted minimum norm algorithm. *IEEE Trans. Signal Processing*, 45:600–616, 1997.

Robert M. Haralick and Linda G. Shapiro. Image segmentation techniques, 1985.

Glenn Healey. Color. chapter Segmenting Images Using Normalized Color, pages 166–198. Jones and Bartlett Publishers, Inc., USA, 1992. ISBN 0-86720-295-5.

Reiner Horst and Hoang Tuy. *Global optimization - deterministic approache*. Springer, 1996. ISBN 978-3-540-61038-0.

Chung-Lin Huang, Tai-Yuen Cheng, and Chaur-Chin Chen. Color images' segmentation using scale space filter and markov random field. *Pattern Recognition*, 25(10):1217 – 1229, 1992. ISSN 0031-3203.

Joshua Zhexue Huang, Michael K. Ng, Hongqiang Rong, and Zichen Li. Automated variable weighting in k-means type clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(5):657–668, 2005.

Kan Jiang, Qing-Min Liao, and Sheng-Yang Dai. A novel white blood cell segmentation scheme using scale-space filtering and watershed clustering. In *Machine Learning and Cybernetics, 2003 International Conference on*, volume 5, pages 2820–2825, Nov 2003.

Liping Jing, Michael K. Ng, Huang, and Joshua Zhexue. An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *IEEE Transactions on Knowledge and Data Engineering*, 19(8):1026–1041, 2007. ISSN 1041-4347.

E. Kokiopoulou, D. Kressner, N. Paragios, and P. Frossard. Optimal image alignment with random projections of manifolds: algorithm and geometric analysis. In *Proceedings of EUSIPCO*, 2009.

K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T. Lee, and T. J. Sejnowski. Dictionary learning algorithms for sparse representation. *Neural Comput.*, 15:349–396, feb 2003. ISSN 0899-7667.

M-J. Lai, Y. Xu, and W. Yin. Improved iteratively reweighted least squares for unconstrained smoothed ℓ_p minimization. *SIAM J. Numer. Anal.*, Vol. 51, Issue 2:927–957, 2013.

P.J. Laurent. *Approximation et optimisation*, volume 1 of *Collection Enseignement des sciences*. Université Scientifique et Médicale de Grenoble, 1972.

M. Le Hoai and M. T. Ta. Dc programming and dca for solving minimum sum-of-squares clustering using weighted dissimilarity measures. *Transaction Computational Collective Intelligence*, 13:113–131, 2014.

M. Le Hoai, H. A. Le Thi, T. Pham Dinh, and Pascal Bouvry. A deterministic optimization approach for generating highly nonlinear balanced boolean functions in cryptography. In HansGeorg Bock, Ekaterina Kostina, HoangXuan Phu, and Rolf Rannacher, editors, *Modeling, Simulation and Optimization of Complex Processes*, pages 381–391. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-79408-0.

M. Le Hoai, H. A. Le Thi, T. Pham Dinh, and Pascal Bouvry. A combined dca: Ga for constructing highly nonlinear balanced boolean functions in cryptography. *Journal of Global Optimization*, 47(4):597–613, 2010. ISSN 0925-5001.

M. Le Hoai, Adnan Yassine, and Riadh Moussi. Dca for solving the scheduling of lifting vehicle in an automated port container terminal. *Computational Management Science*, 9(2):273–286, 2012. ISSN 1619-697X.

M. Le Hoai, H. A. Le Thi, and M. C. Nguyen. Dca based algorithms for feature selection in semi-supervised support vector machines. In *Machine Learning and Data Mining in Pattern Recognition*, volume 7988 of *Lecture Notes in Computer Science*, pages 528–542. 2013a. ISBN 978-3-642-39711-0.

M. Le Hoai, H. A. Le Thi, T. Pham Dinh, and V. N. Huynh. Block clustering based on difference of convex functions (dc) programming and dc algorithms. *Neural Computation*, 25(10):2776–2807, 2013b.

M. Le Hoai, T. B. T. Nguyen, M. T. Ta, and H. A. Le Thi. Image segmentation via feature weighted fuzzy clustering by a dca based algorithm. In *Advanced Computational Methods for Knowledge Engineering*, volume 479 of *Studies in Computational Intelligence*, pages 53–63. 2013c. ISBN 978-3-319-00292-7.

H. A. Le Thi. *Analyse numérique des algorithmes de l'optimisation DC. Approches locale et globale. Codes et simulations numériques en grande dimension. Applications.* Thèse de doctorat, Université de Rouen, 1994.

H. A. Le Thi. *Contribution à l'optimisation non convexe et l'optimisation globale: Théorie, Algoritmes et Applications.* Habilitation à diriger des recherches, Université de Rouen, 1997.

H. A. Le Thi. Dc programming and dca in machine learning. Technical report, University of Lorraine, 2012a. Submitted.

H. A. Le Thi. A new approximation for the l_0–norm. Research report lita ea 3097, University of Lorraine, France, 2012b.

H. A. Le Thi and M. Moeini. Long-short portfolio optimization under cardinality constraints by difference of convex functions algorithm. *Journal of Optimization Theory and Applications*, 161(1):199–224, 2014. ISSN 0022-3239.

H. A. Le Thi and T. Pham Dinh. Solving a class of linearly constrained indefinite quadratic problems by dc algorithms. *Journal of Global Optimization*, 11(3):253–285, 1997.

H. A. Le Thi and T. Pham Dinh. Dc programming approach for solving the multidimensional scaling problem. *Nonconvex Optimizations and Its Applications: Special Issue From Local to Global Optimization*, pages 231–276, 2001.

H. A. Le Thi and T. Pham Dinh. Dc programming: Theory, algorithms and applications. In *The State of the Proceedings of The First International Workshop on Global Constrained Optimization and Constraint Satisfaction (Cocos' 02)*, Valbonne-Sophia Antipolis, France, October, 2002.

H. A. Le Thi and T. Pham Dinh. Large-scale molecular optimization from distance matrices by a d.c. optimization approach. *SIAM Journal on Optimization*, 14(1):77–114, 2003.

H. A. Le Thi and T. Pham Dinh. The dc (difference of convex functions) programming and dca revisited with dc models of real world nonconvex optimization problems. *Annals of Operations Research*, 133:23–46, 2005.

H. A. Le Thi and T. Pham Dinh. Dc programming in communication systems: challenging problems and methods. *Vietnam Journal of Computer Science*, 1(1):15–28, 2014. ISSN 2196-8888.

H. A. Le Thi and D. Q. Tran. Solving continuous min max problem for single period portfolio selection with discrete constraints by DCA. *Optimization*, 61(8):1025–1038, 2012.

H. A. Le Thi and D. Q. Tran. Optimizing a multi-stage production/inventory system by dc programming based approaches. *Computational Optimization and Applications*, 57(2):441–468, 2014. ISSN 0926-6003.

H. A. Le Thi, T. Pham Dinh, and M. Le Dung. Exact penalty in d.c. programming. *Vietnam Journal of Mathematics*, 27(2):169–178, 1999.

H. A. Le Thi, T. Pham Dinh, and V.T. Nguyen. Combination between local and global methods for solving an optimization problem over the efficient set. *European Journal of Operational Research*, 142:257–270, 2002.

H. A. Le Thi, T. Pham Dinh, and V. N. Huynh. Exact penalty techniques in dc programming. *Research Report, LMI, National Institute for Applied Sciences - Rouen, France*, 2005.

H. A. Le Thi, M. T. Belghiti, and T. Pham Dinh. A new efficient algorithm based on dc programming and dca for clustering. *Journal of Global Optimization*, 37(4):593–608, 2007a.

H. A. Le Thi, M. Le Hoai, and T. Pham Dinh. Optimization based dc programming and dca for hierarchical clustering. *European Journal of Operational Research*, 183(3):1067–1085, 2007b.

H. A. Le Thi, M. Le Hoai, and T. Pham Dinh. Fuzzy clustering based on nonconvex optimisation approaches using difference of convex (dc) functions algorithms. *Adv. Data Analysis and Classification*, 1(2):85–104, 2007c.

H. A. Le Thi, T. P. Nguyen, and T. Pham Dinh. A continuous dc programming approach to the strategic supply chain design problem from qualified partner set. *European Journal of Operational Research*, 183 (3):1001 – 1012, 2007d. ISSN 0377-2217.

H. A. Le Thi, M. Le Hoai, T. P. Nguyen, and T. Pham Dinh. Noisy image segmentation by a robust clustering algorithm based on dc programming and dca. In *ICDM*, pages 72–86, 2008a.

H. A. Le Thi, M. Le Hoai, V. V. Nguyen, and T. Pham Dinh. A dc programming approach for feature selection in support vector machines learning. *Adv. Data Analysis and Classification*, 2(3):259–278, 2008b.

H. A. Le Thi, M. Le Hoai, V. V. Nguyen, and T. Pham Dinh. Combined feature selection and classification using dca. In *IEEE International Conference on Research, Innovation and Vision for the future in Computing & Communications Technologies, Ho Chi Minh (RIVF 2008)*, pages 233–239, July 2008c.

H. A. Le Thi, Q. T. Nguyen, K. Phan Tran, and T. Pham Dinh. Energy minimization-based cross-layer design in wireless networks. In *Proceeding of the High Performance Computing & Simulation Conference (HPCS 2008), Nicosia, Cyprus*, pages 283–289, June 3 - 6, 2008d.

H. A. Le Thi, V. V. Nguyen, and O. Samir. Gene selection for cancer classification using dca. In *Advanced Data Mining and Applications*, volume 5139 of *Lecture Notes in Computer Science*, pages 62–72. 2008e. ISBN 978-3-540-88191-9.

H. A. Le Thi, M. Moeini, and T. Pham Dinh. Portfolio selection under downside risk measures and cardinality constraints based on dc programming and dca. *Computational Management Science*, 6(4):459–475, 2009a. ISSN 1619-697X.

H. A. Le Thi, M. Moeini, and T. Pham Dinh. DC programming approach for portfolio optimization under step increasing transaction costs. *Optimization journal*, 58(3):267–289, 2009b.

H. A. Le Thi, Q. T. Nguyen, H. T. Nguyen, and T. Pham Dinh. Solving the earliness tardiness scheduling problem by dc programming and dca. *Mathematica Balkanica,*, pages 271–288, 2009c.

H. A. Le Thi, Q. T. Nguyen, H. T. Nguyen, and T. Pham Dinh. A time-indexed formulation of earliness tardiness scheduling via dc programming and dca. In *Computer Science and Information Technology, 2009. IMCSIT '09. International Multiconference on*, pages 779–784, Oct 2009d.

H. A. Le Thi, T. Pham Dinh, and S. Bouallagui. Cryptanalysis of an identification scheme based on the perceptron problem using a hybridization of deterministic optimization and genetic algorithm. In *Proceedings of the 2009 International Conference on Security and Management, World Congress in Computer Science Computer Engineering, and Applied Computing, Las Vegas, USA*, pages 117–123, July 13-16 2009e.

H. A. Le Thi, T. Pham Dinh, and V.N. Huynh. Exact penalty and error bounds in dc programming. *J. Global Optimization*, 52(3):509–535, 2012a.

H. A. Le Thi, T. Pham Dinh, and D. Q. Tran. A DC programming approach for a class of bilevel programming problems and its application in portfolio selection. *Numerical Algebra, Control and Optimization (NACO)*, 2(1):167–185, 2012b.

H. A. Le Thi, M. T. Le, and T. B. T. Nguyen. A novel approach to automated cell counting based on a difference of convex functions algorithm (dca). In *Computational Collective Intelligence. Technologies and Applications*, volume 8083 of *Lecture Notes in Computer Science*, pages 336–345. 2013a. ISBN 978-3-642-40494-8.

H. A. Le Thi, M. Le Hoai, T. Pham Dinh, and V.N. Huynh. Binary classification via spherical separator by dc programming and dca. *J. Global Optimization*, 56(4):1393–1407, 2013b.

H. A. Le Thi, T. B. T. Nguyen, and M. Le Hoai. Sparse signal recovery by difference of convex functions algorithms. In *Intelligent Information and Database Systems*, volume 7803 of *Lecture Notes in Computer Science*, pages 387–397. 2013c. ISBN 978-3-642-36542-3.

H. A. Le Thi, D. Q. Tran, and H. A. Kondo. A difference of convex functions algorithm for optimal scheduling and real-time assignment of preventive maintenance jobs on parallel processors. *Journal of Industrial and Management Optimization (JIMO)*, pages 1–20, 2013d. ISSN 0925-5001.

H. A. Le Thi, X. T. Vo, and T. Pham Dinh. Robust feature selection for svms under uncertain data. In *Advances in Data Mining. Applications and Theoretical Aspects*, volume 7987 of *Lecture Notes in Computer Science*, pages 151–165. 2013e. ISBN 978-3-642-39735-6.

H. A. Le Thi, P. Damel, P. Nadège, and T. P. Nguyen. The confrontation of two clustering methods in portfolio management: Ward's method versus dca method. In *Advanced Computational Methods for Knowledge Engineering*, volume 282 of *Advances in Intelligent Systems and Computing*, pages 87–98. 2014a. ISBN 978-3-319-06568-7.

H. A. Le Thi, A. V. Le, X. T. Vo, and Z. Ahmed. A filter based feature selection approach in msvm using dca and its application in network intrusion detection. In *Intelligent Information and Database Systems*, volume 8398 of *Lecture Notes in Computer Science*, pages 403–413. 2014b. ISBN 978-3-319-05457-5.

H. A. Le Thi, M. Le Hoai, and T. Pham Dinh. New and efficient dca based algorithms for minimum sum-of-squares clustering. *Pattern Recognition*, 47(1):388–401, 2014c.

H. A. Le Thi, T. Pham Dinh, M. Le Hoai, and Vo X. T. Dc approximation approaches for sparse optimization. *European Journal of Operational Research*, 2014d.

H.A. Le Thi and M. Moeini. Portfolio selection under buy-in threshold constraints using dc programming and dca. In *Service Systems and Service Management, 2006 International Conference on*, volume 1, pages 296–300, Oct 2006.

H. A. Le Thi (home page). Dc programming and dca. http://lita.sciences.univ-metz.fr/~lethi.

H. Lee, A. Battle, R. Raina, and A. Y Ng. Efficient sparse coding algorithms. *Advances in Neural Information Processing Systems*, 19:801–808, 2007.

Stan Z. Li. *Markov Random Field Modeling in Image Analysis*. Springer Publishing Company, Incorporated, 3rd edition, 2009. ISBN 9781848002784.

J. Liang, M. Fadili, and G. Peyre. Local linear convergence of forward–backward under partial smoothness. *Technical report, arxiv preprint arXiv:1407.5611, 2014*, 2014.

Qingmin Liao and Yingying Deng. An accurate segmentation method for white blood cell images. In *IEEE Intl Sym on Biomedical Imaging*, pages 245–248. IEEE, 2002.

Benjamin F. Jr Logan. *Properties of high-pass signals*. PhD thesis, 1965. PhD thesis, Columbia University.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *J. Mach. Learn. Res.*, 11:19–60, March 2010. ISSN 1532-4435.

S.G. Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *Trans. Sig. Proc.*, 41 (12):3397–3415, December 1993. ISSN 1053-587X.

D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

G. Matheron. *Random Sets and Integral Geometry*. Wiley Series in Probability and Mathematical Statistics. Wiley, 1974.

Image. Medical (home page). Foodmedical image database. http://foodmedicaleponyms.blogspot.fr/2011/03/fried-egg-like-cells.htm. Accessed on March 2014.

G. H. Mohimani, M. Babaie-Zadeh, and C. Jutten. Fast sparse representation based on smoothed l0 norm. In *7th International Conference on Independent Component Analysis and Signal Separation (ICA2007)*, volume 4666 of *Lecture Notes in Computer Science*, pages 389–396. Springer, 2007. ISBN 978-3-540-74493-1.

G. H. Mohimani, M. Babaie-Zadeh, and C. Jutten. A fast approach for overcomplete sparse decomposition based on smoothed l0 norm. *IEEE Transactions on Signal Processing*, 57(1):289–301, January 2009. ISSN 1053-587X.

Mehryar Mohri and Andres Muñoz Medina. Learning theory and algorithms for revenue optimization in second-price auctions with reserve. In *Proceedings of the 31st International Conference on Machine Learning, Beijing, China. JMLR: W&CP*, volume 32, 2014.

M. Mokhtar, A. Shuib, and D. Mohamad. Mathematical programming models for portfolio optimization problem: A review. *International Journal of Social, Human Science and Engineering*, 8(2):76 – 83, 2014. ISSN 1307-6892.

Database. Nanotechweb (home page). Nanotechweb image database. http://nanotechweb.org/cws/article/tech/33794. Accessed on March 2014.

B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM J. Comput.*, 24(2), April 1995. ISSN 0097-5397.

B. Ndiaye, T. Pham Dinh, and H. A. Le Thi. Single straddle carrier routing problem in port container terminals: Mathematical model and solving approaches. In *International Journal of Intelligent Information and Database Systems IJIIDS*, volume 14, pages 21–31. 2008. ISBN 978-3-540-87476-8.

D. Needell and J. A. Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Appl. Comput. Harmon. Anal.*, 26:301–321, 2009.

C. N. Nguyen, H. A. Le Thi, and T. Pham Dinh. A branch and bound algorithm based on dc programming and dca for strategic capacity planning in supply chain design for a new market opportunity. In Karl-Heinz Waldmann and UlrikeM. Stocker, editors, *Operations Research Proceedings 2006*, volume 2006 of *Operations Research Proceedings*, pages 515–520. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-69994-1.

D. M. Nguyen. *La programmation DC et la méthode Cross-Entropy pour certaines classes de problèmes en finance, affectation et recherche d'informations : codes et simulations numériques*. PhD thesis, LMI, Ecole doctorale Sciences Physiques Mathématiques et de l'Information pour l'ingénieur , INSA–Rouen, 2012.

D. M. Nguyen, H. A. Le Thi, and T. Pham Dinh. A cross-entropy method for value-at-risk constrained optimization. In *Intelligent Information and Database Systems*, volume 6592 of *Lecture Notes in Computer Science*, pages 442–451. 2011. ISBN 978-3-642-20041-0.

Q. T. Nguyen and H. A. Le Thi. Solving an inventory routing problem in supply chain by dc programming and dca. In *Intelligent Information and Database Systems*, volume 6592 of *Lecture Notes in Computer Science*, pages 432–441. 2011. ISBN 978-3-642-20041-0.

B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: a strategy employed by v1? *Vision Research*, 37:3311–3325, 1997.

N.R. Pal and S.K. Pal. A review on image segmentation techniques. *Pattern Recognition*, 26(9):1277–1294, 1993.

Wei Pan, Xiaotong Shen, and Binghui Liu. Cluster analysis: unsupervised learning via supervised learning with a non-convex penalty. *Journal of Machine Learning Research*, 14(1):1865–1889, 2013.

D.K. Panjwani and G. Healey. Unsupervised segmentation of textured colour images using markov random field models. *Pattern Analysis and Machine Intelligence, IEEE Transactions*, 17:939–954, 1993. ISSN 0031-3203.

Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad. Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. *Conference Record of The Twenty-Seventh Asilomar Conference on Signals, Systems and Computers, 1993.*, 1:40–44, 1993.

D. Peleg and R. Meir. A bilinear formulation for vector sparsity optimization. *Signal Processing*, 88(2):375 – 389, 2008. ISSN 0165-1684.

V. N. Pham. *Programmation DC et DCA pour l'optimisation non convexe/optimisation globale en variables mixtes entières. Codes et Applications.* PhD thesis, LMI, Ecole doctorale Sciences Physiques Mathématiques et de l'Information pour l'ingénieur , INSA–Rouen, 2013.

V. N. Pham, H. A. Le Thi, and T. Pham Dinh. A dc programming framework for portfolio selection by minimizing the transaction costs. In *Advanced Computational Methods for Knowledge Engineering*, volume 479 of *Studies in Computational Intelligence*, pages 31–40. 2013. ISBN 978-3-319-00292-7.

T. Pham Dinh. Elements homoduaux relatifs à un couple de normes (φ, ψ). applications au calcul de $s_{\varphi\psi}(a)$. Technical report, Grenoble, 1975.

T. Pham Dinh. Calcul du maximum d'une forme quadratique définie positive sur la boule unité de la norme du max. Technical report, Grenoble, 1976.

T. Pham Dinh. *Algorithms for solving a class of non convex optimization problems. Methods of subgradients*, volume 129 of *North-Holland Mathematics Studies*. Elsevier Science Publishers, 1986.

T. Pham Dinh and S. E. Bernoussi. Duality in d. c. (difference of convex functions) optimization. Subgradient methods. Trends in mathematical optimization, 4th French-German Conference, Irsee/FRG 1986, ISNM 84, 277-293, 1988.

T. Pham Dinh and H. A. Le Thi. Convex analysis approach to d.c. programming: theory, algorithms and applications. *Acta Mathematica Vietnamica*, 22(1):289–355, 1997. ISSN 0251-4184.

T. Pham Dinh and H. A. Le Thi. Recent advances in dc programming and dca. *Transactions on Computational Collective Intelligence*, 8342:1–37, 2014.

T. Pham Dinh and H.A. Le Thi. Dc optimization algorithms for solving the trust region subproblem. *SIAM Journal of Optimization*, 8(2):476–505, 1998.

T. Pham Dinh, C. N. Nguyen, and H. A. Le Thi. Dc programming and dca for globally solving the value-at-risk. *Computational Management Science*, 6(4):477–501, 2009. ISSN 1619-697X.

T. Pham Dinh, V. N. Pham, and H. A. Le Thi. Dc programming and dca for portfolio optimization with linear and fixed transaction costs. In *Intelligent Information and Database Systems*, volume 8398 of *Lecture Notes in Computer Science*, pages 392–402. 2014. ISBN 978-3-319-05457-5.

D. N. Phan, M. C. Nguyen, and H. A. Le Thi. A dc programming approach for sparse linear discriminant analysis. In *Advanced Computational Methods for Knowledge Engineering*, volume 282 of *Advances in Intelligent Systems and Computing*, pages 65–74. 2014. ISBN 978-3-319-06568-7.

R. Prony. Essai expérimental et analytique sur les lois de la dilatabilité des uides élastique et sur celles de la force expansive de la vapeur de leau et de la vapeur de lalkool, à différentes températures. *J. École Polytechnique*, 1, 1975. ISSN 24-76.

B.D. Rao and K. Kreutz-Delgado. An affine scaling methodology for best basis selection. *IEEE Trans. Signal Processing*, 47:87–200, 1999.

N. Rao and F. Porikli. A clustering approach to optimize online dictionary learning. In *ICASSP*, pages 1293–1296. IEEE, 2012. ISBN 978-1-4673-0046-9.

F. Rinaldi. *Mathematical Programming Methods for minimizing the zero norm over polyhedral sets*. PhD thesis, Dipartimento di Statistica, Probabilità e Statistiche Applicate, Sapienza, Università di Roma, 2009.

F. Rinaldi. Concave programming for finding sparse solutions to problems with convex constraints. *Optimization Methods and Software*, 26:6:971–992, 2011.

F. Rinaldi, F. Schoen, and M. Sciandrone. Concave programming for minimizing the zero-norm over polyhedral sets. *Comput. Opt. Appl.*, 46(3):467–486, 2010.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D*, 60(1-4):259–268, November 1992. ISSN 0167-2789.

Fadil Santosa and William W. Symes. Linear Inversion of Band-Limited Reflection Seismograms. *SIAM Journal on Scientific and Statistical Computing*, 7(4):1307–1330, 1986.

Peter Schmieder, Alan S. Stern, Gerhard Wagner, and Jeffrey C. Hoch. Application of nonlinear sampling schemes to COSY-type spectra. *Journal of Biomolecular NMR*, 3(5):559–576, 1993.

Christoph Schnörr. Signal and image approximation with level-set constraints. *Computing*, 81(2-3):137–160, 2007.

Jean Serra. *Image Analysis and Mathematical Morphology*. Academic Press, Inc., Orlando, FL, USA, 1983. ISBN 0126372403.

Khang Siang Tan and Nor Ashidi Mat Isa. Color image segmentation using histogram thresholding - fuzzy c-means hybrid approach. *Pattern Recogn.*, 44(1):1–15, January 2011. ISSN 0031-3203.

USC. SIPI Image Database. The usc-sipi image database. http://sipi.usc.edu/database/. Accessed on March 2014.

Wladyslaw Skarbek and Andreas Koschan. Colour image segmentation: A survey, 1994a.

Wladyslaw Skarbek and Andreas Koschan. Colour image segmentation: A survey, 1994b.

K. Skretting and K. Engan. Recursive least squares dictionary learning algorithm. *Signal Processing, IEEE Transactions on*, 58(4):2121–2130, April 2010. ISSN 1053-587X.

ToolBox Sparco (home page). Sparco: A toolbox for testing sparse reconstruction algorithms. http://www.cs.ubc.ca/labs/scl/sparco/. Accessed on March 2014.

A. S. Ta. *Programmation DC et DCA pour la résolution de certaines classes des problèmes dans les systèmes de transport et de communication*. PhD thesis, LMI, Ecole doctorale Sciences Physiques Mathématiques et de l'Information pour l'ingénieur , INSA–Rouen, 2012.

A. S. Ta, H. A. Le Thi, D. Khadui, and T. Pham Dinh. Solving qos routing problems by dca. In *Intelligent Information and Database Systems*, volume 5991 of *Lecture Notes in Computer Science*, pages 460–470. 2010a. ISBN 978-3-642-12100-5.

A. S. Ta, H. A. Le Thi, D. Khadui, and T. Pham Dinh. Solving multicast qos routing problem in the context v2i communication services using dca. In *Computer and Information Science (ICIS), 2010 IEEE/ACIS 9th International Conference, Yamagata, Japan*, pages 471–476, Aug 2010b.

A. S. Ta, H. A. Le Thi, D. Khadui, and T. Pham Dinh. Solving partitioning-hub location-routing problem using dca. *Journal of Industrial and Management Optimization*, 8(1):87–102, 2012a.

A. S. Ta, T. Pham Dinh, H. A. Le Thi, and D. Khadui. Solving many to many multicast qos routing problem using dca and proximal decomposition technique. In *International Conference on Computing, Networking and Communications (ICNC2012), Hawaii, American*, pages 809–814, Jan 2012b.

M. T. Ta. *Techniques d'optimisation non convexe basée sur la programmation DC et DCA et méthodes évolutives pour la classification non supervisée*. PhD thesis, LITA, IAEM, Université de Lorraine, 2014.

H.L. Taylor, S.C. Banks, and J.F. McCoy. Deconvolution with the ℓ_1 norm. *Geophysics*, 44(1):39–52, 1979.

R.I. Taylor and P.H. Lewis. Colour image segmentation using boundary relaxation. pages III:721–724, 1992.

Nipon Theera-Umpon. White blood cell segmentation and classification in microscopic bone marrow images. In *Proceedings of the Second International Conference on Fuzzy Systems and Knowledge Discovery - Volume Part II*, volume 3614 of *FSKD'05*, pages 787–796, 2005. ISBN 3-540-28331-5, 978-3-540-28331-7.

Sergios Theodoridis, Yannis Kopsinis, and Konstantinos Slavakis. Sparsity-aware learning and compressed sensing: An overview. *arXiv*, 1211.5231, 2012.

Mamadou Thiao, T. Pham Dinh, and H. A. Le Thi. Dc programming approach for a class of nonconvex programs involving ℓ_0 norm. *Communications in Computer and Information Science, Springer*, 14:348–357, 2008.

Mamadou Thiao, T. Pham Dinh, and H. A. Le Thi. A dc programming approach for sparse eigenvalue problem. In *Internationale Conference on Machine learninh ICML 2010*, pages 1063–1070, 2010.

Xilan Tian, Gilles Gasso, and Stéphane Canu. A multiple kernel framework for inductive semi-supervised svm learning. *Neurocomputing*, 90(0):46 – 58, 2012. ISSN 0925-2312. Advances in artificial neural networks, machine learning, and computational intelligence (ESANN 2011).

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58(1):267–288, 1996.

J. F. Toland. Direct calculation of the information matrix via the EM algorithm. *Journal of Mathematical Analysis and Applications*, 66:399–415, 1978.

J. B. H. Urruty. Generalized differentiability duality and optimization for problem dealing with differences of convex functions. *Lecture Notes in Economics and Mathematical Systems, volume*, 256:260–277, 1986.

J. Verge Llahi. *Color Constancy and Image Segmentation Techniques for Applications to Mobile Robotics*. PhD thesis, Universitat Politècnica de Catalunya, 2005.

Luc Vincent and Pierre Soille. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(6):583–598, June 1991. ISSN 0162-8828.

Nikola Vucic, Shuying Shi, and Martin Schubert. Dc programming approach for resource allocation in wireless networks. In *WiOpt*, pages 380–386, 2010.

S. Osher W. Yin, D. Goldfarb, and J. Darbon. Bregman iterative algorithms for ℓ_1 minimization with applications to compressed sensing. *SIAM Journal on Imaging Sciences*, 1:143–168, 2008.

Junhui Wang, Xiaotong Shen, and Wei Pan. On efficient large margin semisupervised learning: Method and theory. *Journal of Machine Learning Research*, 10:719–742, 2009.

Kuaini Wang, Ping Zhong, and Yaohong Zhao. Training robust support vector regression via d. c. program. *Journal of Information & Computational Science*, 7(12):2385–2394, 2010.

Stefan Weber, Thomas Schüle, Attila Kuba, and Christoph Schnörr. Binary tomography with deblurring. In Ralf Reulke, Ulrich Eckardt, Boris Flach, Uwe Knauer, and Konrad Polthier, editors, *Combinatorial Image Analysis*, volume 4040 of *Lecture Notes in Computer Science*, pages 375–388. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-35153-5.

Jason Weston, André Elisseeff, Bernhard Schölkopf, and Mike Tipping. Use of the zero norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, March 2003. ISSN 1532-4435.

L.B. Wolff. Diffuse-reflectance model for smooth dielectric surfaces. 11(11):2956–2968, November 1994.

Changzhi Wu, Chaojie Li, and Qiang Long. A dc programming approach for sensor network localization with uncertainties in anchor positions. *Journal of Industrial and Management Optimization*, 10(3):817 – 826, 2014.

Liming Yang and Laisheng Wang. A class of semi-supervised support vector machines by dc programming. *Adv. Data Analysis and Classification*, 7(4):417–433, 2013.

P. Yin, Y. Lou, Q. He, and J. Xin. Minimizaiton of $\ell_1 - \ell_2$ for compressed sensing. *CAM-report 14-01, UCLA*, 2014.

Yiming Ying, Kaizhu Huang, and Colin Campbell. Enhanced protein fold recognition through a novel data integration approach. *BMC Bioinformatics*, 10:267, 2009.

S. Zhang, Y. Zhan, Y. Zhou, M. Uzunbas, and D. N. Metaxas. Shape prior modeling using sparse representation and online dictionary learning. In *MICCAI (3)*, pages 435–442, 2012.

T. Zhang. Some sharp performance bounds for least squares regression with regularization. *Ann. Statist.*, 37:2109–2144, 2009.

Y. Zhao and D. Li. Reweighted ℓ_1-minimization for sparse solutions to underdetermined linear systems. *SIAM J. Opt.*, 22, no.3:1065–1088, 2012.

Yiqing Zhong and El Houssaine Aghezzaf. A dc programming approach to solve the single-vehicle inventory routing problem. In *Proceedings of the international conference CIE39*, 2009.

Yiqing Zhong and El Houssaine Aghezzaf. Combining dc-programming and steepest-descent to solve the single-vehicle inventory routing problem. *Computers & Industrial Engineering*, 61(2):313–321, 2011.

H. Zou. The adaptive lasso and its oracle properties. *J. Amer. Stat. Ass.*, 101:1418–1429, 2006.

H. Zou and R. Li. One-step sparse estimates in nonconcave penalized likelihood models. *The Annals of Statistics*, vol. 36, no. 4:1509–1533, 2008.