DESCRIPTION DE L'UTILISATION DE UNLOCBOX

CONTEXTE ET MOTIVATION

L'optimisation convexe est un outil essentiel pour l'apprentissage automatique, car nombre de ses problèmes peuvent être formulés comme des problèmes de minimisation de fonctions objectives spécifiques. Bien qu'il existe une grande variété d'algorithmes pour résoudre les problèmes convexes, il est de plus en plus important de privilégier des méthodes efficaces et évolutives, capables de traiter le big data. Lorsque la fonction objective peut être écrite comme une somme de termes « simples », les méthodes de séparation proximale constituent un choix judicieux. UNLocBoX est une bibliothèque MATLAB qui implémente nombre de ces méthodes et qui est conçue pour résoudre des problèmes d'optimisation convexe de la forme $\min_{x \in \mathbb{R}^N} \sum_{n=1}^{K} f_n(x).$

QU’EST CE QUE UNLOCBOX

UNLocBoX est une boîte à outils d'optimisation convexe permettant de résoudre des problèmes de la forme (2) avec MATLAB. Elle se concentre plus particulièrement sur les méthodes de découpage proximal.Elle intègre les solveurs les plus récents tels que FISTA, DouglasRachford, SDMM ainsi que des techniques duales primales telles que ChambollePock et les méthodes forwardbackwardforward. Elle inclut également une liste exhaustive d'opérateurs proximaux courants, combinables entre eux, permettant une implémentation rapide d'une grande variété de problèmes convexes

ARCHITECTURE ET COMPOSANTS CLES

- Solveurs : le cœur de la boîte à outils. Ils contiennent la plupart des techniques récentes, comme les algorithmes FISTA(forwardbackward), DouglasRachford, PPXA, ADMM, SDMM et autres.
- Opérateurs proximaux : de nombreux opérateurs proximaux prédéfinis aident l'utilisateur à résoudre rapidement de nombreux problèmes courants. La boîte à outils comprend également des opérateurs de projection pour gérer les contraintes strictes.
- Fichiers de démonstration : aident l’utilisateur à démarrer avec la boîte à outils.
- Exemples de signaux : principalement utilisés dans les fichiers de démonstration.
- Fonctions utilitaires : quelques fonctions utiles.

INSTALLATION ET PRISE EN MAIN

Télécharger les Fichiers

- Obtenez le Code : Rendez-vous sur la page GitHub du projet (le site où le code est stocké).
- Téléchargez l'Archive : Cherchez le bouton pour "Télécharger le ZIP" (Download ZIP) et enregistrez l'archive compressée sur votre ordinateur.
- Décompressez : Extrayez le contenu de ce fichier ZIP. Vous obtiendrez un dossier principal rempli d'autres sous-dossiers (prox, solver, demos, etc.).

Ouvrir dans MATLAB

- Lancez MATLAB.
- Naviguez : Dans la fenêtre "Répertoire Courant" (Current Folder) de MATLAB, accédez au dossier que vous venez de décompresser.

Initialiser la Boîte à Outils

- Exécutez le Script : Dans la Fenêtre de Commande (Command Window) de MATLAB, tapez la commande (**init_unlocbox**) et appuyez sur Entrée :
- Vérifiez (Optionnel) : Pour être sûr que tout fonctionne, vous pouvez lancer une démonstration :(**demo_unlocbox**)
- Si une simulation ou une figure apparaît, c'est que l'installation a réussi

CHOIX DU SOLVER

En général, un solveur prend trois types d'entrées : un point d'initialisation x0, les fonctions à minimiser et une structure optionnelle de paramètres. En règle générale, dans UNLocBoX, nous utilisons la convention suivante. Le point d'initialisation est toujours le premier argument (sauf pour SDMM), puis viennent les fonctions et enfin la structure optionnelle des paramètres.

La boîte à outils UNLocBoX est composée de plusieurs solveurs. Nous les classons en deux groupes distincts.

- Premièrement, il existe des solveurs spécifiques qui minimisent seulement deux fonctions (Forward backward, Douglas Rachford, ADMM, etc.). Ceux-ci sont généralement plus efficaces.
- Deuxièmement, il existe des solveurs généraux, plus généraux mais moins efficaces (Generalized forward backward, PPXA, SDMM, etc.). Tous les solveurs possibles ne sont pas inclus dans UNLocBoX. Cependant, celui-ci offre un cadre général permettant d'ajouter vos propres solveurs.

FONCTIONS DANS UNLOCBOX

1-Définition d’une fonction

Chaque fonction fk(x) est modélisée comme une structure contenant différents champs : eval, prox, grad, beta. une fonction f, le champ f.eval est une fonction Matlab handle qui prend en entrée les variables d'optimisation x et renvoie la valeur f(x). Le champ f.prox est alors un autre handle de fonction prenant en entrée un vecteur x, avec un nombre réel positif τ et renvoie le vecteur proxτ f

2- sélection d’un solveur

L'UNLocBoX contient une fonction de résolution générale appelée solvep. Elle peut être utilisée pour de nombreux problèmes. aveuglément. Cette fonction calculera le pas de temps pour vous et sélectionnera un algorithme compatible avec votre Problème. Pour minimiser la somme des fonctions f1, f2, f3, écrivez dans MATLAB

PRINCIPAUX PROX-OPERATEURS DISPONIBLES

- Définition des opérateurs proximaux

en général, un opérateur proximal d'une fonction convexe définie semi-continue inférieure f_i de $\mathbb{R}^N$ a $(-\infty, +\infty]$ est définie comme suit :

$$\operatorname{prox}_f(x) := \underset{y \in \mathbb{R}^N}{\arg\min} \left\{ \frac{1}{2}\|x - y\|_2^2 + f(y) \right\}$$

Si la fonction f n'est pas différentiable, elle doit être minimisée par son opérateur proximal. Pour une fonction f inférieure semi-continue, on les définit comme suit :

$$\operatorname{prox}_{\lambda f}(x) := \underset{y \in \mathbb{R}^N}{\arg\min} \left(\frac{1}{2}\|x - y\|_2^2 + \lambda f(y) \right)$$

Afin de faciliter la définition des fonctions et d'accélérer les implémentations, UNLocBoX inclut divers opérateurs proximaux. Ces opérateurs sont définis comme des fonctions MATLAB prenant trois paramètres d'entrée : x, lambda et param. x représente le signal initial, c'est-à-dire le signal de l'itération courante, avant application de l'opérateur proximal. lambda représente ensuite le poids de la fonction objectif, multiplié par le pas de calcul requis par chaque algorithme. Enfin, param est une structure MATLAB contenant un ensemble de paramètres optionnels.

ASTUCES ET BONNES PRATIQUES

Définir les Opérateurs Nécessaires : Chaque fonction fi doit inclure les champs nécessaires à la méthode de résolution choisie.

- Pour une fonction non-différentiable (comme la norme l1 pour la sparsity), vous devez définir son opérateur proximal : f.prox.
- Pour une fonction différentiable (ou lisse), vous devez définir son gradient : f.grad et la constante de Lipschitz de son gradient : f.beta.
- Astuce : Utilisez des fonctions anonymes (@(x)) pour définir f.eval, f.prox et f.grad. C'est plus propre et rapide que de créer un fichier .m séparé pour chaque petite fonction.
- Champ f.eval : Incluez toujours le champ f.eval (l'évaluation de la fonction) même si le solveur ne l'utilise pas pour l'itération. Cela permet de suivre la convergence et de vérifier que le minimum trouvé est cohérent.

Utiliser l'Option verbose : Définissez param.verbose = 1 (ou plus) pour afficher l'évolution de la fonction objectif à chaque itération. C'est l'outil principal pour diagnostiquer si l'algorithme converge, stagne ou diverge.

- Gérer le Critère d'Arrêt (param.tol et param.maxit) :
- param.tol : La tolérance d'arrêt. Fixez-la à une valeur plus lâche (1e-4) pour un prototypage rapide et plus stricte (1e-6 ou 1e-8) pour le résultat final.
- param.maxit : Le nombre maximal d'itérations. Fixez-le pour éviter les boucles infinies en cas de divergence ou de stagnation

INTEGRATION AVEC D’AUTRES TOOLBOXES

1. Intégration avec l'Optimization Toolbox

Bien qu'UNLocBoX et l'Optimization Toolbox de MathWorks offrent des fonctionnalités d'optimisation, leur intégration est indirecte.

2. intégration avec l’image processing toolbox ou le traitement du signal

UNLocBoX est souvent utilisé pour le débruitage, la déconvolution et la reconstruction d'images/signaux

3. Intégration avec la Parallel Computing Toolbox

L'optimisation, en particulier les méthodes itératives, peut être accélérée par la parallélisation.

LIMITES ET POINTS A NOTER

1. Limite Fondamentale : La Convexité

La limite la plus critique et fondamentale d'UNLocBoX est son objectif : la résolution de problèmes d'optimisation convexe non-lisse (non-smooth convex optimization).

2. Limites Algorithmiques et de Performance

Bien que les méthodes proximales soient puissantes, elles ont des contraintes de performance et de conception spécifiques.

3. Limites de Portée et de Flexibilité

UNLocBoX n'est pas un solveur d'optimisation généraliste comme fmincon ou quadprog de l'Optimization Toolbox.

CAS D'USAGES ET APPLICATIONS

UNLocBoX est particulièrement puissant dans des domaines où l'on cherche à retrouver des informations à partir de données bruitées, incomplètes ou de faible qualité. Ses applications se concentrent principalement sur les problèmes inverses, tirant parti de la régularisation non-lisse pour obtenir des solutions avec des propriétés désirées (comme la parcimonie, la faible rang, ou la variation totale).

- **1. Traitement et Reconstruction d'Images**

  C'est le domaine d'application le plus courant et le plus illustratif des méthodes proximales

- **2. Traitement du Signal et Acquisition Compressive (Compressed Sensing)**

  UNLocBoX est idéal pour les problèmes impliquant la parcimonie (sparsity)

- **3. Apprentissage Automatique et Statistiques**

  L'optimisation convexe est la clé de voûte de nombreux modèles d'apprentissage

- **4. Problèmes Inverses Généraux**

  Chaque fois qu'une observation y est liée à une cause x par une transformation A (où A est une matrice ou un opérateur linéaire), UNLocBoX peut s'appliquer si le problème est convexe

DEMONSTRATION

```
%% Initialisation
clear;
close all;

% Loading toolbox
init_unlocbox;

verbose = 2; % verbosity level


%% Load an image

% Original image
im_original = cameraman;

% Displaying original image
imagesc_gray(im_original, 1, 'Original image');

%% Creation of the problem

sigma_noise = 10/255;
im_noisy = im_original + sigma_noise * randn(size(im_original));

% Create a matrix with randomly 50 % of zeros entry
p = 0.5;
matA = rand(size(im_original));
matA = (matA > (1-p));
% Define the operator
A = @(x) matA .* x;

% Masked image
y = A(im_noisy);

% Displaying the noisy image
imagesc_gray(im_noisy, 2, 'Noisy image');

% Displaying masked image
imagesc_gray(y, 3, 'Measurements');


%% Setting the proximity operator

lambda = 1;
% setting the function f1 (norm TV)
param_tv.verbose = verbose - 1;
param_tv.maxit = 100;
f1.prox = @(x, T) prox_tv(x, lambda*T, param_tv);
f1.eval = @(x) lambda * norm_tv(x);

% setting the function f2
param_proj.epsilon = sqrt(sigma_noise^2 * numel(im_original) * p);
param_proj.A = A;
param_proj.At = A;
param_proj.y = y;
```

RESSOURCES ET DOCUMENTATION

La documentation de l'UNLocBoX est complète, ce qui signifie que chaque fonction est documentée. Vous pouvez le trouver en ligne à l' adresse http://unlocbox.sourceforge.net/doc.

A. Beck et M. Teboulle. Un algorithme itératif rapide de seuillage de rétrécissement pour les problèmes inverses linéaires. Journal SIAM sur les sciences de l'imagerie, 2(1):183–202, 2009.

- [2] PL Combettes. Résolution d'inclusions monotones via des compositions d'opérateurs moyennés non expansifs. Optimisation, 53(56), 2004.
- [3] PL Combettes et JC Pesquet. Une approche par séparation DouglasRachford pour la récupération de signaux variationnels convexes non lisses. Thèmes choisis en traitement du signal, IEEE Journal of, 1(4) : 564–574, 2007.
- [4] PL Combettes et JC Pesquet. Méthodes de séparation proximale en traitement du signal. Algorithme à virgule fixe Rythmes pour les problèmes inverses en sciences et en ingénierie, pages 185–212, 2011.
- [5] PL Combettes et VR Wajs. Récupération du signal par séparation proximale avantarrière. Modélisation et simulation multiéchelles, 4(4) : 1168–1200, 2005.
- [6] J. Douglas et HH Rachford. Sur la résolution numérique des problèmes de conduction thermique à deux et trois variables spatiales. Transactions of the American Mathematical Society, 82(2) : 421–439, 1956.
- [7] J. Eckstein et DP Bertsekas. Sur la méthode de séparation DouglasRachford et l'algorithme du point proximal pour les opérateurs monotones maximaux. Programmation mathématique, 55(1) : 293–318, 1992.
- [8] D. Gabay. Chapitre ix applications de la méthode des multiplicateurs aux inégalités variationnelles. Études en mathématiques et ses applications, 15:299–331, 1983.
- [9] Michael Grant et Stephen Boyd. Cvx : logiciel Matlab pour la programmation convexe disciplinée.
- Chatgtp

CONCLUSION ET PERSPECTIVES

- La boîte à outils UNLocBoX pour MATLAB est un outil puissant et spécialisé, essentiel dans le domaine de l'optimisation convexe non-lisse, en particulier pour la résolution des problèmes inverses régularisés. Elle implémente des algorithmes de pointe basés sur le splitting proximal (tels que FISTA, ADMM, etc.), ce qui la rend incontournable pour les applications nécessitant des solutions structurées (parcimonie, faible rang, variation totale).

- **Perspectives d'Évolution et d'Intégration Futures**

  L'avenir d'UNLocBoX et des méthodes proximales se situe dans la levée des contraintes de convexité et l'amélioration de l'intégration logicielle.