UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA MATEMÁTICA

ALGORITMOS INERCIALES BASADOS EN MULTIPLICADORES PARA RESOLVER PROBLEMAS DE OPTIMIZACIÓN SEPARABLES

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN CIENCIAS DE LA INGENIERÍA, MENCIÓN MATEMÁTICAS APLICADAS
MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL MATEMÁTICO

ROLANDO SEBASTIÁN ROGERS MEDINA

PROFESOR GUÍA:
JUAN PEYPOUQUET

MIEMBROS DE LA COMISIÓN:
JORGE AMAYA
ANDRÉS CONTRERAS

Este trabajo ha sido parcialmente financiado por Proyecto FONDECYT 1181179 y CMM-Conicyt PIA AFB170001

SANTIAGO DE CHILE
2021

RESUMEN DE LA TESIS PARA OPTAR
AL GRADO DE MAGÍSTER EN CIENCIAS DE LA INGENIERÍA,
MENCIÓN MATEMÁTICAS APLICADAS Y MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERO CIVIL MATEMÁTICO
POR: ROLANDO SEBASTIÁN ROGERS MEDINA
FECHA: 2021
PROF. GUÍA: JUAN PEYPOUQUET

ALGORITMOS INERCIALES BASADOS EN MULTIPLICADORES PARA RESOLVER PROBLEMAS DE OPTIMIZACIÓN SEPARABLES

En este trabajo de tesis, enmarcado en el estudio de algoritmos de optimización, se examinarán una familia de algoritmos usados para encontrar puntos sillas del Lagrangiano de un problema de optimización separable con restricciones lineales, estos surgen de aproximaciones del algoritmo Rapid Inertial Proximal Algorithm (RIPA) el cual es usado para encontrar elementos del conjunto $T^{-1}(0)$ donde T es un operador. Esta familia de algoritmos recibe el nombre de RIPA generalizado, donde cada algoritmo perteneciente a esta familia se caracteriza por la forma en que aproxima la implementación del algoritmo RIPA al problema separable.

En particular, dado un algoritmo perteneciente a la familia de RIPA generalizado, donde θ^1 y θ^2 son los errores generados en cada iteración del algoritmo con respecto a la implementación directa de RIPA, se prueba que si los errores θ^{i} convergen a 0 y las sucesiones generadas por el algoritmo no crecen más rápido (en norma) que cierto factor, entonces el algoritmo convergerá. Posterior a esto, se estudiará si estas condiciones son también necesarias para la convergencia del algoritmo.

Finalmente, se presentarán implementaciones numéricas de una variedad de algoritmos pertenecientes a la familia de RIPA generalizado, donde se comparan los pros y contras de cada uno.

Dedicado a mi tata Carlos y yaya Lucía.
Q.E.P.D

Agradecimientos

Quería aprovechar este espacio para partir agradeciendo a dos personas muy importantes en mi vida, Rolando, mi padre, el cual desde pequeño encendió mi interés por las matemáticas y me ha apoyado en todos mis proyectos de vida y Lucía, mi madre, la cual ha llenado mi vida con cariño y amor incondicional. También agradecer a Blanquita, la cual ha sido como una segunda madre para mi, con la cual comparto recuerdos inolvidables de mi infancia. A mi hermano Diego, por ser siempre alguien a quien puedo tener la confianza de hablar todos mis problemas y, junto a Arlette, por darme el privilegio de ser tío. Agradecer a mis abuelos Carlos y Lucía, y toda la Medinada, por hacerme darme cuenta que la familia es lejos el pilar que debemos apreciar más en nuestra vida.

A la Feña, por haberme acompañado de cerca en todo este duro proceso de escribir un tesis, bancandose todos mis reclamos, mañas y siempre dándome ánimo para continuar, estoy seguro que sin ella nada de esto hubiera sido posible. A su familia por ser mi familia adoptiva Santiaguina y ayudarme a no sentirme solo tan lejos de mi ciudad. A mi gato, Amaro, por ser el mejor compañero de oficina y con un solo ronroneo hacerme sentir que estaba todo bien.

Gracias a toda la gente que conocí durante mi paso por el DIM, a mis cabros: Matías, P. , Barri, Rodrigo, Ivan, Pinguino, Choco y Jipi por ser siempre tan apañadores y ser eternos personajes en todas las locuras que viví en la FCFM y DIM, al igual que toda la gente de la 437. A los de generaciones mayores, en especial a Pablo, por haber sido siempre una guía en el paso por esta dura carrera y a la gente de la generación 2016 por ser tan simpáticos y hacer más ameno el departamento. Agradecer también a los amigos que hice en la FCFM, a la gente de Natación Uchile y a Usmash.

Agradezco a dos profesores que fueron clave en mi formación universitaria, a Héctor Ramírez, por haberme dado desde muy temprano en la carrera la oportunidad de formar parte de un cuerpo docente y trabajos de investigación, además de ser un excelente profesor, y a Juan Peypouquet, mi profesor guía, por haberme acompañado desde cerca en el proceso de escritura de este trabajo, ayudándome a resolver los miles de obstáculos que surgen de la escritura de una tesis. Agradecer también a todos los profesores que fueron parte de mi formación y también a todos los alumnos que tuve como profesor auxiliar, haber vivido la docencia de esa forma es una experiencia que jamás olvidaré. A todos los funcionarios del departamento, en especial a Karen, Eterin y Oscar, por siempre tener la disposición de ayudar.

A Coquimbo Unido por ser siempre el más grande de la IV región y enseñarme la pasión y amor eterno a los colores.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Introducción

Un área importante de la Optimización Matemática consiste en los Algoritmos de Optimización, donde el principal objetivo de esta rama es encontrar algoritmos que converjan a soluciones de un problema de optimización, esto es, llevar algo de la abstracción matemática a la implementacion, lo cual tiene amplias aplicaciones en la industria.

Una de estas aplicaciones consiste en el tratamiento de imágenes, en la cual suelen aparecer problemas de optimización del tipo

$$\begin{cases} \min\limits_{(x,y)\in X\times Y} & f(x)+g(y) \\ \text{s.a} & Ax-By=0, \end{cases}$$

Donde usualmente el elemento factible $x \in X$ representa la imagen original a tratar y el elemento $y \in Y$ una transformación de la imagen a través de un operador lineal y f,g son funciones convexas no necesariamente diferenciables que representan algún valor que se desea minimizar, sobre la imagen x y su transformación y.

En esta tesis se presentarán una serie de algoritmos utilizados para resolver este tipo de problema, además de estudiar en profundidad uno de ellos y una familia de algoritmos que nace de este.

El algoritmo principal que estudiaremos corresponde al algoritmo (ip-ADMM), una forma de aproximar la aplicación del algoritmo RIPA descrito en [3]. Además presentaremos una forma de extender este proceso de aproximación a través de la familia de algoritmos de RIPA generalizado.

El centro de esta memoria consiste en conseguir condiciones suficientes para la convergencia del algoritmo (ip-ADMM) y la familia RIPA generalizado, además de estudiar algunas de sus propiedades, presentar implementaciones de los algoritmos y comparar con otros algoritmos estudiados para resolver el problema anterior.

La estructura de este texto se divide en tres partes. En primer lugar, un capítulo de preliminares, donde introduciremos herramientas de Análisis convexo y Optimización, las cuales serán necesarias para lograr un entendimiento de los resultados.

En segundo lugar, en el capítulo siguiente se presentará el algoritmo de (ip-ADMM) y una familia más general a la cual este algoritmo pertenece, llamada RIPA generalizado, donde expondremos las condiciones suficientes que tienen que cumplir las sucesiones generadas por

el algoritmo para asegurar su convergencia.

Finalmente, en el tercer capitulo se revelan posibles problemas que puede tener el algoritmo (ip-ADMM) para ser implementado y muestra algunas de las soluciones que ofrece la familia de algoritmos RIPA generalizado. Además, serán presentados los resultados de implementación de estos algoritmos y se compararán con los resultados obtenidos por otros algoritmos.

Capítulo 1

Conceptos preliminares

1.1. Problema de optimización separable

Durante el desarrollo de este trabajo nos enfocaremos en estudiar una familia de algoritmo para resolver un problema específico de optimización.

Consideremos el siguiente problema

$$\begin{cases} \min\limits_{(x,y)\in X\times Y} & f(x)+g(y) \\ \text{s.a} & Ax-By=0, \end{cases} \tag{1.1}$$

donde asumimos que

- X, Y, Z son espacios de Hilbert.
- $A \in \mathcal{L}(X,Z)$ y $B \in \mathcal{L}(Y,Z)$.
- $f : X \to \mathbb{R}\cup\{+\infty\}$ y $g : Y \to \mathbb{R}\cup\{+\infty\}$ son funciones convexas, propias e inferiores semicontinuas.

El Lagrangiano del siguiente problema es

$$\begin{aligned} L : X\times Y\times Z &\to \mathbb{R}\cup\{+\infty\} \\ (x,y,z) &\mapsto f(x)+g(y)+\langle z, Ax-By\rangle \end{aligned}$$

Para resolver (1.1) debemos encontrar un punto silla del Lagrangiano, es decir, debemos resolver el siguiente problema

$$\min_{(x,y)\in X\times Y}\ \max_{z\in Z} L(x,y,z). \tag{1.2}$$

Este problema es equivalente al problema presentado en (1.1) si se satisface dualidad débil, al ser un problema convexo con restricciones lineales podemos asumir que se satisface dualidad débil, esto es discutido con mayor profundidad en [8].

Dada la convexidad de f, g y como $\langle z, Ax-By\rangle$ es una función afín (por ende cóncava), luego, la función L definida anteriormente es una función convexa con respecto al par

(x, y) y cóncava (lineal) con respecto a la variable z. Por ende, obtenemos que (x, y) son soluciones de (1.1) y z es un multiplicador de Lagrange si y solo si (x, y, z) es un punto silla de L.

La condición de optimalidad de primer orden (suficiente y necesaria en este caso) nos dan lo siguiente

$$(x, y, z) \text{ es un punto silla de } L \Leftrightarrow 0 \in (\partial_{x,y} L, -\partial_z L)(x, y, z). \tag{1.3}$$

Donde $(\partial_{x,y} L, -\partial_z L)(x, y, z) = (\partial f(x) + A^t z, \partial g(y) - B^t z, By - Ax)$.

Por lo que si definimos el operador de punto silla $M_P(x, y, z) = (\partial f(x) + A^t z, \partial g(y) - B^t z, By - Ax)$ usando (1.3) obtenemos que

$$(x, y, z) \text{ es un punto silla de } L \Leftrightarrow 0 \in M_P(x, y, z). \tag{1.4}$$

El problema de encontrar una solución $x \in A^{-1}(0)$ ha sido ampliamente estudiado, ahondaremos más en esto en las siguientes secciones.

Para presentar los algoritmos ya existentes usados para resolverlo es necesario presentar algunas definiciones preliminares para el completo entendimiento de estos.

1.2. Nociones de Análisis Convexo y Teoría de Operadores Monótonos.

A lo largo de este capítulo se definirán conceptos y elementos fundamentales para el estudio de la optimización convexa, los cuales han sido principalmente recopilados de [13].
La estructura de este apartado consta de una revisión de conceptos claves para el estudio de funciones convexas, como lo es la teoría de subdiferenciales y sus aplicaciones en el área de optimización.
Partiremos con la noción de convergencia débil y funciones convexas, las cuales serán usadas constantemente durante el desarrollo de este trabajo.

Definición 1.1 *Sea $(H, \langle\cdot\rangle)$ un espacio de Hilbert.*
Una sucesión $(x_n) \subseteq H$ converge débilmente a $\bar{x} \in H$, lo cual denotaremos $x_n \rightharpoonup \bar{x}$ si satisface que

$$\langle y, x_n\rangle \to \langle y, \bar{x}\rangle \ \forall y \in H. \tag{1.5}$$

Definición 1.2 *Sea H un espacio de Hilbert, $x \in H$ y $x^* \in H^*$, diremos que x^* pertenece al subdiferencial de f en x, el cual denotaremos $\partial f(x)$ si*

$$f(y) \geq f(x) + \langle x^*, y - x\rangle \ \forall y \in H.$$

Si $\partial f(x) \neq \emptyset$ diremos que f es subdiferenciable en x.

Además tenemos la siguiente proposición que relaciona el subdiferencial de una función con el concepto de diferencial.

Proposición 1.3 *Sea $f : X \to \mathbb{R} \cup \{+\infty\}$ una función convexa. Si f es Gateaux diferenciable en x, entonces $\partial f(x) = \{\nabla f(x)\}$.*

Definición 1.4 *Sea $f : X \to \mathbb{R} \cup \{+\infty\}$ una función convexa propia y s.c.i, definimos el operador proximal de f en $v \in X$ como*

$$prox_f(v) = \underset{x \in X}{argmin} \left(f(x) + \frac{1}{2}\|x - v\|_2^2 \right)$$

El elemento minimizante del lado derecho es único para esta clase de funciones (dada la convexidad fuerte de la función $f(\cdot)+(1/2)\|\cdot - v\|_2^2$), por lo que el operador está bien definido.

Además el subdiferencial de la función nos da una regla de cálculo para el operador proximal.

Proposición 1.5 *Sea $x = prox_f(v)$, luego x, v satisfacen lo siguiente*

$$0 \in \partial f(x) + x - v$$

, o equivalentemente diremos que

$$x \in (I + \partial f(x))^{-1}(v).$$

También es necesario definir algunos conceptos de teoría de operadores.

Definición 1.6 *Dado H un espacio de Hilbert y un operador $A : H \to 2^H$, diremos que A es maximalmente monótono si satisface lo siguiente*

- $\langle u - v, x - y \rangle \geq 0 \ \forall u \in A(x), v \in A(y), x, y \in H$ *(monótono).*
- $\forall F : H \to 2^H$ *tal que F es monótono y $A(x) \subseteq F(x) \ \forall x \in H$ se debe satisfacer que $F \equiv A$. (maximalidad).*

Definición 1.7 *El resolvente de indice λ de un operador monótono $A : G \to 2^H$ corresponde a $J_{\lambda A} = (I + \lambda A)^{-1}$.*
Con esto podemos definir el siguiente operador

$$A_\lambda = \frac{1}{\lambda}(I - J_{\lambda A}),$$

La cual corresponde a la regularizada de Yosida de indice λ de el operador A.

Definición 1.8 *Sea $A : D \subseteq H \to H$ un operador definido en un abierto en un espacio de Hilbert. Para $\lambda > 0$ diremos que A es λ-cocoerciva si λA es firmemente no expansiva, o sea, satisface lo siguiente*

$$\langle x - y, Ax - Ay \rangle \geq \lambda \|Ax - Ay\|^2. \ \forall x, y \in D. \tag{1.6}$$

Proposición 1.9 *La regularizada A_λ es un operador λ-cocoercivo.*

Algunas de las propiedades importantes que satisface la regularizada de Yosida están descritas en el Capítulo 23 de [6], incluyendo esta última.

1.3. Algoritmos usados para resolver el problema separable.

Existen una serie de algoritmos para resolver el problema descrito en (1.1). Nos concentraremos aquí en los métodos Lagrangianos, los cuales pueden ser acelerados con métodos inerciales, presentaremos los más relevantes para nuestro trabajo.

1.3.1. Ascenso dual y Lagrangiano Aumentado

Para fijar ideas, presentaremos primero una versión simplificada del problema (1.1)

$$\begin{cases} \min\limits_{x \in X} & f(x) \\ \text{s.t.} & Ax = b, \end{cases} \tag{1.7}$$

El Lagrangiano del problema anterior es

$$L(x, z) = f(x) + \langle z, Ax - b \rangle,$$

y la función dual en el sentido de Fenchel-Rockafellar descrita en [6] corresponde a

$$g(z) = \inf_{x \in X} = -f^*(-A^t z) - \langle b, z \rangle,$$

donde z se conoce como el multiplicador de Lagrange, y f^* es la conjugada de Fenchel de f. Como el problema dual corresponde a

$$(D) \max g(z), \tag{1.8}$$

si asumimos que hay dualidad fuerte, es decir, el valor óptimo del problema primal (1.7) es igual al del problema dual (D), podemos obtener la solución primal x^* de un óptimo dual z^* en el caso que exista un único minimizador de $L(x, z^*)$ (por ejemplo, si f es estrictamente convexa) de la siguiente manera

$$x^* = \operatorname*{argmin}_{x \in X} L(x, z^*). \tag{1.9}$$

En otros términos, obtenemos un problema de optimización sin restricciones, el cual es mucho más fácil de resolver.

El método de **Ascenso dual** corresponde a resolver el problema dual (D) con método del gradiente. Es posible probar que $\nabla g(z) = Ax^+ - b$ donde $x^+ = \text{argmin}_x L(x, z)$. Luego, el método de Ascenso dual consiste en el siguiente algoritmo

$$\begin{aligned} x^{k+1} &= \underset{x \in X}{\text{argmin}} L(x, z^k) \\ z^{k+1} &= z^k + \alpha^k (Ax^{k+1} - b). \end{aligned} \tag{1.10}$$

Donde $\alpha_k > 0$ se conoce como el paso. Finalmente para obtener la solución del primal usamos (1.9).
Es importante mencionar que para la correcta definición de este algoritmo es necesaria la existencia de los mínimos presentados en (1.9) y (1.10). Lamentablemente este no existe para cualquier función convexa, basta por ejemplo tomar $z^0 = 0$ y $f(x) = x$ para $x \in \mathbb{R}$. Por lo que es común pedirle ciertas condiciones a la función f como ser estrictamente convexa o coerciva.

Un gran beneficio que tiene usar Ascenso dual en problemas separables como el mencionado (1.1), es que el algoritmo también se puede separar, es decir, actuar sobre cada espacio de forma independiente.
Sea $X = X_1 \times \ldots \times X_n$ un espacio, diremos que una función $f : X \to \mathbb{R}$ es separable si existen $f_1, \cdots, f_n$ funciones tal que

$$f(x) = \sum_{\text{i}=1}^{n} f_{\text{i}}(x_{\text{i}}), \text{ con } x_{\text{i}} \in X_{\text{i}}.$$

Luego un operador $A : X \to Y$, podemos escribirlo como $Ax = \sum_{\text{i}} A_{\text{i}} x_{\text{i}}$, por ende, el Lagrangiano de el problema (1.7) es

$$L(x, z) = \sum_{\text{i}=1}^{n} L_{\text{i}}(x_{\text{i}}, z) = \sum_{\text{i}=1}^{N} \left[f_{\text{i}}(x_{\text{i}}) + \langle z, A_{\text{i}} x_{\text{i}} \rangle - (1/N) \langle z, b \rangle \right],$$

el cual también es separable en X. Esto quiere decir que el cálculo del paso x^{k+1} mencionado en (1.10) puede ser obtenido de forma paralela como N problemas de optimización. El algoritmo (1.10) puede ser escrito de la siguiente manera

$$\begin{aligned} x_{\text{i}}^{k+1} &= \underset{x_{\text{i}} \in X_{\text{i}}}{\text{argmin}} L_{\text{i}}(x_{\text{i}}, z^k), \\ z^{k+1} &= z^k + \alpha^k (Ax^{k+1} - b). \end{aligned} \tag{1.11}$$

Este método recibe el nombre de **descomposición dual**.

Una forma de obtener un algoritmo de mejor convergencia en términos de velocidad a través del método de Ascenso dual consiste en agregar una penalización cuadrática en el Lagrangiano, es decir, el Lagrangiano aumentado es

$$L_\rho(x, z) = f(x) + \langle z, Ax - b \rangle + (\rho/2) \|Ax - b\|_2^2,$$

donde $\rho > 0$ se conoce como un parámetro de penalización.
Notemos que el Lagrangiano aumentado es el Lagrangiano del siguiente problema

$$\begin{cases} \underset{x \in X}{\text{mín}} & f(x) + (\rho/2) \|Ax - b\|_2^2 \\ \text{s.a} & Ax = b, \end{cases} \tag{1.12}$$

el cual es equivalente al problema (1.7). Luego, al aplicar el método de ascenso dual obtenemos el siguiente algoritmo

$$\begin{aligned} x^{k+1} &= \operatorname*{argmin}_{x\in X} L_\rho(x, z^k) \\ z^{k+1} &= z^k + \rho(Ax^{k+1} - b), \end{aligned} \tag{1.13}$$

el cual recibe el nombre de **Método de los multiplicadores**. Este método converge ante condiciones mucho más generales que el método de Ascenso dual sin necesidad de pedir que f sea un función convexa o propia (a diferencia del método de Ascenso Dual que pide estas dos condiciones).

1.3.2. Método de Multiplicadores de direcciones alternantes (ADMM).

Volvamos al problema original (1.1), este algoritmo corresponde a usar el método del Lagrangiano aumentado descrito anteriormente en el espacio producto $X \times Y$, si consideramos

$$L_\lambda(x, y, z) = f(x) + g(y) + \langle z, Ax - By\rangle + (\lambda/2)\|Ax - By\|^2. \tag{1.14}$$

Este corresponde al Lagrangiano del siguiente problema de optimización

$$\begin{cases} \min\limits_{(x,y)\in X\times Y} & f(x) + g(y) + (\lambda/2)\|Ax - By\|^2 \\ \text{s.t.} & Ax - By = 0, \end{cases} \tag{1.15}$$

Notemos que este problema es equivalente a (1.1). Luego, si aplicamos el método de ascenso dual obtenemos el siguiente algoritmo

$$\begin{cases} (x_{k+1}, y_{k+1}) = \operatorname*{argmin}\limits_{(x,y)\in X\times Y} L_\lambda(x, y, z_k) \\ z_{k+1} = z_k + \lambda(Ax_{k+1} - By_{k+1}). \end{cases} \tag{1.16}$$

Aplicar este algoritmo directamente tiene un problema de implementación, pues no podemos calcular directamente las iteraciones x_{k+1} e y_{k+1} por separado, ya que debemos resolver un problema de optimización no separable en el espacio $X \times Y$. Por ende, el algoritmo (ADDM) hace la siguiente modificación a (1.16)

$$(\text{ADMM}) \begin{cases} x_{k+1} = \operatorname*{argmin}\limits_{x\in X} L_\lambda(x, y_k, z_k) \\ y_{k+1} = \operatorname*{argmin}\limits_{y\in Y} L_\lambda(x_{k+1}, y, z_k) \\ z_{k+1} = z_k + \lambda(Ax_{k+1} - By_k + 1). \end{cases} \tag{1.17}$$

De esta forma las variables x, y se calculan de forma alternada, los resultados de convergencia de este algoritmo están descritos en [7].

1.3.3. Algoritmo proximal.

Dado el siguiente problema de optimización

$$\min_{x\in X} f(x),$$

el método del algoritmo proximal corresponde a la siguiente iteración

$$x^{k+1} = \text{prox}_{\lambda f}(x^k).$$

Bajo ciertas condiciones es posible probar que $x_k \to x^*$ donde x^* es un minimizante de f [13].

Este método es útil para todo tipo de funciones convexas, sin importar su suavidad, por lo que sirve para un conjunto mucho más grande de funciones que otros métodos, como el método del gradiente. Si consideramos las condiciones de primer orden para el cálculo del proximal obtenemos que el proximal satisface la siguiente ecuación. Una forma más simple de describir el algoritmo anterior es con la siguiente formulación $x^{k+1} = (I + \lambda\partial f)^{-1}(x^k)$, esto es demostrado en mayor profundidad en [4].

1.3.4. Proximal ADMM (PADMM)

Ahora, si extrapolamos el algoritmo anterior al caso de buscar un punto silla mostrado al inicio del capitulo, si cambiamos f en (1.3.3) por el operador de punto silla $M_P = (\partial_{x,y}L, -\partial_z L)$ obtenemos el siguiente algoritmo proximal

$$(x_{k+1}, y_{k+1}, z_{k+1}) = (I + \lambda M_P)^{-1}(x_k, y_k, z_k)$$

Es decir, $(x_{k+1}, y_{k+1}, z_{k+1})$ deben satisfacer el siguiente sistema de ecuaciones

$$\begin{cases} \frac{1}{\lambda}(x_{k+1} - x_k) + \partial f(x_{k+1}) + A^t(z_k + \lambda(Ax_{k+1} - By_{k+1})) \ni 0 \\ \frac{1}{\lambda}(y_{k+1} - y_k) + \partial g(y_{k+1}) - B^t(z_k + \lambda(Ax_{k+1} - By_{k+1})) \ni 0 \\ \frac{1}{\lambda}(z_{k+1} - z_k) + By_{k+1} - Ax_{k+1} = 0. \end{cases} \tag{1.18}$$

La cual tiene la siguiente formulación variacional

$$\begin{cases} (x_{k+1}, y_{k+1}) = & \text{argmin}_{(\xi,\eta)\in X\times Y}\{f(\xi) + g(\eta) + \langle z_k, A\xi - B\eta\rangle + \frac{\lambda}{2}\|A\xi - B\eta\|^2 \\ & + \frac{1}{2\lambda}\|\xi - x_k\|^2 + \frac{1}{2\lambda}\|\eta - y_k\|^2\} \\ z_{k+1} = & z_k + \lambda(Ax_{k+1} - By_{k+1}). \end{cases} \tag{1.19}$$

Notemos que todos los términos a excepción de $\|A\xi - B\eta\|^2$ hacen que la formulación variacional anterior sea separable, por lo que haciendo lo mismo que hace ADMM sobre el algoritmo (1.16), podemos calcular los términos de manera alternada de la siguiente manera

$$\begin{cases} x_{k+1} = \underset{\xi\in X}{\text{argmin}}\ \{f(\xi) + \langle z_k, A\xi\rangle + \frac{\lambda}{2}\|A\xi - By_k\|^2 + \frac{1}{2\lambda}\|\xi - x_k\|^2\} \\ y_{k+1} = \underset{\eta\in Y}{\text{argmin}}\ \{g(\eta) - \langle z_k, B\eta\rangle + \frac{\lambda}{2}\|Ax_{k+1} - B\eta\|^2 + \frac{1}{2\lambda}\|\eta - y_k\|^2\} \\ z_{k+1} = z_k + \lambda(Ax_{k+1} - By_{k+1}). \end{cases} \tag{1.20}$$

Con esto obtenemos el algoritmo ADMM Proximal

$$(PADMM)\begin{cases} \frac{1}{\lambda}(x_{k+1} - x_k) + \partial f(x_{k+1}) + A^t(z_k + \lambda(Ax_{k+1} - By_K)) \ni 0 \\ \frac{1}{\lambda}(y_{k+1} - y_k) + \partial g(y_{k+1}) - B^t(z_k + \lambda(Ax_{k+1} - By_{k+1})) \ni 0 \\ z_{k+1} = z_k + \lambda(Ax_{k+1} - By_{k+1}). \end{cases} \tag{1.21}$$

La convergencia de este algoritmo fue probada en [5].
Este método es muy similar al algoritmo de Chen-Teboulle de predicción-corrección descrito en [11], el cual es usado para resolver una familia de problemas más acotada que la presentada en (1.1), donde se toma $B = I$, el algoritmo es el siguiente

$$\begin{cases} p_{k+1} &= z_k + \lambda(Ax_k - y_k) \text{ "prediccion"}, \\ x_{k+1} &= \text{argmin} f(x) + \langle p_{k+1}, Ax\rangle + \dfrac{1}{2\lambda}\|x - x_k\|^2, \\ y_{k+1} &= \text{argmin} g(y) + \langle p_{k+1}, y\rangle + \dfrac{1}{2\lambda}\|y - y_k\|^2, \\ z_{k+1} &= z_k + \lambda(Ax_{k+1} - y_{k+1}) \text{ "corrección"} \end{cases} \tag{1.22}$$

Este método tiene cierto parentesco con el método PADMM con dos principales diferencias, este contiene una predicción de la variable dual en el siguiente paso con las variables anteriores y este método es para un problema más simple, lo que resulta más interesante es que la demostración de convergencia de este algoritmo toma en cuenta posibles errores en el calculo de los resolventes (argmin en este caso) mientras no sean muy grandes, esto será algo similar que intentaremos hacer en esta investigación.

1.3.5. Métodos Inerciales.

Dado un operador $T : X \to X$, definimos el siguiente algoritmo

$$\begin{cases} y_k &= x_k + (1 - \frac{\alpha}{k})(x_k - x_{k-1})) \\ x_{k+1} &= (I + sT_{\lambda_k})^{-1}(y_k) \end{cases}$$

Este algoritmo tiene muchísimas similitudes con el algoritmo proximal presentado anteriormente, con la diferencia que agrega un término de extrapolación con los datos anteriores para calcular una variable nueva y_k.
Además si usamos la siguiente propiedad sobre las regularizadas $(T_\lambda)_s = T_{\lambda+s}$ podemos re formular el sistema anterior obteniendo el algoritmo (RIPA) (Rapid Inertial Proximal Algorithm)

$$\text{(RIPA)} \begin{cases} y_k &= x_k + \alpha_k(x_k - x_{k-1}) \\ x_{k+1} &= \dfrac{\lambda_k}{\lambda_k + s_k} y_k + \dfrac{s}{\lambda_k + s_k}(I + (\lambda_k + s)A)^{-1}(y_k). \end{cases} \tag{1.23}$$

En [4] se comprobó que si $\alpha > 2$ y $\lambda_k = (1 + \varepsilon)\dfrac{s}{\alpha^2}k^2$ con $\varepsilon > \dfrac{2}{\alpha - 2}$, luego $x_k \rightharpoonup \hat{x}$ y $y_k \rightharpoonup \hat{x}$ para algún $\hat{x} \in S = A^{-1}(0)$.

Para un mejor entendimiento del trabajo que será presentado a continuación debemos tener en cuenta que este algoritmo al ser usado para resolver el problema (1.1) a través del operador M_P en $X \times Y \times Z$ se escribe de la siguiente manera

$$\text{(RIPA)} \begin{cases} (u_k, v_k, w_k) &= (x_k, y_k, z_k) + \alpha_k((x_k, y_k, z_k) - (x_{k-1}, y_{k-1}, z_{k-1})) \\ (x_{k+1}, y_{k+1}, z_{k+1}) &= \dfrac{\lambda_k}{\lambda_k + s_k} y_k + \dfrac{s}{\lambda_k + s_k}(I + (\lambda_k + s)M_P)^{-1}(u_k, v_k, w_k). \end{cases} \tag{1.24}$$

La aparición de este término extrapolado no solo tiene sentido geométrico, si no que también aparece si consideramos una versión continua del algoritmo anterior, es decir, si consideramos la siguiente ecuación diferencial.

$$\ddot{x}(t) + \gamma\dot{x}(t) + T(x(t)) = 0 \tag{1.25}$$

Donde A es un operador cocoercivo, en [1] y [2] mostraron que es posible probar la convergencia débil de la solución de (1.25) a una solución en $S = T^{-1}(0)$ si el parámetro de cocoercividad λ de A y el coeficiente γ satisfacen $\lambda\gamma^2 > 1$, luego si consideramos que para $\lambda > 0$, la regularizada T_λ es λ-cocoerciva y además se satisface que $T_\lambda^{-1}(0) = T^{-1}(0)$ [4], por lo que podemos deducir que bajo la condición de que $\lambda\gamma^2 > 1$ la solución de la ecuación diferencial

$$\ddot{x}(t) + \gamma\dot{x}(t) + T_\lambda(x(t)) = 0 \tag{1.26}$$

converge débilmente a un elemento en S.
En [4] prueban la convergencia débil sobre una dinámica similar donde los coeficientes γ y λ varían con respecto al tiempo, donde $\gamma(t) = \alpha/t$ con $\alpha > 0$ y se incluye la condición

$$\frac{\alpha}{t}\lambda(t) > 1,$$

es decir, la ecuación diferencial corresponde a

$$\ddot{x}(t) + \frac{\alpha}{t}\dot{x}(t) + T_{\lambda(t)}(x(t)) = 0. \tag{1.27}$$

Ahora, notemos que esta dinámica es suficiente para resolver el problema descrito en (1.1) si calculamos la regularizada del operador M_P, pero este método es continuo pues debemos resolver en tiempo continuo la ecuación. Si consideramos una discretización de la ecuación anterior podemos obtener un algoritmo similar al propuesto en (1.3.5)
Tomando un salto temporal fijo h, y $t_k = kh$, $x_k = x(t_k)$ y $\lambda_k = \lambda(t_k)$, luego con un esquema discreto implícito de la ecuación (1.27) obtenemos

$$\frac{1}{h^2}(x_{k+1} - 2x_k + x_{k-1}) + \frac{\alpha}{kh^2}(x_k - x_{k-1} + A_{\lambda_k}(x_{k+1}) = 0 \tag{1.28}$$

Entonces desarrollando la expresión anterior y tomando $s = h^2$, obtenemos

$$x_{k+1} = \left(I + sA_{\lambda_k}\right)^{-1}\right)^{-1}\left(x_k + \left(1 - \frac{\alpha}{k}\right)(x_k - x_{k-1})\right) \tag{1.29}$$

Capítulo 2

Inertial Proximal ADMM (ip-ADMM) y RIPA generalizado.

En el presente capítulo discutiremos sobre una familia de algoritmos de optimización y un algoritmo particular perteneciente a esta familia. RIPA generalizado corresponde a una familia de algoritmos que son usados para aproximar una implementación del algoritmo RIPA sobre el mismo problema mencionado anteriormente, el algoritmo (ip-ADMM) está contenido en esta familia y es una forma particular de aproximar una implementación de RIPA usando ideas similares de los algoritmos presentados en el capitulo anterior.

El principal enfoque de este capítulo será probar un resultado de convergencia sobre esta familia de algoritmos.

2.1. ip-ADMM

El algoritmo ADMM Inercial Proximal (ip-ADMM) nace como una idea para acelerar el algoritmo (PADMM) del capítulo anterior, para lo cual se agregan términos inerciales como los de RIPA y el término λ pasa a ser variable con cada paso del algoritmo de la forma $\lambda_k + s$. Los términos proximales son calculados de igual forma que PADMM, pero tomando en cuenta el término λ_k, es decir, los términos proximales (p, q, r) de RIPA presentados en (1.19) son aproximados por la siguiente formulación variacional

$$\begin{cases} p = \operatorname{argmin}_{\xi \in X}\{f(\xi) + \langle w, A\xi\rangle + \dfrac{\lambda_k + s}{2}\|A\xi - Bv\|^2 + \dfrac{1}{2(\lambda_k + s)}\|\xi - u\|^2\} \\ q = \operatorname{argmin}_{\eta \in Y}\{g(\eta) - \langle w, B\eta\rangle + \dfrac{\lambda_k + s}{2}\|Ap - B\eta\|^2 + \dfrac{1}{2(\lambda_k + s)}\|\eta - v\|^2\} \\ r = w + (\lambda_k + s)(Ap - Bq), \end{cases} \tag{2.1}$$

es decir, de forma análoga a PADMM, el término $\|A\xi - B\eta\|^2$ es aproximado por $\|A\xi - Bv\|^2$, con esto podemos calcular p antes de q.
Finalmente, aplicando la condición de primer orden a (2.1) y agregando los términos inerciales

de igual manera que RIPA, obtenemos el algoritmo ADDM Inercial Proximal

$$\begin{cases} u_k = x_k + \alpha_k(x_k - x_{k-1}) \\ v_k = y_k + \alpha_k(y_k - y_{k-1}) \\ w_k = z_k + \alpha_k(z_k - z_{k-1}) \\ \dfrac{1}{\lambda_k + s}(p_k - u_k) + \partial f(p_k) + A^t(w_k + (\lambda_k + s)(Ap_k - Bv_k)) \ni 0 \\ \dfrac{1}{\lambda_k + s}(q_k - v_k) + \partial g(q_k) - B^t(w_k + (\lambda_k + s)(Ap_k - Bq_k)) \ni 0 \\ r_k = w_k + (\lambda_k + s)(Ap_k - Bq_k) \\ x_{k+1} = \dfrac{\lambda_k}{\lambda_k + s}u_k + \dfrac{s}{\lambda_k + s}p_k \\ y_{k+1} = \dfrac{\lambda_k}{\lambda_k + s}v_k + \dfrac{s}{\lambda_k + s}q_k \\ z_{k+1} = \dfrac{\lambda_k}{\lambda_k + s}w_k + \dfrac{s}{\lambda_k + s}r_k. \end{cases} \tag{2.2}$$

Donde $\alpha_k = (1 - \frac{\alpha}{k})$ y $\lambda_k \to \infty$.

Notemos que ip-ADMM es una forma particular de aproximar los términos proximales del algoritmo RIPA. En efecto recordando que los términos proximales (p_k, q_k, r_k) calculados por el algoritmo RIPA son soluciones de la siguiente formulación variacional,

$$\begin{cases} (p_k, q_k) = & \operatorname{argmin}_{(\xi,\eta)\in X\times Y}\{f(\xi) + g(\eta) + \langle w_k, A\xi - B\eta\rangle + \dfrac{\lambda}{2}\|A\xi - B\eta\|^2 \\ & + \dfrac{1}{2\lambda}\|\xi - u_k\|^2 + \dfrac{1}{2\lambda}\|\eta - v_k\|^2\} \\ r_{k+1} = & w_k + \lambda(Ap_k - Bq_k). \end{cases} \tag{2.3}$$

Por ende, si calculamos las condiciones de primer orden de este sistema obtenemos lo siguiente

$$\begin{cases} \dfrac{1}{\lambda_k + s}(p_k - u_k) + \partial f(p_k) + A^t(w_k + (\lambda_k + s)(Ap_k - Bq_k)) \ni 0 \\ \dfrac{1}{\lambda_k + s}(q_k - v_k) + \partial g(q_k) - B^t(w_k + (\lambda_k + s)(Ap_k - Bq_k)) \ni 0. \end{cases} \tag{2.4}$$

Este sistema tiene claras similitudes con las ecuaciones presentadas en (2.2) (cambiando un v_k por q_k podemos recuperar el algoritmo (ip-ADMM)). El principal problema que tiene RIPA con ser aplicado directamente a este problema es que el sistema presentado en (2.4) está acoplado y no puede ser resuelto secuencialmente a diferencia de ip-ADMM.

2.2. RIPA generalizado.

En (ip-ADMM) obtuvimos una aproximación de (RIPA) cambiando una variable del sistema de ecuaciones a resolver. Este proceso puede ser bastante intuitivo, pero puede ser generalizado de mejor manera, obteniendo una familia de aproximaciones de este algoritmo con el beneficio que podemos estudiar su comportamiento de manera más general para luego volver al caso particular de (ip-ADMM), por lo que en esta sección presentaremos el algoritmo de RIPA generalizado y presentaremos unos resultados de convergencia.

Primero notemos que (RIPA) puede ser escrito de la siguiente manera

$$(RIPA)\begin{cases} u_k = x_k + \alpha_k(x_k - x_{k-1}) \\ v_k = y_k + \alpha_k(y_k - y_{k-1}) \\ w_k = z_k + \alpha_k(z_k - z_{k-1}) \\ \dfrac{1}{\lambda_k + s}(p_k - u_k) + \partial f(p_k) + A^t r_k \ni 0 \\ \dfrac{1}{\lambda_k + s}(q_k - v_k) + \partial g(q_k) - B^t r_k \ni 0 \\ r_k = w_k + (\lambda_k + s)(Ap_k - Bq_k) \\ x_{k+1} = \dfrac{\lambda_k}{\lambda_k + s}u_k + \dfrac{s}{\lambda_k + s}p_k \\ y_{k+1} = \dfrac{\lambda_k}{\lambda_k + s}v_k + \dfrac{s}{\lambda_k + s}q_k \\ z_{k+1} = \dfrac{\lambda_k}{\lambda_k + s}w_k + \dfrac{s}{\lambda_k + s}r_k. \end{cases} \tag{2.5}$$

Recordando que (ip-ADMM) aproxima r_k en la cuarta línea de (2.5) por el término $w_k + (\lambda_k + s)(Ap_k - Bv_k)$, esto puede ser visto como un caso particular de el siguiente algoritmo generalizado.

$$(RIPA)_{(r^1,r^2)}\begin{cases} u_k = x_k + \alpha_k(x_k - x_{k-1}) \\ v_k = y_k + \alpha_k(y_k - y_{k-1}) \\ w_k = z_k + \alpha_k(z_k - z_{k-1}) \\ \dfrac{1}{\lambda_k + s}(p_k - u_k) + \partial f(p_k) + A^t r_k^1 \ni 0 \\ \dfrac{1}{\lambda_k + s}(q_k - v_k) + \partial g(q_k) - B^t r_k^2 \ni 0 \\ r_k = w_k + (\lambda_k + s)(Ap_k - Bq_k) \\ x_{k+1} = \dfrac{\lambda_k}{\lambda_k + s}u_k + \dfrac{s}{\lambda_k + s}p_k \\ y_{k+1} = \dfrac{\lambda_k}{\lambda_k + s}v_k + \dfrac{s}{\lambda_k + s}q_k \\ z_{k+1} = \dfrac{\lambda_k}{\lambda_k + s}w_k + \dfrac{s}{\lambda_k + s}r_k. \end{cases} \tag{2.6}$$

En el caso de (ip-ADMM) $r_k^1 = w_k + (\lambda_k + s)(Ap_k - Bv_k)$ y $r_k^2 = r_k$, otro ejemplo es una variante "forward-backward" donde aproximamos de la siguiente manera $r_k^1 = w_k + (\lambda_k + s)(Ap_k - Bv_k)$ y $r_k^2 = w_k + (\lambda_k + s)(Au_k - Bq_k)$, esta es una variación más simétrica para aproximar los términos proximales, en el siguiente capítulo estudiaremos similitudes de estos dos algoritmos al aplicarlos numéricamente.

2.2.1. Condiciones de convergencia.

El principal objetivo de este trabajo sera obtener condiciones suficientes para la convergencia a través de subsucesiones de los términos proximales (p_k, q_k, r_k) a una solución del problema original. Durante el desarrollo de esta sección asumiremos que estamos trabajando sobre un espacio de Hilbert de dimensión finita, es decir, isomorfo a $\mathbb{R}^n$, pero todo lo que será presentado a continuación es extendible a dimensión infinita cambiando la convergencia fuerte por convergencia débil.

Lema 2.1 *Si las siguientes condiciones se cumplen*

(i) Las sucesiones (p_k, q_k, r_k) *y* (u_k, v_k, w_k) *son* $o(\lambda_k)$.

(ii) Las sucesiones $r_k^1 - r_k$, $r_k^2 - r_k$ *y* $Ap_k - Bq_k$ *convergen fuertemente a 0 cuando* $k \to +\infty$.

Luego, la sucesión (p_k, q_k, r_k) *generada por* $(RIPA)_{(r_k^1, r_k^2)}$ *satisface que todo punto de acumulación* $(p_\infty, q_\infty, r_\infty)$ *está en* S.

Demostración. Sea $(p_\infty, q_\infty, r_\infty)$ un punto de acumulación de la sucesión (p_k, q_k, r_k), sin pérdida de generalidad asumiremos que $(p_k, q_k, r_k) \to (x_\infty, y_\infty, z_\infty)$.
Luego, notemos que (p_k, q_k, r_k) satisface la siguiente inclusión dada por el algoritmo

$$M_P(p_k, q_k, r_k) \ni \Big(\frac{1}{\lambda_k + s}(p_k - u_k) + A^t(r_k^1 - r_k), \frac{1}{\lambda_k + s}(q_k - v_k) - B^t(r_k^2 - r_k), Bq_k - Ap_k \Big), \quad (2.7)$$

donde $M_P(x, y, z) = (\partial f(x) + A^t z, \partial g(y) - B^t z, By - Ax)$.

Con la condición (i) y (ii) obtenemos que el lado derecho converge fuertemente a 0 cuando $k \to +\infty$ y el lado izquierdo converge a $M_P(x_\infty, y_\infty, z_\infty)$. Luego, como el operador maximalmente monótono M_P tiene su grafo cerrado [9], obtenemos que el punto de acumulación $(p_\infty, q_\infty, r_\infty)$ satisface que $M_P(p_\infty, q_\infty, r_\infty) \ni 0$, es decir, $(p_\infty, q_\infty, r_\infty) \in S$. □

Es importante mencionar que es natural pedir que las secuencias $r_k^1 - r_k$, $r_k^2 - r_k$ converjan fuertemente a 0 , pues estamos diciendo que el error obtenido al aproximar el algoritmo RIPA converja a 0 cuando $k \to +\infty$.
También es razonable pedir que las sucesiones generadas por el algoritmo sean $o(\lambda_k)$, pues si esperamos que las sucesiones converjan a una solución podemos intuitivamente esperar que sean acotadas o a lo más $o(\lambda_k)$.

La única condición que parece fuerte es la condición de que $A_p k - Bq_k \to 0$, en otras palabras, que el par (p_k, q_k) converja a un par factible del problema (1.1). En la siguiente sección demostraremos que esta condición se obtiene de manera directa de las otras condiciones presentes en el Lema 2.1.

Es importante notar que estas condiciones también son satisfechas por el algoritmo RIPA, por lo cual no resulta descabellado pedir estas condiciones para algoritmos que se asemejen mucho a RIPA.

2.2.2. Condiciones para obtener la factibilidad.

Para simplificar al máximo las condiciones suficientes que debemos verificar para asegurar la convergencia de las subsucesiones del algoritmo debemos intentar encontrar inclusiones en estas condiciones, una de las cuales resulta muy importante verificar es la condición sobre el par (p_k, q_k), la cual pide que $Ap_k - Bq_k$ converja fuertemente a 0 cuando $k \to +\infty$.

En un acercamiento para obtener un resultado relacionado con esta condición probaremos la siguiente desigualdad.

Lema 2.2 *Las secuencias generadas por el algoritmo satisfacen la siguiente desigualdad.*

$$\frac{1}{2\tau_k}[\|p_k-u_k\|^2+\|q_k-v_k\|^2+\|p_k-\bar{x}\|^2+\|q_k-\bar{y}\|^2] \leq \frac{1}{2\tau_k}[\|v_k-\bar{y}\|^2+\|u_k-\bar{x}\|^2 + \|w_k-\bar{z}\|^2-\|r_k-\bar{z}\|^2] + \frac{\tau_k}{2}\|Ap_k-Bq_k\|^2+R_k. \quad (2.8)$$

Donde $R_k=\langle r_k^1-w_k, A(\bar{x}-p_k)\rangle+\langle w_k-r_k^2, B(\bar{y}-q_k)\rangle$.

Demostración. Sea $(\bar{x},\bar{y},\bar{z})\in S$ luego usando la monotonía del operador subdiferencial [14] obtenemos que

$$\langle \partial f(\bar{x})-\partial f(p_k), \bar{x}-p_k\rangle \geq 0,$$

Por simplicidad consideremos $\tau_k=\lambda_k+s$, como $-\frac{1}{\tau_k}(p_k-u_k)-A^tr_k^1\in\partial f(p_k)$ y $-A^t\bar{z}\in\partial f(\bar{x})$ obtenemos que

$$\left\langle -A^t\bar{z}+\frac{1}{\tau_k}(p_k-u_k)+A^tr_k^1, \bar{x}-p_k\right\rangle \geq 0$$
$$\Rightarrow \langle r_k^1-\bar{z}, A(\bar{x}-p_k)\rangle \geq \frac{1}{\tau_k}\langle p_k-u_k, p_k-\bar{x}\rangle$$

Por ende, como $\langle p_k-u_k,p_k-\bar{x}\rangle=\frac{1}{2}\left[\|p_k-u_k\|^2+\|p_k-\bar{x}\|^2-\|u_k-\bar{x}\|^2\right]$ obtenemos que

$$\langle r_k^1-\bar{z}, A(\bar{x}-p_k)\rangle \geq \frac{1}{2\tau_k}\left[\|p_k-u_k\|^2+\|p_k-\bar{x}\|^2-\|u_k-\bar{x}\|^2\right]. \quad (2.9)$$

De forma análoga, tenemos la siguiente desigualdad para r_k^2

$$\langle \bar{z}-r_k^2, B(\bar{y}-q_k)\rangle \geq \frac{1}{2\tau_k}\left[\|q_k-v_k\|^2+\|q_k-\bar{y}\|^2-\|v_k-\bar{y}\|^2\right]. \quad (2.10)$$

Sumando ambas desigualdades conseguimos lo siguiente

$$\frac{1}{2\tau_k}[\|p_k-u_k\|^2+\|q_k-v_k\|^2+\|p_k-\bar{x}\|^2+\|q_k-\bar{y}\|^2 \leq \frac{1}{2\tau_k}[\|v_k-\bar{y}\|^2+\|u_k-\bar{x}\|^2]+Q_k, \quad (2.11)$$

Donde

$$\begin{aligned} Q_k &= \langle r_k^1-\bar{z}, A(\bar{x}-p_k)\rangle+\langle \bar{z}-r_k^2, B(\bar{y}-q_k)\rangle \\ &= \underbrace{\langle w_k-\bar{z}, A(\bar{x}-p_k)\rangle+\langle \bar{z}-w_k, B(\bar{y}-q_k)\rangle}_{L_k} + \underbrace{\langle r_k^1-w_k, A(\bar{x}-p_k)\rangle+\langle w_k-r_k^2, B(\bar{y}-q_k)\rangle}_{R_k}. \end{aligned}$$

Teniendo esto recordemos que $A\bar{x}-B\bar{y}=0$, por ende $L_k=\langle \bar{z}-w_k, Ap_k-Bq_k\rangle$.

Como $r_k-\bar{z}=w_k-\bar{z}+\tau_k(Ap_k-Bq_k)$ tenemos que

$$\|r_k-\bar{z}\|^2=\|w_k-\bar{z}\|^2+\tau_k^2\|Ap_k-Bq_k\|^2+2\tau_k\langle \bar{z}-w_k, Ap_k-Bq_k\rangle,$$

Por ende

$$L_k=\frac{1}{2\tau_k}\left[\|w_k-\bar{z}\|^2-\|r_k-\bar{z}\|^2\right]+\frac{\tau_k}{2}\|Ap_k-Bq_k\|^2$$

Es decir, obtenemos la desigualdad deseada

$$\begin{aligned}\frac{1}{2\tau_k}[\|p_k-u_k\|^2+\|q_k-v_k\|^2+\|p_k-\bar{x}\|^2+\|q_k-\bar{y}\|^2] \leq \frac{1}{2\tau_k}[\|v_k-\bar{y}\|^2+\|u_k-\bar{x}\|^2 \\ +\|w_k-\bar{z}\|^2-\|r_k-\bar{z}\|^2] \\ +\frac{\tau_k}{2}\|Ap_k-Bq_k\|^2+R_k.\end{aligned}$$

□

La desigualdad anterior nos da una idea de como la elección de r_k^{i} afecta la convergencia de $Ap_k - Bq_k$ a través del término R_k. Usaremos lo anterior para encontrar una condición suficiente para que $Ap_k - Bq_k \to 0$.

Lema 2.3 *Si las sucesiones (p_k, q_k, r_k) y (u_k, v_k, w_k) y los errores $\theta_k^{\mathrm{i}} = r_k^{\mathrm{i}} - r_k$ con $\mathrm{i} = 1, 2$ son $o(\lambda_k)$. Luego, $Ap_k - Bq_k$ converge fuertemente a 0 cuando $k \to \infty$ en el algoritmo (RIPA)$_{(r^1,r^2)}$.*

Demostración. Notemos que R_k puede ser reescrito de la siguiente manera

$$\begin{aligned}R_k &= \langle r_k^1 - w_k, A(\bar{x}-p_k)\rangle + \langle w_k - r_k^2, B(\bar{y}-q_k)\rangle \\ &= \langle r_k + \theta_k^1 - w_k, A(\bar{x}-p_k)\rangle + \langle w_k - r_k - \theta_k^2, B(\bar{y}-q_k)\rangle \\ &= \langle r_k - w_k, A(\bar{x}-p_k)\rangle + \langle w_k - r_k, B(\bar{y}-q_k)\rangle + \langle \theta_k^1, A(\bar{x}-p_k)\rangle - \langle \theta_k^2, B(\bar{y}-q_k)\rangle\end{aligned}$$

Primero notemos que los primeros dos términos son los mismos que aparecen en la demostración del Lema 2.4, por ende

$$R_k = -\tau_k\|Ap_k - Bq_k\|^2 + \langle \theta_k^1, A(\bar{x}-p_k)\rangle - \langle \theta_k^2, B(\bar{y}-q_k)\rangle$$

Es decir, remplazando esto en (2.8) obtenemos que

$$\begin{aligned}\frac{1}{2\tau_k}[\|p_k-u_k\|^2+\|q_k-v_k\|^2+\|p_k-\bar{x}\|^2+\|q_k-\bar{y}\|^2] \leq \frac{1}{2\tau_k}[\|v_k-\bar{y}\|^2+\|u_k-\bar{x}\|^2] \\ +\frac{1}{2\tau_k}[\|w_k-\bar{z}\|^2-\|r_k-\bar{z}\|^2] \\ -\frac{\tau_k}{2}\|Ap_k-Bq_k\|^2 \\ +\langle\theta_k^1, A(\bar{x}-p_k)\rangle - \langle\theta_k^2, B(\bar{y}-q_k)\rangle,\end{aligned}$$

Entonces, dividiendo esto por τ_k conseguimos la siguiente desigualdad

$$\frac{1}{2\tau_k^2}F_k + \frac{1}{2}\|Ap_k - Bq_k\|^2 \leq \frac{1}{2\tau_k^2}G_k + \frac{1}{\tau_k}\langle\theta_k^1, A(\bar{x}-p_k)\rangle - \frac{1}{\tau_k}\langle\theta_k^2, B(\bar{y}-q_k)\rangle, \tag{2.12}$$

Donde $F_k = \|p_k-u_k\|^2+\|q_k-v_k\|^2+\|p_k-\bar{x}\|^2+\|q_k-\bar{y}\|^2$ y $G_k = \|v_k-\bar{y}\|^2+\|u_k-\bar{x}\|^2+\|w_k-\bar{z}\|^2-\|r_k-\bar{z}\|^2$.
Como $(\langle\theta_k^1, A(\bar{x}-p_k)\rangle)^2 \leq \|\theta_k^1\|^2\|A(\bar{x}-p_k)\|^2$ y $(\langle\theta_k^2, B(\bar{y}-q_k)\rangle)^2 \leq \|\theta_k^2\|^2\|B(\bar{y}-q_k)\|^2$ por

la desigualdad de Cauchy-Schwarz.
Conseguimos que

$$\left|\frac{1}{\tau_k}\langle\theta_k^1, A(\bar{x}-p_k)\rangle\right| \leq \left\|\frac{\theta_k^1}{\tau_k}\right\| \|A(\bar{x}-p_k)\|$$
$$\left|\frac{1}{\tau_k}\langle\theta_k^2, B(\bar{y}-q_k)\rangle\right| \leq \left\|\frac{\theta_k^2}{\tau_k}\right\| \|B(\bar{y}-q_k)\|$$

Por ende como las sucesiones θ_k^{i} son $o(\lambda_k)$ obtenemos que $\frac{1}{\tau_k}\langle\theta_k^1, A(\bar{x}-p_k)\rangle$ and $\frac{1}{\tau_k}\langle\theta_k^2, B(\bar{y}-q_k)\rangle$ convergen fuertemente a 0.
Finalmente, usando esto en la desigualdad (2.12) obtenemos que $Ap_k - Bq_k \to 0$. □

También para ejemplificar lo que mencionamos al final de la subsección anterior, donde RIPA satisfacía todas las condiciones del Lema, con el resultado anterior podemos calcular la velocidad de convergencia a la factibilidad del par (p_k, q_k) en el caso de RIPA.

Lema 2.4 *Para $\alpha > 2$ y $\lambda_k = (1+\varepsilon)(sk^2/\alpha)$, luego, la sucesión (p_k, q_k, r_k) generada por (RIPA) satisface que*

$$Ap_k - Bq_k = o\,(1/k)\,.$$

Demostración. Como es probado en el Teorema 3.6 de [3] las sucesiones (p_k, q_k, r_k) y (u_k, v_k, w_k) son acotadas, luego como (RIPA) es equivalente al algoritmo generalizado $(\mathrm{RIPA})_{(r_k, r_k)}$ obtenemos que

$$\begin{aligned} R_k &= \langle r_k - w_k, A(\bar{x}-p_k)\rangle + \langle w_k - r_k, B(\bar{y}-q_k)\rangle \\ &= \langle w_k - r_k, Ap_k - A\bar{x} + B\bar{y} - Bq_k\rangle \\ &= \langle w_k - r_k, Ap_k - Bq_k\rangle \\ &= \langle -\tau_k(Ap_k - Bq_k), Ap_k - Bq_k\rangle \\ &= -\tau_k\|Ap_k - Bq_k\|^2 \end{aligned}$$

Luego, al remplazar esto en (2.8) conseguimos lo siguiente

$$\begin{aligned} \frac{1}{2\tau_k}[\|p_k-u_k\|^2 + \|q_k-v_k\|^2 + \|p_k-\bar{x}\|^2 + \|q_k-\bar{y}\|^2] \leq{}& \frac{1}{2\tau_k}[\|v_k-\bar{y}\|^2 + \|u_k-\bar{x}\|^2 +] \\ &+ \frac{1}{2\tau_k}[\|w_k-\bar{z}\|^2 - \|r_k-\bar{z}\|^2] \\ &- \frac{\tau_k}{2}\|Ap_k - Bq_k\|^2 \end{aligned}$$

Finalmente como (p_k, q_k, r_k) y (u_k, v_k, w_k) son acotadas obtenemos que

$$\frac{\tau_k}{2}\|Ap_k - Bq_k\|^2 \to 0 \text{ cuando } k \to +\infty$$

Es decir, la sucesión $\sqrt{\tau_k}(Ap_k - Bq_k)$ converge fuertemente a 0 cuando $k \to +\infty$, luego como $\lambda_k = (1+\varepsilon)(sk^2/\alpha)$ obtenemos que $k(Ap_k - Bq_k)$ converge fuertemente a 0 cuando $k \to +\infty$, concluyendo la demostración. □

Algo interesante del Lema (2.3) es que una de las condiciones de convergencia para $(RIPA)_{(r_k^1, r_k^2)}$ es pedir que las sucesiones θ_k^{i} y $Ap_k - Bq_k$ converjan fuertemente a 0, pero con el Lema anterior obtuvimos que $Ap_k - Bq_k \to 0$ si $\theta_k^{\mathrm{i}} = o(\lambda_k)$. Esta es una condición más débil que $\theta_k^{\mathrm{i}} \to 0$, por ende la condición $Ap_k - Bq_k \to 0$ esta contenida en la condición $\theta_k^{\mathrm{i}} \to 0$.
Esto quiere decir que podemos obtener un nuevo Lema de convergencia directamente del Lema 2.1.

Lema 2.5 *Si las siguientes condiciones se cumplen*

(i) Las sucesiones (p_k, q_k, r_k) y (u_k, v_k, w_k) son $o(\lambda_k)$.

(ii) Las sucesiones $r_k^1 - r_k$, $r_k^2 - r_k$ convergen fuertemente a 0 cuando $k \to +\infty$.

Luego la sucesión (p_k, q_k, r_k) generada por $(RIPA)_{(r_k^1, r_k^2)}$ satisface que todo punto de acumulación $(p_\infty, q_\infty, r_\infty)$ esta en S.

2.2.3. Formulación variacional del algoritmo generalizado.

Una de las adaptaciones que hace el algoritmo (ip-ADMM) para aproximar RIPA es pasar de un paso de minimización (en la formulación variacional) a hacerlo de forma secuencial con dos pasos de minimización. Esto puede ser evidenciado en (1.20), a pesar de ser intuitivo a simple vista, esto hace muy difícil compararlo con un algoritmo que corresponde a un solo paso de minimización como lo es RIPA.
Algo positivo que tiene el algoritmo RIPA es que podemos definir su formulación variacional en un solo paso de minimización, esta formulación variacional es mencionada en (2.3), nos gustaría obtener algo similar para un subconjunto de algoritmos de la familia RIPA generalizada.

Lema 2.6 *Dado r_k^{i} con $\mathrm{i} = 1, 2$. Si consideramos la función*

$$\begin{aligned} \Psi_k \colon X \times Y \times X \times Y &\to \mathbb{R} \\ (p, q, u, v) &\mapsto \Psi_k(p, q, u, v). \end{aligned}$$

Tal que satisface la siguiente ecuación diferencial

$$\partial_p \Psi_k = A^t \theta_k^1, \ \partial_q \Psi_k = -B^t \theta_k^2. \tag{2.13}$$

Donde $\theta_k^{\mathrm{i}} = r_k^{\mathrm{i}} - r_k$ para $\mathrm{i} = 1, 2$.
Luego, los pasos proximales del algoritmo $(RIPA)_{(r_k^1, r_k^2)}$ (p_k, q_k) son mínimos de la siguiente función

$$\frac{1}{2\tau_k}\|p - u_k\|^2 + \frac{1}{2\tau_k}\|q - v_k\|^2 + f(p) + g(q) + \langle w, Ap - Bq\rangle + \frac{\tau_k}{2}\|Ap - Bq\|^2 + \Psi_k(p, q, u_k, v_k).$$

Demostración. Notemos que el algoritmo generalizado puede ser escrito de la siguiente ma-

nera

$$(RIPA)_{(r^1,r^2)}\begin{cases} u_k = x_k + \alpha_k(x_k - x_{k-1}) \\ v_k = y_k + \alpha_k(y_k - y_{k-1}) \\ w_k = z_k + \alpha_k(z_k - z_{k-1}) \\ \dfrac{1}{\lambda_k + s}(p_k - u_k) + \partial f(p_k) + A^t r_k + A^t\theta_k^1 \ni 0 \\ \dfrac{1}{\lambda_k + s}(q_k - v_k) + \partial g(q_k) - B^t r_k - B^t\theta_k^2 \ni 0 \\ r_k = w_k + (\lambda_k + s)(Ap_k - Bq_k) \\ x_{k+1} = \dfrac{\lambda_k}{\lambda_k + s}u_k + \dfrac{s}{\lambda_k + s}p_k \\ y_{k+1} = \dfrac{\lambda_k}{\lambda_k + s}v_k + \dfrac{s}{\lambda_k + s}q_k \\ z_{k+1} = \dfrac{\lambda_k}{\lambda_k + s}w_k + \dfrac{s}{\lambda_k + s}r_k. \end{cases} \tag{2.14}$$

Luego, notando la ecuación diferencial que satisface Ψ los términos proximales satisfacen el siguiente sistema

$$\begin{aligned} &\frac{1}{\lambda_k + s}(p_k - u_k) + \partial f(p_k) + A^t(w_k + \tau_k(Ap_k - Bq_k)) + \partial_p\Psi(p_k, q_k, u_k, v_k) \ni 0 \\ &\frac{1}{\lambda_k + s}(q_k - v_k) + \partial g(q_k) - B^t(w_k + \tau_k(Ap_k - Bq_k)) + \partial_q\Psi(p_k, q_k, u_k, v_k \ni 0, \end{aligned}$$

Por ende, dado por las condiciones de primer orden, obtenemos que (p_k, q_k) son mínimos de la siguiente función

$$\frac{1}{2\tau_k}\|p - u_k\|^2 + \frac{1}{2\tau_k}\|q - v_k\|^2 + f(p) + g(q) + \langle w, Ap - Bq\rangle + \frac{\tau_k}{2}\|Ap - Bq\|^2 + \Psi_k(p, q, u_k, v_k).$$

Probando lo pedido. □

Es importante mencionar que no todos los posibles algoritmos generalizados pueden ser escritos de esta manera, pues la ecuación diferencial (2.13) no tiene solución para cualquier θ_k^{i}, por ejemplo (ip-ADMM) no tiene función Ψ que la satisfaga.

Sin embargo hay ejemplos en los cuales si funciona. Uno de ellos es aproximando los términos proximales de (RIPA) de la siguiente manera

$$\begin{aligned} &\frac{1}{\lambda_k + s}(p_k - u_k) + \partial f(p_k) + A^t(w_k + \tau_k(Ap_k - Bv_k)) \ni 0 \\ &\frac{1}{\lambda_k + s}(q_k - v_k) + \partial g(q_k) - B^t(w_k + \tau_k(Au_k - Bq_k)) \ni 0 \end{aligned} \tag{2.15}$$

Luego $A^t\theta_k^1 = \tau_k A^t B(q_k - v_k)$ y $-B^t\theta_k^2 = \tau_k B^t A(p_k - u_k)$, entonces si consideramos

$$\Psi_k(p, q, u, v) = \tau_k\langle A(p - u), B(q - v)\rangle,$$

esta función sí satisface la ecuación (2.13), llamaremos a esta función **representante** del algoritmo generalizado.

2.3. Resultados de convergencia de ip-ADMM.

Gracias a que ip-ADMM es un algoritmo perteneciente a la familia de RIPA generalizado, podemos obtener algunos resultados de convergencia a través de condiciones suficientes para este algoritmo, las cuales son directas del Lema 2.1.

Lema 2.7 *Si las siguientes condiciones se cumplen*

(i) Las sucesiones (p_k, q_k, r_k) *y* (u_k, v_k, w_k) *son* $o(\lambda_k)$.

(ii) La sucesión $\lambda_k(q_k - v_k)$ *converge fuertemente a 0 cuando* $k \to +\infty$.

Luego, la sucesión (p_k, q_k, r_k) *generada por (ip-ADMM) satisface que todo punto de acumulación* $(p_\infty, q_\infty, r_\infty)$ *está en* S.

Algo interesante del lema anterior es que no solo se pide que la sucesión $q_k - v_k$ converja fuertemente a 0, si no que además lo haga a velocidad $o(1/\lambda_k)$, esto quiere decir que debemos ser cuidadosos al momento de elegir el λ_k a usar, pues si crece muy rápido la condición (ii) se puede no satisfacer, pero si crece muy lento el algoritmo no converge suficientemente rápido.

Dado esto, resulta muy interesante verificar bajo qué hipótesis la condición (ii) es una condición necesaria para la convergencia de ip-ADMM.

Lema 2.8 *Si las funciones* f, g *son diferenciables, el operador de punto silla* M_P *es continuo, la sucesión* (p_k, q_k, r_k) *tiene un punto de acumulación* $(x_\infty, y_\infty, z_\infty)$ *y* (u_k, v_k, w_k), (p_k, q_k, r_k) *son* $o(\lambda_k)$, *luego*

$$\lambda_k(q_k - v_k) \to 0 \text{ cuando } \lambda_k \to \infty \Leftrightarrow (x_\infty, y_\infty, z_\infty) \in S$$

Demostración. ($\Rightarrow$) Como se satisfacen las condiciones del Lema 2.7 esta implicancia es directa.
($\Leftarrow$) Primero notemos como las funciones f, g son diferenciables, el operador M_P pasa de ser una multi-función a una función, por ende la inclusión descrita en 2.7 pasa a ser una igualdad, es decir, las sucesiones satisfacen la siguiente igualdad

$$M_P(p_k, q_k, r_k) = \left(\frac{-1}{\lambda_k + s}(p_k - u_k) - (\lambda_k + s)A^t B(q_k - v_k), \frac{-1}{\lambda_k + s}(q_k - v_k), Bq_k - Ap_k \right), \tag{2.16}$$

teniendo esto probaremos lo pedido.
Primero supongamos que $\lambda_k(q_k - v_k) \to y \in Y$ tal que $y \neq 0$, luego usando la ecuación (2.16) sobre las subsucesiones que convergen al punto de acumulación y usando además que M_P es continua obtenemos que

$$M_p(x_\infty, y_\infty, z_\infty) = (y, 0, 0) \neq 0,$$

con lo que llegamos a que $(x_\infty, y_\infty, z_\infty) \notin S$, lo cual es una contradicción.
Finalmente en el caso que $\lambda_k(q_k - v_k)$ sea una secuencia no acotada (en el caso que sea acotada podemos repetir el argumento anterior usando subsucesiones), de (2.16) obtendremos que es

una sucesión no acotada, pero como $(p_k, q_k, r_k) \rightharpoonup (x_\infty, y_\infty, z_\infty)$, dado que el operador M_P es continuo obtenemos una contradicción. □

Teniendo el Lema anterior podemos formular el siguiente Teorema.

Teorema 2.9 *Si la sucesión (p_k, q_k, r_k) tiene los mismos puntos de acumulación que (x_k, y_k, z_k), f,g son diferenciables, M_P es continuo y (u_k, v_k, w_k), (p_k, q_k, r_k) son $o(\lambda_k)$, luego la hipótesis $\lambda_k(q_k - v_k) \to 0$ es una condición **necesaria** para la convergencia del algoritmo ip-ADMM.*

Demostración. Si el algoritmo (ip-ADMM) converge , luego $(x_k, y_k, z_k) \to (x_\infty, y_\infty, z_\infty) \in S$, como (p_k, q_k, r_k) tiene los mismos puntos de acumulación que (x_k, y_k, z_k) podemos asumir que una subsucesión de (p_k, q_k, r_k) converge a $(x_\infty, y_\infty, z_\infty)$, por simplicidad asumiremos que la sucesión hace esto.
Luego dado que se satisfacen las hipótesis del Lema 2.8 si la condición $\lambda_k(q_k - v_k) \to 0$ no se satisface obtendremos que $(x_\infty, y_\infty, z_\infty) \notin S$ llegando a una contradicción. □

Capítulo 3

Implementación numérica.

En este capítulo presentaremos implementaciones del algoritmo (ip-ADMM) y RIPA generalizado, además discutiremos los resultados obtenidos y presentaremos los principales problemas que tienen estos algoritmos a la hora de ser implementados.

3.1. Problemas de implementación

Cuando inicialmente definimos (ip-ADMM) mencionamos que este nacía como una aproximación del algoritmo RIPA para calcular los términos proximales p_k y q_k de forma secuencial a diferencia de resolver un sistema acoplado como lo hace RIPA, esto funciona bien en teoría suponiendo que podemos calcular de forma analítica la ecuación para p_k y q_k, esto lamentablemente no ocurre siempre.
Recordemos que para obtener los valores de p_k y q_k en una iteración del algoritmo debemos resolver las siguientes ecuaciones.

$$\begin{cases} \frac{1}{\lambda_k+s}(p_k-u_k)+\partial f(p_k)+A^t(w_k+(\lambda_k+s)(Ap_k-Bv_k))\ni 0, \\ \frac{1}{\lambda_k+s}(q_k-v_k)+\partial g(q_k)-B^t(w_k+(\lambda_k+s)(Ap_k-Bq_k))\ni 0. \end{cases}$$

Es decir p_k y q_k satisfacen lo siguiente

$$u_k-(\lambda_k+s)A^tw_k+(\lambda_k+s)^2A^tBv_k\in\left(I+(\lambda_k+s)^2A^tA+(\lambda_k+s)\partial f\right)p_k, \tag{3.1}$$
$$v_k+(\lambda_k+s)B^tw_k+(\lambda_k+s)^2B^tAp_k\in\left(I+(\lambda_k+s)^2B^tB+(\lambda_k+s)\partial g\right)q_k. \tag{3.2}$$

Por ende, para obtener estos valores es necesario obtener los operadores $(I+(\lambda_k+s)^2A^tA+(\lambda_k+s)\partial f)^{-1}$ y $(I+(\lambda_k+s)^2B^tB+(\lambda_k+s)\partial g)^{-1}$, ambos pueden ser caracterizados como el operador proximal de las funciones

$$F_k(x)=f(x)+\frac{(\lambda_k+s)}{2}\|Ax\|^2,\ G_k(y)=g(y)+\frac{(\lambda_k+s)}{2}\|By\|^2. \tag{3.3}$$

El principal problema que surge de esto es que no podemos obtener estos operadores proximales de forma analítica aunque tengamos los operadores proximales de f y g (no existe

una regla de cálculo del operador proximal de la suma de dos funciones),por lo que esto puede resultar en que implementar el algoritmo sea muy costoso, pues deberemos calcular o aproximar el operador proximal de forma numérica en vez de usar directamente su forma analítica.

Esta es otra razón por la cual estudiar la familia de algoritmos RIPA generalizado puede ser muy útil, pues podemos encontrar otras formas de aproximar RIPA de forma eficiente e implementable.

3.1.1. Algoritmos de RIPA generalizado que solucionan los problemas de implementación.

Para ejemplificar como RIPA generalizado puede resolver este problema presentaremos algunos ejemplos de algoritmos pertenecientes a la familia de RIPA generalizado que hacen esto.

Una posible adaptación para evitar los problemas presentados anteriormente es calcular los términos proximales de la siguiente manera

$$\begin{cases} \dfrac{1}{\lambda_k + s}(p_k - u_k) + \partial f(p_k) + A^t r_k^1 \ni 0 \\ \dfrac{1}{\lambda_k + s}(q_k - v_k) + \partial g(q_k) - B^t r_k^2 \ni 0, \end{cases} \tag{3.4}$$

donde los términos r^1 y r^2 que caracterizan al algoritmo son

$$r_k^1 = w_k + (\lambda_k + s)(Au_k - Bv_k), \quad r_k^2 = w_k + (\lambda_k + s)(Ap_k - Bv_k).$$

Por lo que los errores θ^1 y θ^2 son

$$\theta_k^1 = (\lambda_k + s)(A(u_k - p_k) - B(v_k - q_k)), \quad \theta_k^2 = (\lambda_k + s)B(q_k - v_k).$$

Luego, recordando que una de las condiciones cruciales para asegurar la convergencia a través de subsucesiones de los términos proximales mencionadas en el Lema 2.5 es que θ^1 y θ^2 converjan a 0, es decir, para que ocurra esto necesitamos que las sucesiones $\lambda_k(q_k - v_k)$ y $\lambda_k(u_k - p_k)$ se vayan a 0, recordemos que el algoritmo solo pedía que la primera sucesión convergiera a 0, esto nos muestra que entre más aproximemos el algoritmo RIPA, más condiciones serán necesarias satisfacer para asegurar la convergencia.

Otro ejemplo de como usar RIPA generalizado para resolver este problema de implementación corresponde a tomar el camino que usa el algoritmo del gradiente proximal.
Para obtener los operadores proximales $(I + \lambda_k F_k)^{-1}$ y $(I + \lambda_k G_k)^{-1}$ donde F_k y G_k son las funciones presentadas en (3.3) podemos usar una idea parecida a la usada por el algoritmo de gradiente proximal para calcular un término similar, presentaremos el algoritmo del gradiente proximal, descrito en mayor profundidad en [13], a continuación. Este algoritmo es usado para resolver el siguiente problema

$$\text{mín } p(x) + h(x) \tag{3.5}$$

donde p es una función convexa y diferenciable y h es una función cualquiera a la cual le podemos calcular el operador proximal fácilmente, si aplicamos el algoritmo proximal presentado en la Sección 1.3.3 al problema (3.5) obtenemos el siguiente algoritmo

$$x_{k+1} = (I + \lambda_k \nabla p + \lambda_k \partial h)^{-1} x_k.$$

Esto deriva a un problema similar al mencionado en la Sección 3.1, por lo que el método del gradiente proximal propone aproximar el término proximal de la siguiente manera

$$x_{k+1} = \text{prox}_{\lambda_k h}(x_k - \lambda_k \nabla p(x_k))$$

Finalmente como $u \in \text{prox}_h(x) \Leftrightarrow x - u \in \partial h(u)$ obtenemos que

$$\text{prox}_{\lambda_k h}(x - \lambda_k \nabla p(x)) = (I + \lambda_k \partial h)^{-1}(I - \lambda_k \nabla p)(x). \tag{3.6}$$

Luego volviendo al problema original que resuelve (ip-ADMM) y consideramos $f = h$, $g = p$ podemos aproximar los operadores proximales necesarios para implementar el algoritmo de la siguiente forma

$$(I + \lambda_k F_k)^{-1} = \left(I + \lambda_k \left(f(x) + \frac{(\lambda_k + s)}{2}\|Ax\|^2\right)\right)^{-1} \approx (I + \lambda_k \partial f)^{-1} \left(I - \lambda_k \nabla \left(\frac{\lambda_k}{2}\|Ax\|^2\right)\right)$$
$$(I + \lambda_k G_k)^{-1} = \left(I + \lambda_k \left(g(y) + \frac{(\lambda_k + s)}{2}\|By\|^2\right)\right)^{-1} \approx (I + \lambda_k \partial g)^{-1} \left(I - \lambda_k \nabla \left(\frac{\lambda_k}{2}\|By\|^2\right)\right)$$

Por ende obteniendo el siguiente algoritmo de RIPA generalizado

$$\begin{cases} p_k = (I + (\lambda_k + s)\partial f)^{-1}(I - (\lambda_k + s)^2 A^t A)(u_k - (\lambda_k + s)A^t w_k + (\lambda_k + s)^2 A^t B v_k) \\ q_k = (I + (\lambda_k + s)\partial g)^{-1}(I - (\lambda_k + s)^2 B^t B)(v_k + (\lambda_k + s)B^t w_k + (\lambda_k + s)^2 B^t A p_k) \end{cases}$$

Ambos algoritmos pertenecientes a la familia de RIPA generalizado resuelven el problema de implementación presentado por el algoritmo (ip-ADMM), pero pagan el costo de necesitar condiciones más fuertes para funcionar correctamente.

3.2. Familia de problemas en los cuales puede ser implementado ip-ADMM.

Parte importante del análisis numérico es poder implementar ip-ADMM en un problema específico y comparar con alguna otra variante de RIPA generalizado y otros algoritmos similares, por lo que presentaremos un tipo de problema en el cual ip-ADMM puede ser implementado.

Consideremos el siguiente problema

$$\text{mín}\, f(x) + g(Ax), \tag{3.7}$$

donde A es una matriz unitaria, es decir satisface que $A^t A = I$.
Este problema puede ser transformando a la forma estándar que resuelve el algoritmo ip-ADMM agregando una variable $y = Ax$ y $B = I$ obteniendo el siguiente problema.

$$\begin{cases} \underset{(x,y)\in X\times Y}{\text{mín}} & f(x) + g(y) \\ \text{s.t.} & Ax - y = 0, \end{cases} \tag{3.8}$$

Consideramos que la matriz A sea unitaria para poder sobrepasar el problema de implementación presentado al final del Capítulo 3, pues los pasos proximales que corresponden a

$$\begin{cases} \frac{1}{\lambda_k+s}(p_k-u_k)+\partial f(p_k)+A^t(w_k+(\lambda_k+s)(Ap_k-Bv_k))\ni 0 \\ \frac{1}{\lambda_k+s}(q_k-v_k)+\partial g(q_k)-B^t(w_k+(\lambda_k+s)(Ap_k-Bq_k))\ni 0, \end{cases} \tag{3.9}$$

pueden ser simplifcados si consideramos que $A^tA=I$, en la primera ecuación obtenemos que p_k satisface la siguiente ecuación

$$p_k=\text{prox}_{\frac{\tau_k}{\tau_k^2+1}f}\left(\frac{u_k}{\tau_k^2+1}-\frac{\tau_kA^tw_k}{\tau_k^2+1}+\frac{\tau_k^2A^tBv_k}{\tau_k^2+1}\right) \tag{3.10}$$

Haciendo algo similar para q_k pues $B=I$ obtenemos que p_k y q_k satisfacen

$$\begin{cases} p_k=\text{prox}_{\frac{\tau_k}{\tau_k^2+1}f}\left(\frac{u_k}{\tau_k^2+1}-\frac{\tau_kA^tw_k}{\tau_k^2+1}+\frac{\tau_k^2A^tv_k}{\tau_k^2+1}\right) \\ q_k=\text{prox}_{\frac{\tau_k}{\tau_k^2+1}g}\left(\frac{v_k}{\tau_k^2+1}+\frac{\tau_kw_k}{\tau_k^2+1}+\frac{\tau_k^2Ap_k}{\tau_k^2+1}\right) \end{cases} \tag{3.11}$$

Por ende el algoritmo para resolver un problema del tipo

$$\min f(x)+g(Ax),$$

donde $A:X\to Y$ es un operador unitario, con el algoritmo ip-ADMM es el siguiente

$$\begin{cases} u_k=x_k+\alpha_k(x_k-x_{k-1}) \\ v_k=y_k+\alpha_k(y_k-y_{k-1}) \\ w_k=z_k+\alpha_k(z_k-z_{k-1}) \\ p_k=\text{prox}_{\frac{\tau_k}{\tau_k^2+1}f}\left(\frac{u_k}{\tau_k^2+1}-\frac{\tau_kA^tw_k}{\tau_k^2+1}+\frac{\tau_k^2A^tv_k}{\tau_k^2+1}\right) \\ q_k=\text{prox}_{\frac{\tau_k}{\tau_k^2+1}g}\left(\frac{v_k}{\tau_k^2+1}+\frac{\tau_kw_k}{\tau_k^2+1}+\frac{\tau_k^2Ap_k}{\tau_k^2+1}\right) \\ r_k=w_k+(\lambda_k+s)(Ap_k-Bq_k) \\ x_{k+1}=\frac{\lambda_k}{\lambda_k+s}u_k+\frac{s}{\lambda_k+s}p_k \\ y_{k+1}=\frac{\lambda_k}{\lambda_k+s}v_k+\frac{s}{\lambda_k+s}q_k \\ z_{k+1}=\frac{\lambda_k}{\lambda_k+s}w_k+\frac{s}{\lambda_k+s}r_k, \end{cases} \tag{3.12}$$

donde $\alpha_k=(1-\frac{\alpha}{k})$ y $\tau_k=\lambda_k+s$.
Por ende somos capaces de aplicar el algoritmo (ip-ADMM) a este tipo de problemas si tenemos los operadores proximales de f y g de forma analítica.

3.3. Aplicación a un problema de *sparsity*

Un problema usual en tratamiento de imágenes corresponde al de dado una imagen de referencia $\mathrm{d}\in X$ y un operador $A:X\to Y$ resolver el problema (3.7) donde $f(\cdot)=\frac{1}{2}\|\cdot-\mathrm{d}\|_2^2$

y $g(\cdot) = \theta\|\cdot\|_1$ donde $\theta \in \mathbb{R}$ es una penalización, este problema consiste en buscar la imagen que más se parece a d en norma 2, pero que además no tenga magnitud muy grande en norma 1 al aplicarle el operador A.

Un operador usado comúnmente en este problema corresponde a la Transformada de Fourier discreta, la cual es unitaria, en este caso usaremos una versión equivalente de esta llamada matriz de Hadamard la cual tiene la siguiente construcción recursiva

$$
\begin{aligned}
H_0 &= +\begin{pmatrix} 1 \end{pmatrix} \\
H_1 &= \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \\
H_2 &= \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \\
H_3 &= \frac{1}{2^{\frac{3}{2}}}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix} \\
(H_n)_{\mathrm{i},j} &= \frac{1}{2^{\frac{n}{2}}}(-1)^{\mathrm{i}\cdot j}
\end{aligned}
$$

Donde $\mathrm{i}\cdot j$ es el producto interno de la representación binaria de los números i, j bit a bit. La transformada de Hadamard es equivalente a la transformada de Fourier discreta de tamaño $2 \times 2 \times \ldots \times 2$.

3.3.1. Implementación y comparación de algoritmos pertenecientes a RIPA generalizado.

Durante el desarrollo de este trabajo presentamos la familia de algoritmos como una solución a no poder implementar los algoritmos RIPA o ip-ADMM sin perder la efectividad de estos, por lo que en esta sección presentaremos los resultados de implementar 4 algoritmos pertenecientes a la familia de RIPA generalizado incluyendo ip-ADMM, además discutiremos los pros y contras que tienen el uso de cada algoritmo específicamente.
Primero recordemos que para calcular los términos proximales de un algoritmo perteneciente a RIPA generalizado debemos resolver el siguiente sistema de ecuaciones.

$$
\begin{cases}
\dfrac{1}{\lambda_k + s}(p_k - u_k) + \partial f(p_k) + A^t r_k^1 \ni 0, \\
\dfrac{1}{\lambda_k + s}(q_k - v_k) + \partial g(q_k) - B^t r_k^2 \ni 0.
\end{cases}
\tag{3.13}
$$

Los algoritmos que usaremos a continuación son los siguientes

Algoritmo	r_k^1	r_k^2	θ_k^1	θ_k^2
(ip-ADMM)	$w_k + \tau_k(Ap_k - Bv_k)$	$w_k + \tau_k(Ap_k - Bq_k)$	$\tau_k B(q_k - v_k)$	0
$(\text{RIPAG})_1$	$w_k + \tau_k(Ap_k - Bv_k)$	$w_k + \tau_k(Au_k - Bq_k)$	$\tau_k B(q_k - v_k)$	$\tau_k A(u_k - p_k)$
$(\text{RIPAG})_2$	$w_k + \tau_k(Au_k - Bv_k)$	$w_k + \tau_k(Ap_k - Bq_k)$	$\tau_k B(q_k - v_k)$ $+\tau_k A(u_k - p_k))$	0
$(\text{RIPAG})_3$	$w_k + \tau_k(Au_k - Bv_k)$	$w_k + \tau_k(Au_k - Bv_k)$	$\tau_k B(q_k - v_k)$ $+\tau_k A(u_k - p_k))$	$\tau_k B(q_k - v_k)$ $+\tau_k A(u_k - p_k))$

Donde $\tau_k = \lambda_k + s$.

Notemos que para cada caso enumeramos los errores generados por estos algoritmos con respecto al algoritmo original RIPA, recordemos el Lema 2.5 nos dice que una de las condiciones suficientes para que el algoritmo converja a través de subsucesiones a una solución es que los errores θ^1 y θ^2 converjan a 0, por lo que entender que significa cada error es esencial para analizar estos algoritmos.
Por ejemplo, la temática común de todos algoritmos es que las sucesiones $A(u_k - p_k)$ y $B(q_k - v_k)$ sean $o(1/\tau_k)$, algunos algoritmos solo piden que se cumpla para la segunda sucesión como ip-ADMM y $(\text{RIPAG})_2$, pero el resto pide que ambas sucesiones lo cumplan.

Sobre los problemas de implementación para un caso más general donde A es una matriz cualquiera y la segunda línea de (3.13) puede ser calculada eficientemente teniendo p_k (como lo es el caso de $B = I$) los algoritmos ip-ADMM y $(\text{RIPAG})_1$ no pueden ser implementados, en cambio, el algoritmo $(\text{RIPAG})_2$ si puede ser implementado pues solo debemos calcular el operador proximal de f para obtener el valor de p_k en (3.13), este es un beneficio del algoritmo $(\text{RIPAG})_2$ pues no necesita una matriz unitaria para poder ser implementado, lo mismo se puede mencionar del algoritmo $(\text{RIPAG})_3$ (donde además ambos tienen la misma condición de convergencia dada por sus errores θ^{i}), pero además este ultimo algoritmo tiene el beneficio agregado de que puede ser paralelizado, es decir, podemos calcular los términos p_k y q_k de manera paralela y no secuencial, esto trae muchísimos beneficios en la eficiencia del algoritmo.

La implementación de estos algoritmos para el caso $N = 2^5$ dieron los siguientes resultados.

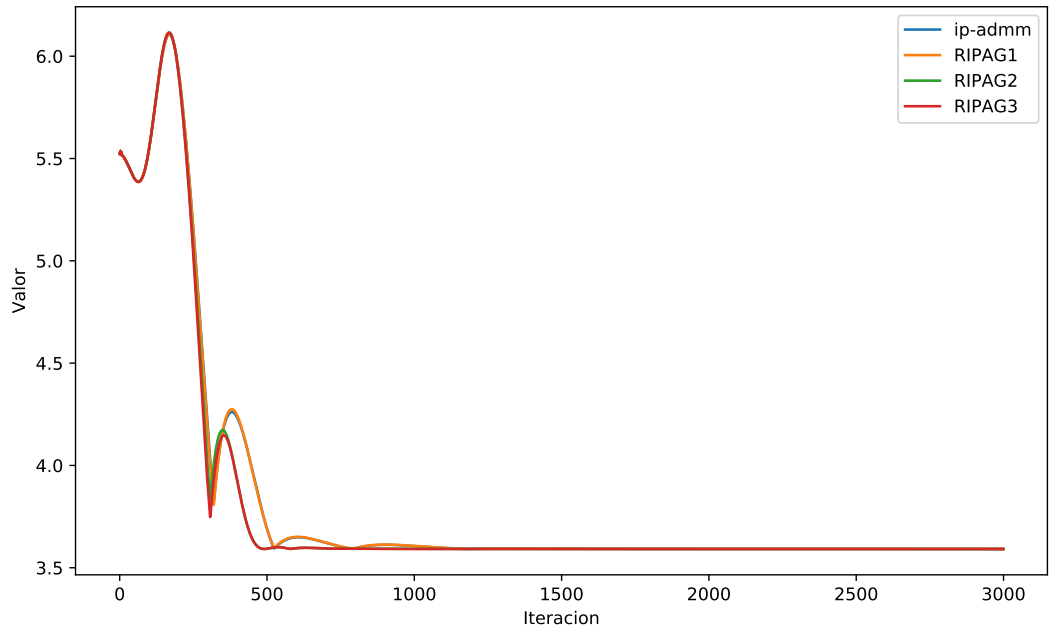

Figura 3.1: Comparación de los 4 algoritmos

Podemos notar inmediatamente que los 4 algoritmos se comportan de manera similar en cada iteración llegando al mismo resultado, además si calculamos la norma 2 del valor de los errores θ_1 y θ_2 de cada iteración obtenemos lo siguiente

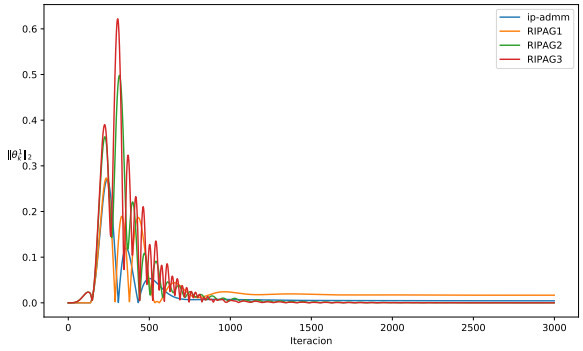

Figura 3.2: Valor de $\|\theta_k^1\|_2$ por iteración.

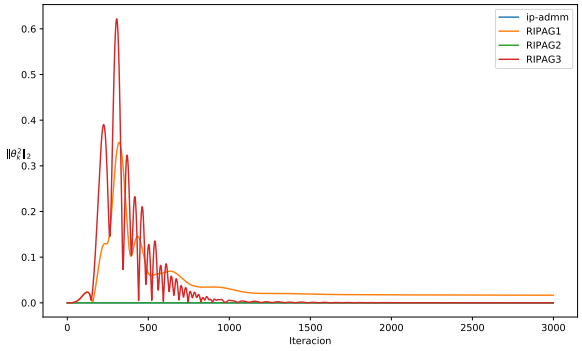

Figura 3.3: Valor de $\|\theta_k^1\|_2$ por iteración.

Podemos evidenciar que para todos los casos los errores θ^{i} convergen a 0, lo cual tiene sentido con las condiciones suficientes pedidas por el Lema 2.5.
También podemos concluir que como todos los algoritmos anteriores se comportan de manera similar podemos usar $(\mathrm{RIPAG})_3$ para poder resolver una familia mucho más amplia de problemas.

3.3.2. Comparación de RIPA generalizado con otro algoritmo especializado.

El problema descrito en (1.1) es ampliamente estudiado en el área de tratamiento de imágenes, por lo que existe una amplia gama de algoritmos para resolverlo, uno de ellos es el algoritmo de Chambolle-Pock [10] el cual tiene la siguiente formulación para este problema

$$\begin{cases} z_k &= (I+\sigma\partial g^*)^{-1}(z_{k-1}+\sigma A\bar{x}_{k-1}) \\ x_k &= (I+\tau\partial f)^{-1}(x_{k-1}+\tau A^t z_k) \\ \bar{x}_k &= x_k+\phi(x_k-x_{k-1}). \end{cases} \tag{3.14}$$

Donde $\sigma,\tau>0$, $\phi\in[0,1]$, $x_0,z_0\in X\times Z$, $\bar{x}_0=x_0$ y $g^*(x^*)=\sup_x\{\langle x^*,x\rangle-g(x)\}$ corresponde a la conjugada de Fenchel de g.
Al implementar este algoritmo para $\phi=1$, $x_0=z_0=\vec{0}$ y distintos valores de σ,τ obtuvimos lo siguiente

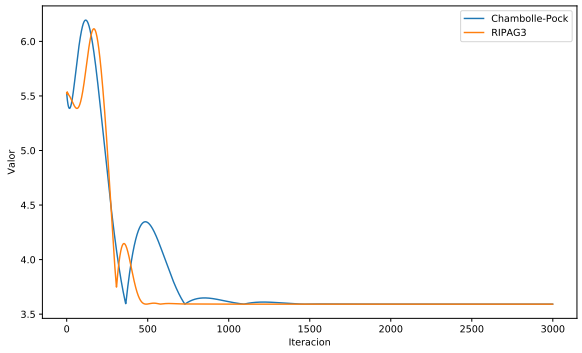

Figura 3.4: Función valor para $\sigma=\tau=0{,}01$.

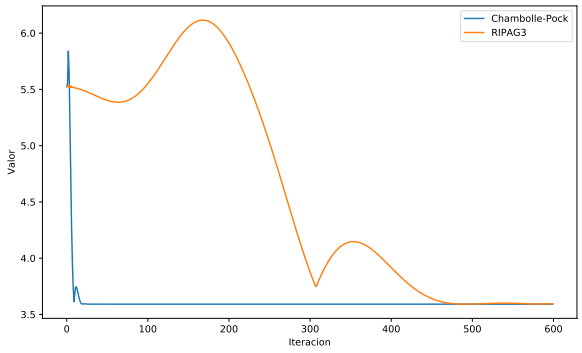

Figura 3.5: Función valor para $\sigma=\tau=0{,}5$.

Otro algoritmo corresponde al **Three-Operator Splitting Scheme** de Davis-Yin [12], el cual es usado para resolver problemas del tipo

$$\min_{\bar{x}\in\bar{X}} f_1(\bar{x})+f_2(\bar{x})+f_3(\bar{x}) \tag{3.15}$$

Donde f_{i} son funciones propias, cerradas, convexas y donde además f_1 es Lipschitz diferenciable.
El algoritmo es el siguiente

$$\begin{cases} z_k &= (I+\gamma\partial f_3)^{-1}(x_{k-1}) \\ y_k &= (I+\gamma\partial f_2)^{-1}(2z_k-x_{k-1}-\gamma\nabla f_1(z_k)) \\ x_{k+1} &= x_k-z_k+y_k \end{cases} \tag{3.16}$$

Donde $\gamma > 0$ y $x_0 \in \bar{X}$.
El problema (1.1) puede ser formulado de esta manera, usando $\bar{X} = X \times Y$, $\bar{x} = (x, y)$, $f_1(x, y) = f(x)$, $g_1(x, y) = g(y)$ y $h(x, y) = \chi_{\{Ax=y\}}(x, y)$. Donde para $C \subseteq \bar{X}$ diremos que la función χ_C es la función indicatriz y satisface

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

Es fácil notar que este problema es equivalente al mencionado en (1.1).

Al implementar este algoritmo para la condición inicial $\bar{x}_0 = (\mathrm{d}, A\mathrm{d})$ y distintos valores de gamma obtuvimos lo siguiente.

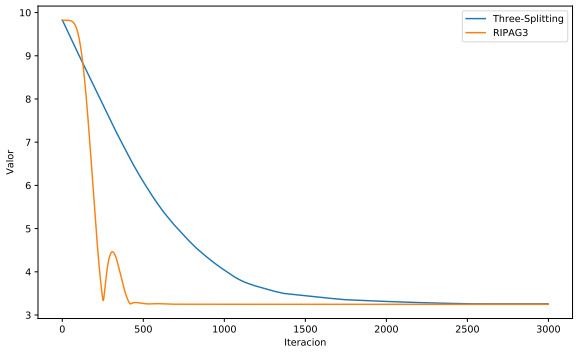

Figura 3.6: Función valor para $\gamma = 5 \cdot 10^{-4}$.

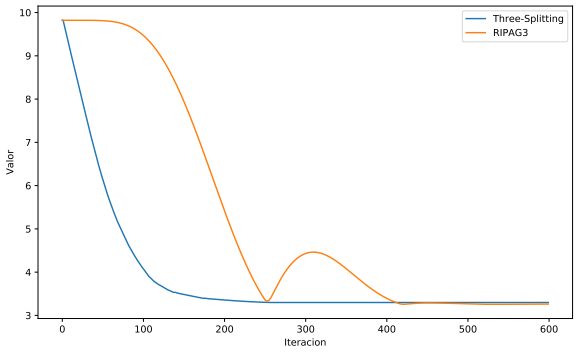

Figura 3.7: Función valor para $\gamma = 5 \cdot 10^{-3}$

Es importante notar que no obtuvimos que el algoritmo convergiera a la solución partiendo de otra condición inicial como la del ejemplo anterior, pero el algoritmo es muy sensible a la elección de parámetros, por lo que es posible que exista un parámetro que consiga la convergencia para otra condición inicial.

3.3.3. Implementación sobre una imagen.

Para esto consideramos una imagen de de 64x64, es decir, la dimensión del problema será de $N = 2^{12}$, la imagen en cuestión es la siguiente

Figura 3.8: Imagen a tratar

Esta imagen puede ser pasada a una matriz de 64x64, para luego ser pasada a un vector de dimension 4096 a través de un isomorfismo entre los espacios $R^{64} \times \mathbb{R}^{64} \to \mathbb{R}^{4096}$, este vector es luego usado como imagen de referencia d para el algoritmo ip-ADMM, en este caso mostraremos la imagen obtenida para distintos valores de θ.

Figura 3.9: $\theta = 1$

Figura 3.10: $\theta = 10$

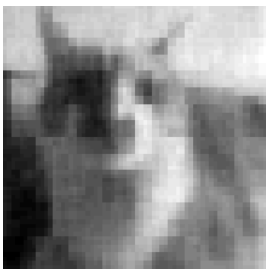

Figura 3.11: $\theta = 20$

Podemos notar que obtuvimos los resultados esperados, para valores grandes de θ la función objetivo va a darle más peso a la parte $\|Ax\|_1$ de la función objetivo, por lo que esperamos obtener una imagen que sea cercana al Kernel del operador A.
En cambio, para valores pequeños de θ le estamos dando más peso a la parte $\|x - \mathrm{d}\|_2^2$, por ende, esperamos obtener una imagen que se parezca más a la imagen de referencia y le de poco peso a minimizar la norma de la aplicación del operador a la imagen.

Conclusión

Como se planteó desde un inicio, en esta tesis se presentaron dos objetivos concretos: encontrar condiciones suficientes para la convergencia de los algoritmos y estudiar las implementaciones de estos.

El primer objetivo se logró con éxito, encontrando que la principal condición que deben satisfacer los algoritmos pertenecientes a la familia de RIPA generalizado es que sus errores respectivos con respecto a algoritmo RIPA, correspondiente a los términos $\theta^1 = r^1 - r$ y $\theta^2 = r^2 - r$ converjan a 0, esto nos da un marco de referencia para poder estudiar el correcto funcionamiento de los algoritmos pertenecientes a esta familia.

El segundo objetivo también fue conseguido: Se presentaron posibles problemas que surgen de implementar el algoritmo (ip-ADMM) y, como la familia de algoritmos RIPA generalizado ofrecen posibles soluciones a este, también se logró comparar este algoritmo con implementaciones de los algoritmos de Chambolle-Pock y Three-Operator Splitting Scheme de Davis-Yin, mostrando que el algoritmo ip-ADMM tiene eficacia y eficiencia similar a estos algoritmos.

Una posible continuación de este trabajo consistiría en no solo encontrar condiciones suficientes de convergencia para los algoritmos de RIPA generalizado, si no también encontrar un resultado general que demuestre la convergencia de estos algoritmos en base a los parámetros usados por éste, sin tener que recurrir a la sucesión generada por éste, además de encontrar la velocidad de convergencia de estos.

Bibliografía

[1] Felipe Alvarez and Hedy Attouch. An inertial proximal method for maximal monotone operators via discretization of a nonlinear oscillator with damping. *Set-Valued Analysis*, 9:3–11, 01 2001.

[2] Hedy Attouch and Paul-Emile Mainge. Asymptotic behavior of second-order dissipative evolution equations combining potential with non-potential effects, 2009.

[3] Hedy Attouch and Juan Peypouquet. Convergence of inertial dynamics and proximal algorithms governed by maximally monotone operators. 2017.

[4] Hedy Attouch and Juan Peypouquet. Convergence rate of proximal inertial algorithms associated with moreau envelopes of convex functions. In *Splitting Algorithms, Modern Operator Theory, and Applications*, pages 1–44. Springer, 2019.

[5] Hedy Attouch and Mohamed Soueycatt. Augmented lagrangian and proximal alternating direction methods of multipliers in hilbert spaces. applications to games, pde's and control. 2008.

[6] Heinz H Bauschke, Patrick L Combettes, et al. *Convex analysis and monotone operator theory in Hilbert spaces*, volume 408. Springer, 2011.

[7] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine learning*, 3(1):1–122, 2011.

[8] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, USA, 2004.

[9] Haim Brezis. *Operateurs maximaux monotones et semi-groupes de contractions dans les espaces de Hilbert*. Elsevier, 1973.

[10] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. working paper or preprint, June 2010.

[11] Gong Chen and Marc Teboulle. A proximal-based decomposition method for convex minimization problems. *Mathematical Programming*, 64(1):81–101, 1994.

[12] Damek Davis and Wotao Yin. A three-operator splitting scheme and its optimization

applications, 2015.

[13] Juan Peypouquet. *Convex optimization in normed spaces: theory, methods and examples.* Springer, 2015.

[14] R. T. Rockafellar. On the maximal monotonicity of subdifferential mappings. *Pacific Journal of Mathematics*, 33(1):209 – 216, 1970.