ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE CIENCIAS

MÉTODOS DE SEGUNDO ORDEN PARA LA RESOLUCIÓN NUMÉRICA DE PROBLEMAS DE OPTIMIZACIÓN DISPERSOS

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERA MATEMÁTICA

KAREN ESTEFANÍA LOAYZA ROMERO
eloayza16@gmail.com

Director: DR. PEDRO MARTÍN MERINO ROSERO
pedro.merino@epn.edu.ec

QUITO, NOVIEMBRE 2014

DECLARACIÓN

Yo KAREN ESTEFANÍA LOAYZA ROMERO, declaro bajo juramento que el trabajo aquí escrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual, correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normatividad institucional vigente.

Karen Estefanía Loayza Romero

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por KAREN ESTEFANÍA LOAYZA ROMERO, bajo mi supervisión.

Dr. Pedro Martín Merino Rosero
Director del Proyecto

AGRADECIMIENTOS

A Pedro Merino y Juan Carlos De los Reyes por la confianza depositada en mi, y por permitirme ser parte de su grupo. Al equipo del MODEMAT por hacer que cada día de trabajo sea agradable.

DEDICATORIA

A mis padres y hermanos quienes me impulsan a ser mejor cada día.

Índice general

Resumen

En este trabajo se propone un algoritmo de segundo orden para resolver numéricamente problemas de optimización dispersos inducidos por la norma ℓ_1. La idea principal de este algoritmo es incluir información de segundo orden asociada al término no diferenciable. El cálculo de las direcciones de descenso se basa en los esquemas de los algoritmos iterativos de descenso tipo Quasi–Newton, donde la matriz Hessiana es enriquecida usando información que se la obtiene a través de la regularización de Huber de la norma ℓ_1. Existen dos estrategias adicionales importantes en la formulación de este tipo de algoritmos. Las direcciones ortantes, las cuales miden la contribución de la norma ℓ_1 en la dirección de descenso, además de definir ortantes en los cuales la función objetivo es diferenciable. El paso de descenso, el cuál mide cuánto nos movemos a lo largo de la dirección de descenso, y se lo obtiene a través de una metodología de búsqueda lineal proyectada. Gracias a la combinación de estas tres estrategias el algoritmo propuesto adquiere características importantes con respecto a la rapidez en la detección de los conjuntos activos, la generación de dirección de descenso en todas sus iteraciones, entre otras. Presentamos algunas propiedades teóricas del algoritmo desarrollado y conducimos experimentos numéricos exhaustivos que revelan un mejor desempeño del algoritmo propuesto en este trabajo con respecto a algoritmos similares desarrollados previamente. Además aplicamos nuestro algoritmo en la resolución numérica de problemas prácticos, como un problema de dinámica poblacional modelado con la ecuación de Fisher.

Abstract

In this paper we propose a second-order orthant-based method to solve numerically sparse problems that involve ℓ_1 norm in their objective function. The main idea of the algorithm is to incorporate the second order information of the non-differentiable term. The computation of the descent direction is made in the context of the Quasi–Newton iterative descent methods, where the Hessian information is enriched using the Huber regularization of the ℓ_1 norm. There are two additional strategies in the context of this methods. The orthant directions, whose principal aim is to measure the contribution of the ℓ_1 norm in the descent direction, and to define orthants in which the objective function is differentiable. The step length, who measures how far we are going to move along this descent directions, this step's calculation is made by using a projected line–search procedure. Thanks of the combination of this three strategies our algorithm acquires some important theoretical properties, like the faster active sets identification and the generation of descent directions in all of the algorithm's iterations. We present this theoretical features and also we conduce exhaustively some numerical experiments from which we can infer a better performance of our algorithm than the other similar algorithms proposed before. Additionally, we apply our algorithm in the numerical solution of a population dynamics problem modeled via the Fisher's equation.

Capítulo 1

Introducción

En la actualidad, los problemas de reconocimiento de voz *c.f.* [4], reconocimiento de patrones utilizados para la identificación de correo electrónico no deseado, reconocimiento de caracteres, el problema de interfaz "Cerebro–Computador" [21], entre otras son aplicaciones que involucran problemas de optimización dispersos cuya resolución numérica necesita de algoritmos eficientes. Los problemas de optimización asociados a estas aplicaciones tienen una estructura similar, buscan minimizar un funcional el cuál mide la probabilidad de clasificación incorrecta de un dato en el caso del reconocimiento de patrones o caracteres; o buscan un patrón en común de tal manera que se pueda identificar que señales son más aceptadas para el caso del problema "Cerebro–computador".

Además, en aplicaciones de problemas de control óptimo con ecuaciones en derivadas parciales, es importante encontrar una estrategia de control óptima la cuál tenga una acción localizada, debido a que en la práctica es menos costoso aplicar este tipo de estrategias que aquellas que se aplican en todo el territorio en estudio. Dentro de este grupo de problemas podemos incluir aquellos de control de dinámica poblacional, donde el objetivo es controlar de manera óptima la tasa de mortalidad de cierta especie. La estrategia de control para este problema específico puede ser la aplicación de un insecticida o bactericida, el cuál provoca un aumento en la tasa de mortalidad natural de la especie. Una acción localizada del control en este caso, quiere decir que el insecticida debe ser aplicado en sectores específicos del territorio considerado. Matemáticamente, la acción localizada se traduce en que la función que representa el control tiene un soporte pequeño. En problemas de dimensión finita, esta característica puede ser representada como vectores que tiene gran cantidad de componentes iguales a cero. En ambos casos, función (en dimensión infinita) o vector (en dimensión finita) esta característica especial se conoce como "dispersión" o "sparsity" en inglés. La dispersión se ilustra mediante el principio de parsimonia; el cuál dice: "La explica-

ción más simple que se puede dar a un fenómeno será preferida sobre una explicación más complicada".

Existen varios términos que inducen dispersión en una solución de un problema de optimización. Por ejemplo están: la norma ℓ_1, las normas mixtas $\ell_1 - \ell_2$, $\ell_1 - \ell_\infty$ [33] y además las normas ℓ_q con $q < 1$. En este trabajo nos concentramos en el estudio de aquellos problemas que involucran únicamente la norma ℓ_1.

En términos generales el problema en estudio tiene la siguiente forma

$$\min_x \ \varphi(x) = f(x) + \beta \parallel x \parallel_{\ell_1}, \tag{1.1}$$

donde f es una función diferenciable (regular) y $\| \cdot \|_{\ell_1}$ es el término no diferenciable que induce dispersión en las soluciones. En la sección correspondiente se formula el problema formalmente y se presentan sus principales hipótesis.

El efecto de la norma ℓ_1 en las soluciones puede ser explicado como sigue: Teniendo en cuenta que queremos resolver un problema de minimización, debemos notar el tipo de penalización que representa la norma ℓ_1, para compararla con la norma ℓ_2. Si tuviésemos un problema de minimización de una dimensión, la forma de ambas penalizaciones cuando los valores son cercanos a 0 se muestra en la Figura 1

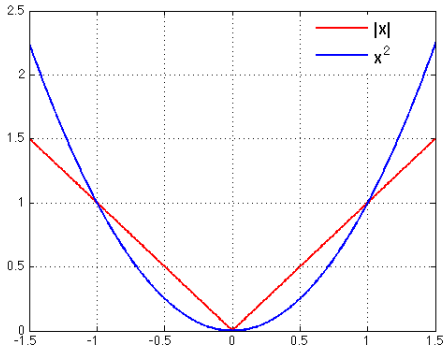


Figura 1. Penalización de la norma ℓ_1 (rojo) y ℓ_2 (azul) para valores en el intervalo $[-1, 1]$.

Es claro que para valores cercanos a 0 la penalización correspondiente a la norma ℓ_1 es mucho mayor que la de la norma ℓ_2. Por lo tanto, cualquier algoritmo de minimización busca fijar los valores a cero, para de esta manera reducir el valor de la función objetivo; mientras que, en el caso de la norma ℓ_2, se admiten valores distintos a cero pero pequeños, sin que el valor de la función objetivo se vea afectado significativamente.

Una explicación geométrica de por qué la norma ℓ_1 induce soluciones sparse consi-

dera el problema (1.1) reformulado como un problema con restricciones de la siguiente manera:

$$\begin{aligned} &\min_{x} \quad f(x) \\ &\text{s.a.} \\ &\qquad \| x \|_1 \leq T, \end{aligned} \tag{1.2}$$

donde T es un parámetro que depende directamente de β. La representación geométrica del conjunto factible del problema (1.2) en $\mathbb{R}^2$ se muestra en la Figura 2.

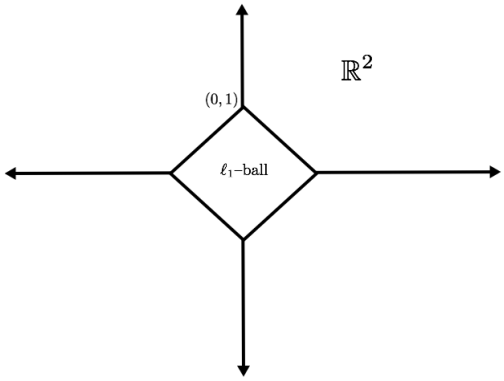


Figura 2. Conjunto factible del problema (1.2)

Si tomamos en cuenta que la proyección de un punto sobre el conjunto factible es la distancia más cercana al conjunto. En la Figura 3 se muestra la proyección de los puntos sobre el conjunto factible, y es claro notar que existe una gran cantidad de puntos cuya proyección coincide con el vértice $(0,1)$ por este motivo se puede decir que este norma induce a un comportamiento de dispersión en las soluciones.

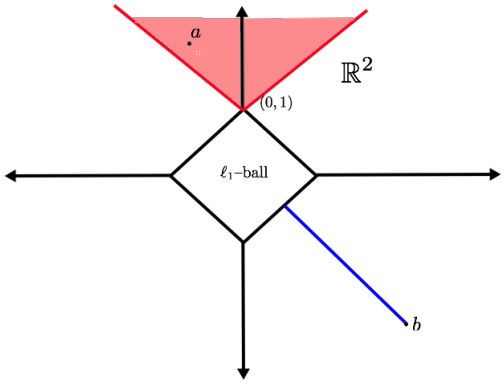


Figura 3. Proyección sobre el conjunto factible. Conjunto rojo corresponde a los puntos cuya proyección sobre el conjunto factible coincide con el vértice del conjunto, puntos azules corresponden a los puntos cuya proyección no es una solución dispersa.

A manera de comparación si en vez de la norma ℓ_1 tuviéramos un problema cuyo conjunto factible es la bola de la norma ℓ_2. En la Figura 4 podemos observar la diferen-

cia entre la proyección sobre la norma ℓ_1 (azul continua) y la proyección sobre la bola ℓ_2 (azul entrecortada).

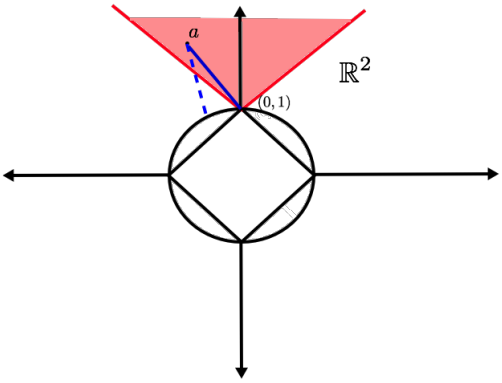


Figura 4. Proyección sobre la bola ℓ_2 y ℓ_1, comparación

De este gráfico podemos concluir que así como en la explicación anterior, con la norma ℓ_2 se admiten valores cercanos a 0 pero no necesariamente 0, sin tener un cambio significativo en la función objetivo.

La principal dificultad que presenta la resolución numérica de este tipo de problemas consiste en que el término asociado a la norma ℓ_1 no es diferenciable. Por lo tanto, no se pueden aplicar algoritmos usuales de optimización, los cuales asumen diferenciabilidad de la función objetivo.

Existe una gran gama de algoritmos que nos permiten resolver estos problemas, por ejemplo algoritmos que consideran información del subgradiente [13, 3], algoritmos que consideran regularizaciones [12], y algoritmos que resuelven el problema equivalente mostrado en (1.2) [19]. Sin embargo, no existe mucho desarrollo de métodos numéricos cuando se trata de problemas funcionales como es el caso de control óptimo de ecuaciones diferenciales.

En el caso de los algoritmos desarrollados para resolver problemas en dimensión finita se pueden clasificar en tres grandes grupos cuyas principales características se presentan a continuación:

Estrategias de sub–gradiente .- La norma ℓ_1 no es diferenciable, sin embargo, su subdiferencial está bien definido. La principal idea de esta clase de algoritmos es utilizar los subgradientes para el cálculo de las direcciones de descenso.

Estrategias de aproximación .- Estos algoritmos reemplazan el funcional objetivo no diferenciable, por una función que si lo sea pero que aproxime a la original en algún sentido. En general, esta nueva función depende de un parámetro que caracteriza la calidad de dicha aproximación. Generalmente, a estas funciones se

las conoce como *regularizaciones*. La inclusión de estas funciones, evitan la consideración de nuevas restricciones; en consecuencia, el problema resultante es un problema diferenciable sin restricciones. Sin embargo, es claro que la regularización tampoco puede ser excesivamente suave pues se pierden características importantes del problema.

Formulación con restricciones .- En este caso, el problema se reformula reemplazando la función no diferenciable por una que si lo sea, esta función no aproxima necesariamente a la original. A cambio, se incluye una restricción inducida por el término no diferenciable de la función objetivo. Los algoritmos utilizados para resolver estos problemas son distintos pues, el problema a resolverse tiene restricciones. Aunque en este caso se minimiza una función diferenciable sobre un convexo cerrado, el conjunto factible tiene esquinas, debido a la no diferenciabilidad de las restricciones. Por tanto existen ciertos inconvenientes en la resolución de los problemas restringidos, por lo que una de las estrategias es la de separar dichas restricciones; en la parte positiva y negativa de las funciones que las describen.

Para una información más detallada sobre las estrategias mencionadas anteriormente, léase [31].

En este trabajo, proponemos un nuevo algoritmo para resolver problemas de optimización convexa, del tipo (1.1), basado en métodos de descenso y la idea poderosa de direcciones ortantes.

Nuestra contribución consiste en incluir información de segundo orden correspondiente al término no diferenciable, usando una regularización de tipo Huber. Esta consideración proporciona información adicional para el cálculo de la direcciones de descenso en comparación a aquellas de los algoritmos NW–CG *c.f.* [3] y OWL *c.f.* [13], los cuales nos servirán de referencia para comparar el desempeño de los desarrollados en este trabajo.

La utilización de esta regularización se debe a su naturaleza local y no excesivamente suave. De hecho, ésta es una vez diferenciable, por lo que es adecuada para capturar la mayor cantidad de información correspondiente a la estructura de la norma ℓ_1. Por tanto, para obtener la información de segundo orden se utiliza subgradientes. Adicionalmente, utilizamos la noción de direcciones ortantes las cuales permiten detectar de manera más eficiente aquellos puntos del dominio en los cuales el control será exactamente cero, es decir, los conjuntos activos. Además, garantizan el descenso de la función objetivo en todas las iteraciones del algoritmo, como discutiremos más adelante.

1. Notaciones

A lo largo de este trabajo utilizaremos la siguiente notación. Las normas correspondientes a los espacios funcionales L^1 y L^2 serán notadas por $\| \cdot \|_{L^1}$ y $\| \cdot \|_{L^2}$ respectivamente. Mientras que la norma del espacio ℓ_1 será notada por $\| \cdot \|_1$ y $\| \cdot \|_2$ respectivamente. Además, el gradiente de una función diferenciable en x se nota por $\nabla f(x)$, y su i–ésima componente por $\nabla_i f(x)$. Para indicar las iteraciones de los algoritmos se utilizarán super–índices, y para cada componente de los vectores sub–índices. Es decir, la notación x_i^k hace referencia a la componente i–ésima de la k–ésima iteración de algún algoritmo. Las matrices se denotarán con letras mayúsculas y los vectores con letras minúsculas. El producto escalar usual en $\mathbb{R}^n$ se nota por $(\cdot, \cdot)$. En la formulación dual del problema se utilizan ciertos espacios especiales cuya notación es presentada a continuación. El espacio de las funciones lineales definidas entre V y Y está notado por $\mathcal{L}(V, Y)$. Y notamos por $\mathcal{L}(Y^*, V^*)$ al espacio de las funciones lineales definidas en Y^* a V^* con Y^* y V^* los espacio duales asociados a Y, V respectivamente. Los elementos de estos espacios serán notados por Λ y Λ^*, respectivamente. Se nota por Λ^* al operador adjunto de Λ.

El esquema de este trabajo, es como sigue: En el capítulo 1 se presenta la formulación de los problemas con dispersión y sus principales características. Además, incluimos las herramientas del análisis convexo necesarios para la formulación de los algoritmos. En el capítulo 2, se presentan los algoritmos que utilizan información parcial de segundo orden para el cálculo de la dirección de descenso. En el capítulo 3, se presenta el algoritmo desarrollado en este trabajo, que incluyen información enriquecida de segundo orden y mencionamos sus principales características teóricas. El capítulo 4 es una recopilación de test numéricos, en los que comparamos el desempeño de nuestro algoritmo con los propuestos previamente en la literatura existente. Finalmente mostramos una aplicación a problemas más complejos de simulación numérica. En el capítulo 5 exponemos las conclusiones de este trabajo.

2. Problemas con dispersión

Estamos interesados en la resolución numérica de problemas de optimización en $\mathbb{R}^n$ que incluyen la norma ℓ_1 en su funcional de costo. Específicamente, consideramos problemas de la forma:

$$\min_{x \in \mathbb{R}^n} \quad \varphi(x) = f(x) + \beta \| x \|_1 \tag{1.3}$$

donde f es una función al menos una vez continuamente diferenciable, llamaremos a f la parte regular de la función de costo, y $\| x \|_1 := \sum_{i=1}^{n} |x_i|$.

En cada una de las aplicaciones mencionadas en la introducción, se resuelven problemas que pueden ser expresados como en (1.3). Dichos problemas se diferencian únicamente en la expresión correspondiente a la parte regular. La constante $\beta > 0$ se puede entender como el parámetro de *dispersión*, el cuál mide cuán dispersa será la solución. Es decir, la cantidad de componentes nulas que ésta tendrá. Algunas posibles elecciones de la forma de la parte regular f para los problemas mencionados anteriormente son:

- $f(x) = \|Ax - y\|_2^2$, donde A es una matriz en $\mathbb{R}^{n \times m}$. Este problema de regresión lineal dispersa recibe el nombre de LASSO por sus siglas en inglés (Least Absolute Shrinkage and Selection Operator). Es una técnica de regresión lineal que se basa en reducir el tamaño de ciertos operadores y fijar otros en cero, pero manteniendo las características importantes del problema [22, 35].

- $f(x) = \|A^{-1}x - y\|_2^2 + \frac{\alpha}{2}\|x\|_2^2$. Esta función aparece en los problemas lineal cuadráticos de Ecuaciones Diferenciales Parciales, después del proceso de discretización. Este funcional incluye el término adicional correspondiente a la regularización de Tikhonov ℓ_2 para garantizar la existencia de soluciones [34, 6].

- $f(x) = -\frac{1}{N} \sum_{j=1}^{N} \log \frac{\exp(x_{y_j}^T z_j)}{\sum_{i \in C} \exp(x_i^T z_j)}$. Esta función aparece en los problemas de reconocimiento de voz, donde N es el número de muestras usados para el reconocimiento, C es el conjunto de las etiquetas de cada clase, y_j es la etiqueta asociada a cada punto de entrenamiento j, z_j es el vector de características y x_i es el sub–vector de parámetros de la etiqueta i. Esta función representa la suma normalizada del logaritmo negativo de la función de verosimilitud de cada dato para ser clasificado de forma correcta [7, 24].

3. Regularización de Huber

Debido a que la norma ℓ_1 no es diferenciable en el origen, no es posible considerar información de segundo orden de esta función, la cuál está asociada a su curvatura. Por tanto, para poder obtener la información de "segundo orden" asociada a este término es necesario usar una regularización. Una regularización es una función diferenciable, la cual aproxima a un término no diferenciable, en general depende de un parámetro de regularización asociado. Existen varios tipos de regularizaciones; en este

trabajo utilizamos la regularización de Huber, la cuál ha mostrado excelentes resultados en la resolución numérica de diferentes tipos de problemas, por ejemplo [9, 27]. Y como se ha mencionado anteriormente al no ser excesivamente suave y local permite capturar mejor la estructura de la norma $\|\cdot\|_1$.

La regularización de Huber es una regularización local. Es decir sólo se modificará en una vecindad del punto en conflicto que en este caso ocurre cuando $x = 0$.
Para $\gamma > 0$, la regularización de Huber del valor absoluto se define como:

$$\begin{aligned} H_\gamma(\cdot): \quad & \mathbb{R} \to \mathbb{R} \\ & x \mapsto \quad H_\gamma(x) = \begin{cases} \gamma\frac{x^2}{2} & \text{si } |x| \leq \frac{1}{\gamma}, \\ |x| - \frac{1}{2\gamma} & \text{si } |x| > \frac{1}{\gamma}. \end{cases} \end{aligned} \tag{1.4}$$

En la Figura 6 se muestra la regularización de Huber y el valor absoluto; en donde se puede observar el tipo de aproximación local en el origen cuando el parámetro de regularización es suficientemente alto.

La norma ℓ_1 está definida de la siguiente manera:

$$\begin{aligned} \|\cdot\|_1: \quad & \mathbb{R}^n \to \mathbb{R} \\ & x \mapsto \quad \|\ x\ \|_1 = \sum_{i=1}^{n} |x_i| \end{aligned}$$

La regularización de la norma $\|\cdot\|_1$ basada en Huber se define de la siguiente manera:

$$\begin{aligned} \mathcal{G}(\cdot): \quad & \mathbb{R}^n \to \mathbb{R} \\ & u \mapsto \quad \mathcal{G}(x) = \sum_{i=1}^{n} H_\gamma(x_i) \end{aligned}$$

donde $H_\gamma(x_i)$ está definido en (1.4).

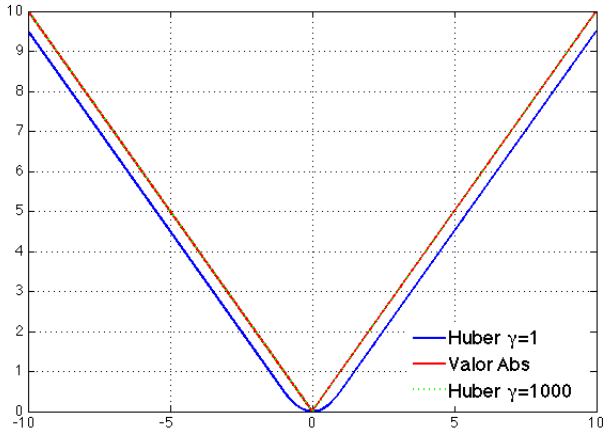
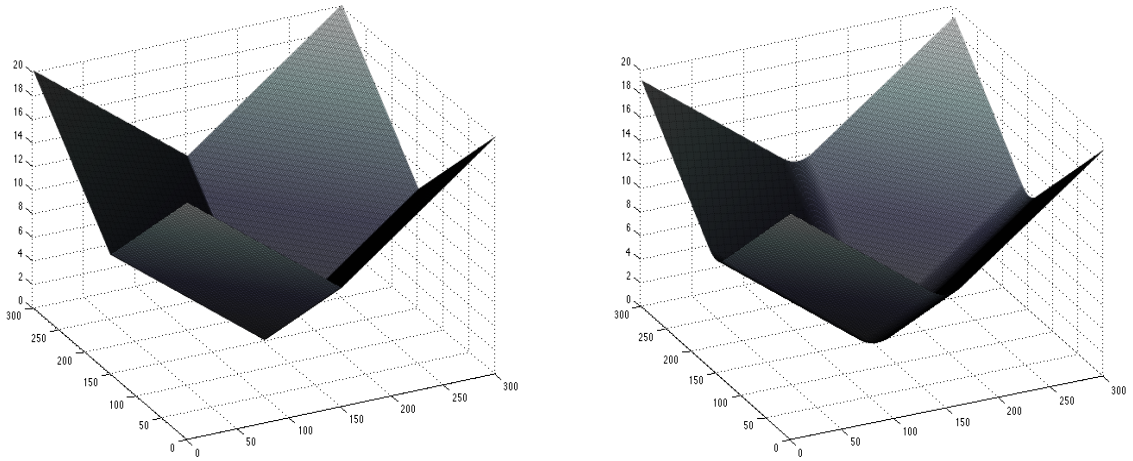


Figura 5. Valor absoluto y regularización tipo Huber 1–dimensional.



(a) $\|\cdot\|_1$ bi–dimensional. **(b)** Regularización tipo Huber bi–dimensional

Figura 6. $\|\cdot\|_1$ en 2D y regularización tipo Huber con $\gamma = 1$.

Propiedades de la regularización de Huber

La primera derivada de la función $H_\gamma(x)$ con respecto a x está dada por la siguiente expresión:

$$\partial H_\gamma(x) = \frac{\gamma x}{\text{máx}\{1, \gamma|x|\}}, \tag{1.5}$$

donde $\text{máx}\{1, \gamma|x|\} = \begin{cases} 1 & \text{si } 1 \geq \gamma|x|, \\ \gamma|x| & \text{caso contrario.} \end{cases}$

En la Figura 7 se muestra la forma que tiene la derivada de $H_\gamma(x)$. El valor del gra-

diente de la función $\mathcal{G}(x)$, está dado por el vector cuyas componentes son:

$$\nabla \mathcal{G}(x) = \left(\frac{\gamma x_i}{\text{máx}\{1, \gamma |x_i|\}} \right)_{i=1,\ldots,n}.$$

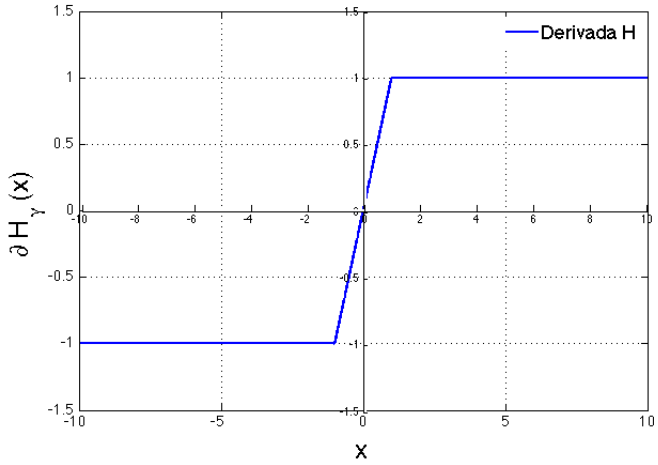


Figura 7. Derivada de la Regularización con $\gamma = 1$.

De la Figura 7 podemos notar que la derivada de la regularización $\partial H_\gamma(x)$ no es diferenciable. Sin embargo la noción más general de subgradiente nos permitirá obtener la información de segundo orden requerida.

DEFINICIÓN 1.1. *Sea $f : \mathbb{R}^n \to \mathbb{R}^n$. Un elemento $z \in \mathbb{R}^n$ se dice un subgradiente de f en x_0 si*

$$f(x) - f(x_0) \geq (x - x_0, z) \tag{1.6}$$

para todo $x \in \mathbb{R}^n$. Se define como el subdiferencial al conjunto que contiene a todos los subgradientes de f en x_0, y se lo nota por $\partial(f(x_0))$.

En el caso de una función diferenciable, su subdiferencial es el conjunto unitario que contiene la derivada. La función valor absoluto, es un caso particular de una función no diferenciable en 0, sin embargo, su subdiferencial está bien definido. Lo mismo ocurre con la función del máximo.

A continuación presentamos dos ejemplos de subgradientes, cuya demostración se encuentra en el apéndice. Estos ejemplos, serán utilizados posteriormente en la construcción de los algoritmos.

Geométricamente, el subgradiente es el conjunto de todos los hiperplanos los cuales se ubican por debajo de la función, en el siguiente ejemplo se mostrará este hecho.

EJEMPLO 1. Función valor absoluto, está definida por

$$\begin{aligned} |\cdot| : \ & \mathbb{R} \to & \mathbb{R} \\ & x \mapsto & |x|, \end{aligned}$$

Sea z un elemento del subdiferencial $z \in \partial(|x|) = \begin{cases} 1 & \text{si } x > 0, \\ -1 & \text{si } x < 0, \\ [-1,-1] & \text{si } x = 0. \end{cases}$

En particular tomemos

$$z = \begin{cases} -1 & \text{si } x < 0, \\ 1 & \text{si } x \geq 0. \end{cases}$$

El subdiferencial de la función valor absoluto se ilustra en la Figura 8

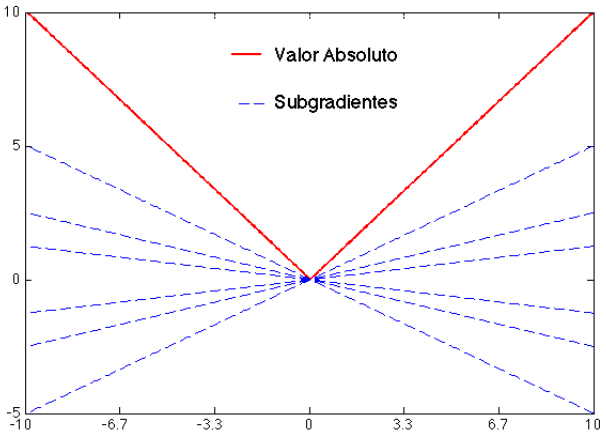


Figura 8. Subgradientes de la función valor absoluto

EJEMPLO 2. En este ejemplo, consideramos el subdiferencial de la función máximo definida por:

$$\begin{aligned} \text{máx}\{0,\cdot\} : \ & \mathbb{R} \to & \mathbb{R} \\ & x \mapsto & \text{máx}\{0,x\} \end{aligned}$$

Sea χ elemento del subdiferencial, $\chi \in \partial\left(\text{máx}\{0,x\}\right)$, definido por

$$\partial\left(\text{máx}\{0,x\}\right) = \begin{cases} 1 & \text{si } x > 0, \\ 0 & \text{si } x < 0, \\ [0,1] & \text{si } x = 0. \end{cases}$$

Tomamos un elemento en especial χ:

$$\chi = \begin{cases} 0 & \text{si } x < 0, \\ 1 & \text{si } x \geq 0. \end{cases} \tag{1.7}$$

Utilizando los ejemplos anteriores procedemos con el cálculo de la segunda derivada de la regularización de Huber, utilizando la regla de la cadena, y ciertos resultados del análisis convexo. Todos los teoremas y definiciones a los cuales se hace referencia en esta demostración se encuentran en el apéndice.

Lema 1. *Sea* $f : \mathbb{R} \to \mathbb{R}$*, tal que* $h(x) = \dfrac{\gamma x}{\text{máx}\{1, \gamma|x|\}}$*. Sea* $\Psi \in \partial\left(h(x)\right)$*, entonces* Ψ *está dado por:*

$$\Psi = \frac{\gamma}{\text{máx}\{1, \gamma|x|\}} - \frac{\gamma^2 \, |x| \, \chi}{\left(\text{máx}\{1, \gamma|x|\}\right)^2}.$$

Demostración. Del Teorema 5 del apéndice podemos aplicar la regla del producto de subgradientes de la siguiente manera:

$$\partial\left(\gamma x \cdot \frac{1}{\text{máx}\{1, \gamma|x|\}}\right) = \frac{\gamma}{\text{máx}\{1, \gamma|x|\}} + \gamma \, x \cdot \partial\left(\frac{1}{\text{máx}\{1, \gamma|x|\}}\right),$$

Del Corolario 1 del apéndice podemos calcular el subgradiente del último término en la expresión anterior.

$$\partial\left(\frac{1}{\text{máx}\{1, \gamma|x|\}}\right) = -\frac{\partial(\text{máx}\{1, \gamma|x|\})}{\left(\text{máx}\{1, \gamma|x|\}\right)^2}.$$

Por último, teniendo en cuenta que del Teorema 7, la función $\text{máx}\{1, \gamma|x|\}$ es semi–suave, definición presentada en 6.1, podemos calcular un subgradiente de esta función a través del uso de la derivada de Newton usando el Teorema 8:

$$\partial\left(\text{máx}\{1, \gamma|x|\}\right) = \gamma \, \chi \, z$$

donde $\chi \in \partial\left(\text{máx}\{1, \gamma|x|\}\right)$ y $z \in \partial\left(|x|\right)$. Además cabe recalcar que $z = \text{sign}(x)$ y que $x \cdot z = |x|$, de donde se obtiene el resultado deseado. □

Con el resultado anterior podemos definir la matriz que representa la información de segundo orden de un vector $x \in \mathbb{R}^n$. Entonces, definimos $\Gamma := \nabla\left(\nabla \mathcal{G}(\cdot)\right)$, con

entradas

$$(\Gamma)_{ij} = \frac{\partial}{\partial x_j}\left(\nabla_i \mathcal{G}(x)\right),$$
$$= \begin{cases} 0 & \text{si } i \neq j, \\ \dfrac{\gamma}{\text{máx}\{1, \gamma|x_i|\}} - \dfrac{\gamma^2 |x_i| \chi_i}{(\text{máx}\{1, \gamma|x_i|\})^2} & \text{caso contrario,} \end{cases}$$

para $i, j = \{1, 2, \ldots, n\}$.

Entonces Γ es una matriz diagonal, donde χ_i está dada por:

$$\chi_i = \begin{cases} 0 & \text{si } \gamma|x_i| < 1, \\ 1 & \text{caso contrario.} \end{cases}$$

Si analizamos por casos los valores de la diagonal de Γ, tenemos:

CASO I $\gamma|x_i| \leq 1$

$$(\Gamma)_{ii} = \gamma.$$

CASO II $\gamma|x_i| > 1$

$$(\Gamma)_{ii} = \frac{\gamma}{\gamma|x_i|} - \frac{\gamma^2 x_i^2}{\gamma^2 x_i^2 |x_i|} = 0.$$

Por lo tanto, podemos escribir Γ de manera más compacta como:

$$(\Gamma)_{ii} = \begin{cases} \gamma & \text{si } \gamma|x_i| \leq 1, \\ \\ 0 & \text{caso contrario.} \end{cases} \tag{1.8}$$

Puesto que $\gamma > 0$, esta matriz es semi–definida positiva.

3.1. Regularización del problema

Una vez definida la regularización de Huber, se puede formular un problema regularizado, que aproxima al problema (1.3), reemplazando la norma ℓ_1 por la regularización de Huber. Se considera el problema regularizado de (1.3) como:

$$\min_{x \in \mathbb{R}^n} \quad \varphi_\gamma(x) = f(x) + \beta \sum_{i=1}^{n} H_\gamma(x_i) \tag{1.9}$$

donde $H_\gamma(x_i)$ es la regularización de Huber de $\|\cdot\|_1$ con parámetro de regularización γ, definida en (1.4). Nótese que (1.9) es un problema diferenciable; y por tanto podemos resolverlo mediante algoritmos usuales de optimización en dimensión finita.

4. Herramientas del análisis convexo

Esta sección está dedicada al estudio de herramientas correspondientes al análisis convexo, necesarias para la formulación de los algoritmos. Varias definiciones y resultados del análisis convexo son importantes para nuestros propósitos, especialmente debido a la no diferenciabilidad de la función objetivo *c.f.* [16] entre otras.

4.1. Formulación dual

En esta sección discutimos la formulación dual del problema (1.3). Para dicho propósito, se utilizará la teoría de dualidad de Fénchel [10]. El análisis realizado en este trabajo es similar al propuesto en [15]. El principal interés de este estudio es la demostración de existencia de multiplicadores de Lagrange. Además, de la formulación de las condiciones de optimalidad en términos del la variable dual; para de esta forma obtener las condiciones de optimalidad presentadas en [34].

Las definiciones y principales resultados sobre este tema han sido tomados de [10] y son presentados en el apéndice.

Condiciones de optmalidad de primer orden

El problema (1.3) tiene asociada la siguiente condición de optimalidad de primer orden, la cual es necesaria y suficiente debido a la convexidad de la función objetivo.

Lema 2. *Sea $\bar{x}$ un mínimo local del problema* (1.3) *se satisface que*

$$0 \in \partial\left(\varphi(\bar{x})\right)$$

donde $\partial\left(\varphi(\bar{x})\right)$ hace referencia al subdiferencial de la función objetivo.

Del Teorema de Moreau–Rockafellar [18], puesto que ambas funciones son continuas y f es diferenciable podemos concluir que, la condición de optimalidad está dada por:

$$-\nabla f(\bar{x}) \in \partial\left(\beta \parallel \bar{x} \parallel_1\right).$$

El análisis del problema dual nos permite encontrar una equivalencia entre la condición de optimalidad y la siguiente expresión:

$$\begin{aligned} \nabla f(\bar{x}) + \beta = 0 \quad & \text{si } \bar{x} > 0 \\ \nabla f(\bar{x}) - \beta = 0 \quad & \text{si } \bar{x} < 0 \\ 0 \in [\nabla f(\bar{x}) - \beta, \nabla f(\bar{x}) + \beta] \quad & \text{si } \bar{x} = 0 \end{aligned}$$

Debemos notar que la última relación es componente a componente.

Problema Dual

En términos generales, el Teorema 9 del apéndice muestra la existencia de un multiplicador. Antes de aplicar estos resultados a nuestro problema es necesario notar que la estructura de (1.3) nos permite descomponerlo en un problema que tiene la siguiente forma:

$$\inf_{x \in V} \mathcal{F}(x) + \mathcal{G}(\Lambda x), \tag{1.10}$$

con $\mathcal{F} : \mathbb{R}^n \to \mathbb{R}$ y $\mathcal{G} : \mathbb{R}^n \to \mathbb{R}$. Entonces el problema dual está dado por:

$$\sup_{q^* \in \mathbb{R}^n} \; -\mathcal{F}^*(-\Lambda^* q^*) - \mathcal{G}^*(q^*), \tag{1.11}$$

donde $\mathcal{F}^*$ y $\mathcal{G}^*$ son las funciones conjugadas de $\mathcal{F}$ y $\mathcal{G}$ respectivamente. Además, Λ^* es el operador adjunto asociado a Λ.

La aplicación del Teorema 10 al problema (1.3) se presenta a través del siguiente razonamiento. Podemos expresar (1.3) como en (1.10) donde $\mathcal{F}$ corresponde a la parte regular de (1.3) y $\mathcal{G}$ a la norma ℓ_1 . El operador Λ es el operador identidad y los espacios $V = Y = \mathbb{R}^n$. Para poder describir el problema dual es necesario calcular las funciones convexas conjugadas dadas en la Definición 6.4 del apéndice.

Lema 3. *Sea $\mathcal{F}(x) = f(x), \forall x \in \mathbb{R}^n$, entonces*

$$f^*(-q) = \begin{cases} & -f(x) + (\nabla f(x), x), \\ \textit{tal que:} & \\ & \nabla f(x) = -q. \end{cases}$$

Demostración. Utilizando la definición de función conjugada y puesto que Λ^* es el

operador identidad entonces calculamos:

$$\begin{aligned} f^*(-q) &= \sup_{x\in\mathbb{R}^n} \{(-q,x) - f(x)\}, \\ &= \sup_{x\in\mathbb{R}^n} \{-(q,x) - f(x)\}, \\ &= \sup_{x\in\mathbb{R}^n} \{-(q,x) - f(x)\}. \end{aligned}$$

Notando $L(x) = -(x,q) - f(x), \forall x \in \mathbb{R}^n$. Es fácil notar que L es cóncava y diferenciable. Por tanto, el máximo global para esta función se alcanza cuando $(\nabla L(x), v) = 0$, para todo $v \in \mathbb{R}^n$. Derivando L con respecto a x tenemos:

$$(\nabla L(x), v) = -(q,v) - (\nabla f(x), v) = 0. \tag{1.12}$$

Entonces el supremo se alcanza para $x(q)$ que satisface (1.12). De las condiciones anteriores podemos concluir que el valor de $f^*(-q)$ es

$$f^*(-q) = \begin{cases} \text{tal que:} & -f(x) + (\nabla f(x), x) \\ & \\ & \nabla f(x) = -q. \end{cases}$$

□

Ahora estudiamos la función convexa conjugada para $\mathcal{G}$, para eso es necesario definir el conjunto:

$$K_\beta = \{q : (q,p) \leq \beta \parallel q \parallel_1, \text{ para todo } p \in \mathbb{R}^n\}.$$

Entonces se tiene el siguiente resultado:

Lema 4. *Sea $\mathcal{G} = \beta \parallel \cdot \parallel_1$ su conjugada convexa está determinada por*

$$\mathcal{G}^*(q^*) = \begin{cases} 0 & \text{si } p \in K_\beta, \\ +\infty & \text{caso contrario.} \end{cases} \tag{1.13}$$

Demostración. De la definición de función convexa conjugada tenemos que

$$\mathcal{G}^*(q) = \sup_{v\in\mathbb{R}^n} \{(q,v) - \beta \parallel v \parallel_1\}.$$

Analizamos los siguientes casos:

CASO 1 $q^* \in K_\beta$. Es fácil notar en este caso que:

$$\mathcal{G}^*(q^*) = 0,$$

pues la función es menor o igual que cero para todo $q \in \mathbb{R}^n$.

CASO 2 $q \notin K_\beta$. Puesto que en este caso la función es no negativa y no acotada superiormente, se tiene que

$$\mathcal{G}^*(q^*) = \infty.$$

de donde se sigue (1.13) □

De lo anterior, se tiene que el problema dual para (1.3) queda descrito por la siguiente expresión:

$$\begin{cases} \sup\limits_{q \in K_\beta} & -f(x) + (\nabla f(x), x) \\ \text{sujeto a:} & \\ & \nabla f(x) = -q. \end{cases} \tag{1.14}$$

A continuación verificamos las hipótesis del Teorema 10 para mostrar la existencia de un multiplicador, y que no existe brecha de dualidad, es decir:

$$\mathcal{F}(\bar{x}) + \mathcal{G}(\Lambda\bar{x}) = -\mathcal{F}^*(\Lambda^*\bar{q}) - \mathcal{G}^*(\bar{q}).$$

Teorema 1. *Sean* (1.3) *y* (1.14) *los problemas primal y dual respectivamente. Entonces, ambos problemas tienen al menos una solución* $\bar{x}$ *y* $\bar{q}$*, respectivamente. Además, para estas soluciones no existe brecha en la dualidad.*

Demostración. Utilizando el Teorema 10, es suficiente verificar las hipótesis de dicho teorema. Por tanto, tomando $V = Y = \mathbb{R}^n$ se tiene que

- $\mathcal{F}$ y $\mathcal{G}$ son propias por definición, y puesto que $\mathcal{G}$ es la norma ℓ_1 entonces es continua.
- $f(x) + \beta \parallel x \parallel_1$ es coerciva pues si $\parallel x \parallel \to \infty$, entonces $\parallel x \parallel_1 \to \infty$ y además, ya que $f(x)$ es acotada inferiormente, entonces $f(x) + \parallel x \parallel_1 \to \infty$.
- $\mathbb{R}^n$ es reflexivo, pues es un espacio del Hilbert.

  Gracias al Teorema 10 podemos concluir que (1.3) y (1.14) tienen soluciones $\bar{x}$ y

$\bar{q}$ respectivamente. Además,

$$\min_{x\in\mathbb{R}^n}\left(f(x)+\beta\parallel x\parallel_1\right)=\begin{cases}\sup\limits_{q\in K_\beta} & f(x(q))-(\nabla f(x(q)),x(q))\\ \text{sujeto a:} & \\ & \nabla f(x(q))+q=0.\end{cases}$$

□

Condiciones de complementariedad

Una vez que se ha probado que no existe brecha en la dualidad se pueden caracterizar la condiciones de complementariedad del problema a través del Teorema 11 del apéndice.

PROPOSICIÓN 1.1. *Sean $\bar{x}$ y $\bar{q}$ soluciones de los problemas* (1.3) *y* (1.14) *respectivamente, entonces éstas satisfacen el siguiente sistema:*

$$\begin{cases}\nabla f(x)-\bar{q}=0,\\ \beta(\parallel\bar{x}\parallel_1-\parallel v\parallel_1)\leq(\bar{q},(\bar{x}-v)) \quad \forall v\in\mathbb{R}^n,\\ \beta\parallel\bar{x}\parallel_1=(\bar{x},\bar{q}).\end{cases} \tag{1.15}$$

Demostración. De la Proposición 1 sabemos que no existe brecha en la dualidad, entonces, el Teorema 11 nos permite concluir que las soluciones $\bar{x}$ y $\bar{q}$ de los problemas primal y dual respectivamente satisfacen:

$$\begin{aligned}-\bar{q}&\in\partial\mathcal{F}(\bar{x}),\\ \bar{q}&\in\partial\mathcal{G}(\bar{x}).\end{aligned}$$

De la diferenciabilidad de $\mathcal{F}$, la primera expresión puede ser escrita de la siguiente manera

$$\nabla f(\bar{x})=-\bar{q}\quad\Leftrightarrow\quad\nabla f(\bar{x})+\bar{q}=0,$$

Para la segunda expresión utilizamos la definición de subdiferencial, por tanto tenemos:

$$\beta\left(\parallel v\parallel_1-\parallel\bar{x}\parallel_1\right)\geq\left(\bar{q},(v-\bar{x})\right),\quad\forall v\in\mathbb{R}^n.$$

Por otro lado, puesto que $\bar{q}\in K_\beta$, tomando $v=0$ se sigue que:

$$\beta\parallel\bar{x}\parallel_1=(\bar{x},\bar{q}).$$

Resumiendo, el sistema de optimalidad está dado por

$$\begin{cases} (\nabla f(\bar{x}) - \bar{q}) = 0, \\ \beta(\| \bar{x} \|_1 - \| v \|_1) \leq \bar{q}(\bar{x} - v) \quad \forall v \in \mathbb{R}^n, \\ \beta \| \bar{x} \|_1 = \bar{x}\bar{q}. \end{cases}$$ □

Los siguientes Lemas nos permiten expresar el sistema (1.15) de forma más simple.

Lema 5.

$$\bar{q} \in K_\beta \Leftrightarrow |\bar{q}_i| \leq \beta \quad \forall i = 1, \ldots, n. \tag{1.16}$$

Demostración. En primer lugar vamos a demostrar que $\bar{q} \in K_\beta \Rightarrow |\bar{q}_i| \leq \beta$. Procedemos por contradicción, es decir, suponemos que existe $\hat{i} \in \{1, \ldots, n\}$ tal que $|\bar{q}_{\hat{i}}| > \beta$.

Si tomamos

$$v^* = \begin{cases} \bar{q}_i & \text{si } i = \hat{i}, \\ 0 & \text{caso contrario,} \end{cases}$$

entonces se cumple que:

$$\begin{aligned} (v^*, \bar{q}) - \beta \| v^* \|_1 &= |\bar{q}_{\hat{i}}|^2 - \beta|\bar{q}_{\hat{i}}|, \\ &= (|\bar{q}_{\hat{i}}| - \beta)|\bar{q}_{\hat{i}}| > 0, \end{aligned}$$

de donde se sigue que $q \notin K_\beta$.
Recíprocamente, suponemos que $|\bar{q}_i| \leq \beta$ para todo $i = 1, \ldots, n$. Entonces, de la desigualdad de Hölder se tiene que

$$\begin{aligned} (v, \bar{q}) - \beta \| v \|_1 &\leq \| v \|_1 \| \bar{q} \|_\infty - \beta \| v \|_1 \\ &\leq (\| \bar{q} \|_\infty - \beta) \| v \|_1 . \end{aligned}$$

Puesto que $|q_i| \leq \beta$, para todo $i = 1, \ldots, n$, tenemos:

$$\| \bar{q} \|_\infty \leq \beta \quad \Leftrightarrow \quad |\bar{q}_i| \leq \beta \qquad \forall i = 1, \ldots, n,$$

se tiene que $\| \bar{q} \|_\infty - \beta \leq 0$, en consecuencia

$$(v, \bar{q}) - \beta \| v \|_1 \leq 0,$$

lo cual implica que $\bar{q} \in K_\beta$. □

Lema 6. *La condición*

$$\beta \| \bar{x} \|_1 = (\bar{x}, \bar{q}), \tag{1.17}$$

puede ser expresada como:

$$\begin{cases} \bar{x}_i = 0 \quad \acute{o}, \\ \bar{x}_i \neq 0 \quad y\ \bar{q}_i = \beta\ sign(\bar{x}_i). \end{cases} \tag{1.18}$$

para todo $i \in \{1, \ldots, n\}$.

Demostración. En primer lugar debemos recordar que de las restricciones del problema dual se tiene que $\bar{q} \in K_\beta$. Entonces se tiene que

$$\begin{aligned} 0 &= \beta \parallel \bar{x} \parallel_1 - (\bar{x}, \bar{q}) && (1.19)\\ &= \beta \sum_{i=1}^{n} |\bar{x}_i| - \sum_{i=1}^{n} \bar{x}_i \bar{q}_i && (1.20)\\ &\geq \beta \sum_{i=1}^{n} |\bar{x}_i| - \sum_{i=1}^{n} |\bar{x}_i||\bar{q}_i| && (1.21)\\ &= \sum_{i=1}^{n} (\beta - |\bar{q}_i|)\bar{x}_i \geq 0. \end{aligned}$$

Puesto que todos los términos de la última expresión son positivos se tiene que:

$$(\beta - |\bar{q}_i|)\,|\bar{x}_i| = 0 \qquad \forall i \in \{1, \ldots, n\}. \tag{1.22}$$

De la expresión anterior podemos deducir los siguientes casos:

I) $\bar{x}_i = 0$ y $|\bar{q}_i| = \beta$ para todo $i = 1, \ldots, n$.

II) $\bar{x}_i = 0$ y $|\bar{q}_i| \neq 0$ para todo $i = 1, \ldots, n$.

III) $\bar{x}_i \neq 0$ y $\bar{q}_i = \beta$ para todo $i = 1, \ldots, n$.

Los dos primeros casos implican directamente (1.18), puesto que no se necesita información adicional sobre $\bar{q}$. El análisis se refiere únicamente al caso iii). De (1.22) tenemos:

$$\beta|\bar{x}_i| = |\bar{q}_i||\bar{x}_i|,$$

de donde se sigue que:

$$|\bar{q}_i| = \beta \qquad \Leftrightarrow \bar{q}_i = \beta\ \text{sign}(q_i),$$

pero teniendo en cuenta que $\text{sign}(\bar{q}_i) = \text{sign}(\bar{x}_i)$, pues la condición (1.17) es verdadera. De esta manera se obtiene el resultado deseado.

Recíprocamente,

$$\begin{aligned}\beta \parallel \bar{x} \parallel_1 - (\bar{x}, \bar{q}) =& \beta \sum_{i=1}^{n} |\bar{x}_i| - \sum_{i=1}^{n} \bar{q}_i \, \bar{x}_i \\ &= \beta \sum_{i:\, x_i=0}^{n} |\bar{x}_i| - \sum_{i:\, x_i=0}^{n} \bar{q}_i \, \bar{x}_i + \beta \sum_{i:\, x_i\neq 0}^{n} |\bar{x}_i| - \sum_{i:\, x_i\neq 0}^{n} \bar{q}_i \, \bar{x}_i \\ &= \beta \sum_{i:\, x_i\neq 0}^{n} |\bar{x}_i| - \sum_{i:\, x_i\neq 0}^{n} \bar{q}_i \, \bar{x}_i \\ &= \beta \sum_{i:\, x_i\neq 0}^{n} |\bar{x}_i| - \beta \sum_{i:\, x_i\neq 0}^{n} \operatorname{sign}(x_i) \, \bar{x}_i = 0.\end{aligned}$$

La última igualdad se obtiene de la hipótesis (1.18). □

Gracias a los resultados anteriores podemos concluir que las condiciones de optimalidad para el problema (1.3) en términos de su variable dual están dadas por:

$$\begin{cases} (\nabla f(\bar{x}) - \bar{q}) = 0, & \\ |\bar{q}_i| \leq \beta & \forall i = 1, \ldots, n. \\ \begin{cases} \bar{x}_i = 0 & \text{ó,} \\ \bar{x}_i \neq 0 & \text{y } \bar{q}_i = \beta \operatorname{sign}(\bar{x}_i). \end{cases} & \end{cases} \tag{1.23}$$

Capítulo 2

Algoritmos de descenso basados en direcciones ortantes con información de segundo orden de la parte regular

En este capítulo discutimos algunos algoritmos propuestos previamente para la resolución de problemas dispersos de la forma (1.3). Las principales estrategias utilizadas por estos algoritmos son:

- Dirección de descenso
- Direcciones ortantes
- Búsqueda lineal proyectada

1. Elementos del análisis de los problemas dispersos

Los elementos que serán descritos en las próximas secciones hacen referencia a ideas que provienen del análisis de los problemas del tipo (1.3), como por ejemplo la dirección tipo descenso más profundo, las direcciones ortantes y las proyecciones sobre el ortante. Estas definiciones serán presentadas a continuación para el correcto entendimiento de los algoritmos.

Teniendo en cuenta que la función objetivo no es diferenciable, no se puede hablar de la dirección del descenso más profundo en términos de sus derivadas, debido a que usualmente una dirección de descenso es caracterizada por medio del gradiente de la función. Sin embargo teniendo en cuenta las herramientas del análisis convexo, se puede definir un análogo a la dirección del descenso más profundo en términos de los

subgradientes. Denominamos a esta dirección la dirección *tipo descenso más profundo*, la cuál está definida por el valor negativo de la siguiente expresión:

$$\widetilde{\nabla}_i \varphi(x) = \begin{cases} \nabla_i f(x) + \beta \operatorname{sign}(x_i) & \text{si } x_i \neq 0, \\ \nabla_i f(x) + \beta & \text{si } x_i = 0 \text{ y } \nabla_i f(x) < -\beta, \\ \nabla_i f(x) - \beta & \text{si } x_i = 0 \text{ y } \nabla_i f(x) > \beta, \\ 0 & \text{caso contrario.} \end{cases} \tag{2.1}$$

La definición anterior se puede expresar de la siguiente manera:

$$\widetilde{\nabla} \varphi(x) = \widetilde{\nabla} f(x) + z,$$

donde $\widetilde{\nabla} f(x)$ está dado por la siguiente expresión:

$$\widetilde{\nabla}_i f(x) = \begin{cases} 0 & \text{si } x_i = 0 \text{ y } |\nabla_i f(x)| \leq \beta, \\ \nabla_i f(x) & \text{caso contrario,} \end{cases}$$

y z es la dirección ortante definida por:

$$z_i = \begin{cases} \operatorname{sign}(x_i) & \text{si } x_i \neq 0, \\ 1 & \text{si } x_i = 0 \text{ y } \nabla_i f(x) < -\beta, \\ -1 & \text{si } x_i = 0 \text{ y } \nabla_i f(x) > \beta, \\ 0 & \text{caso contrario.} \end{cases} \tag{2.2}$$

A continuación se presenta un ejemplo en el cuál se puede evidenciar el funcionamiento de esta dirección.

EJEMPLO 3 (Dirección tipo descenso más profundo)**.** Se tiene el siguiente problema de optimización unidimensional

$$\min_{x \in \mathbb{R}} \ \varphi(x) = (x-2)^2 + \beta \, |x|,$$

con $\beta = 1$. Analizamos las posibles direcciones de descenso, basados en el conocimiento previo de la solución. En la Figura 1 se muestra el gráfico de la función $\varphi(x)$. Claramente se puede observar que la solución se encuentra cercano al valor $x = 1{,}5$. Además, en el gráfico se puede observar que esta función es diferenciable en casi todos los puntos, con excepción del punto marcado en donde además se muestran algunas de las posibles rectas tangentes a este punto. Y es justamente este punto el que no nos permite utilizar algoritmos usuales de optimización.

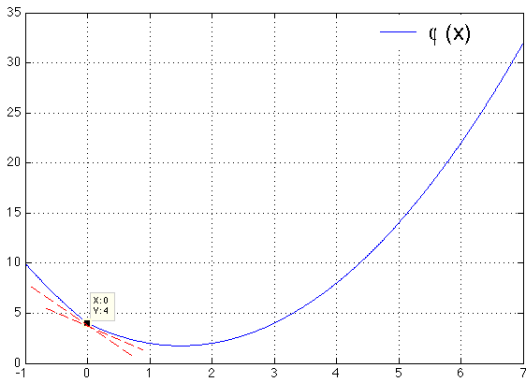


Figura 1. Gráfico de $\varphi(x)$

En la Figura 2 se muestran los gráficos tanto de la parte regular (azul), como el de la norma ℓ_1 (rojo), y la función objetivo $\varphi(x)$ (negro).

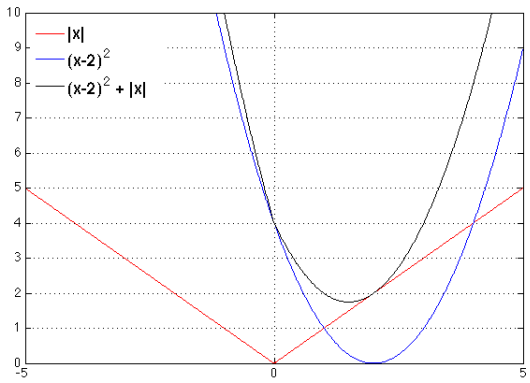


Figura 2. Gráfico de $\varphi(x)$, $f(x)$ y la norma ℓ_1 por separado

Analizamos este ejemplo por casos:

CASO I: $x \neq 0$: En este caso suponemos $x > 0$, el caso $x < 0$ será equivalente. En la Figura 3 se muestra un punto tal que $x > 0$. Es claro que en este caso el valor de la pendientes de la parte no diferenciable es positivo, y la pendiente de la parte regular también, entonces la dirección del descenso más profundo estaría dada por el signo $-(\nabla f(x) + \beta) < 0$, lo que indica que el paso es hacia la izquierda. Además en la figura 3 muestra el sentido de la dirección de descenso.

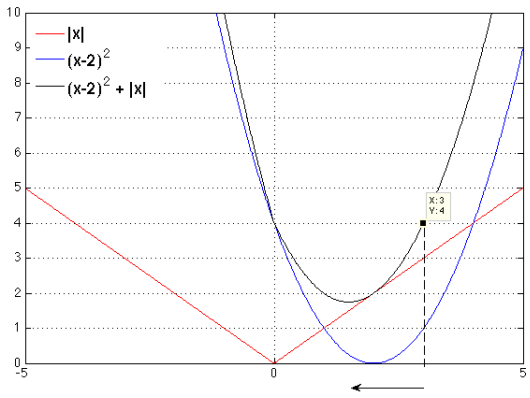


Figura 3. Caso I: dirección de descenso negativa

CASO II: $x = 0$: En este caso al no tener información sobre la pendiente de la norma ℓ_1, se propone utilizar la información de la pendiente de la parte regular $(\nabla f(x))$ para garantizar el decrecimiento; es decir, si la pendiente de la parte regular es significativamente pronunciada. En este caso la pendiente es negativa y claramente se puede ver que $\nabla f(x) < -\beta$; es decir, $\nabla f(x) + \beta < 0$ entonces nos movemos en dirección contraria (positiva). La dirección encontrada para este caso nos garantiza el decrecimiento en la función objetivo y además nos acerca al valor mínimo de la función. Este mismo análisis se puede realizar en el caso análogo $(\nabla f(x) > \beta)$.

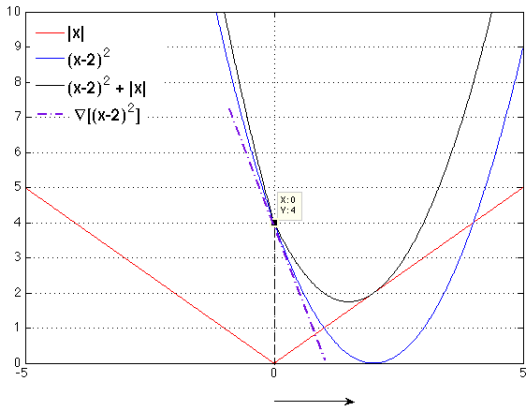


Figura 4. Caso II: dirección de descenso positiva pues $\nabla f(x) + \beta < 0$

CASO III: $x = 0$ **y** $-\beta < \nabla f(x) < \beta$: Por un lado tenemos que $\nabla f(x) + \beta > 0$, es decir la dirección es negativa, pero al analizar el otro caso, es decir cuando $\nabla f(x) - \beta < 0$, tenemos que la dirección es positiva. Este caso se fija el valor de la dirección a 0. Se puede entender este comportamiento como la falta de información para garantizar el decrecimiento en la función objetivo; es decir, la derivada de la

parte regular no es suficientemente pronunciada para garantizar decrecimiento en la función objetivo. En la Figura 5 se muestra la forma que tendría la parte regular en este caso. Cabe recalcar que este gráfico no corresponde al ejemplo en cuestión pero permitirá una ilustración del caso en estudio.

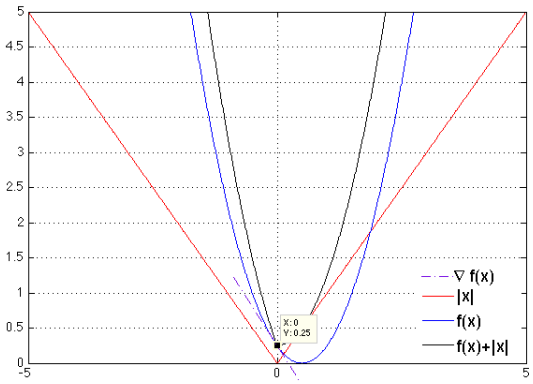


Figura 5. Caso III: dirección de descenso nula pues $|\nabla f(x)| \leq \beta$

Este ejemplo ilustra la definición de la dirección tipo descenso más profundo. Es claro que si z mide la contribución de la norma en la dirección de descenso, este valor debe ser elemento del subdiferencial. La próxima proposición prueba esta afirmación.

PROPOSICIÓN 2.1. *La dirección ortante z definida en* (2.2) *es elemento del subdiferencial de* $\| x_0 \|_1$.

Demostración. Del análisis convexo sabemos que para que z sea elemento del subgradiente de la norma ℓ_1 se debe satisfacer que

$$\| x \|_1 - \| x_0 \|_1 \geq (z, x - x_0),$$

para todo $x \in \mathbb{R}^n$. Sea $x \in \mathbb{R}^n$ analizamos la cantidad $z^T(x - x_0)$. Puesto que z es acotada, ya que únicamente puede tomar los valores $\{-1, 0, 1\}$ y utilizando la desigualdad de Hölder tenemos:

$$z^T(x - x_0) = z^T x - z^T x_0 \leq \| z \|_\infty \| x \|_1 - z^T x_0. \tag{2.3}$$

Analizando el término $z^T x_0$ como sigue:

$$z^T x_0 = \sum_{i=1}^{n} z_i (x_0)_i.$$

De la definición de las direcciones ortantes sabemos que si $(x_0)_i \neq 0$ entonces $z_i = \text{sign}((x_0)_i)$, de este modo $(x_0)_i z_i = |(x_0)_i|$, caso contrario, si $(x_0)_i = 0$ se tiene que el producto $(x_0)_i z_i = 0$ y consecuentemente

$$z^T x_0 = \sum_{i=1}^{n} (x_0)_i z_i = \sum_{i:\ (x_0)_i \neq 0} |(x_0)_i| = \| x_0 \|_1 . \tag{2.4}$$

Reemplazando (2.4) en (2.3) tenemos que:

$$z^T(x - x_0) = z^T x - z^T x_0 \leq \| z \|_\infty \| x \|_1 - \| x_0 \|_1 .$$

Notando que $\| z \|_\infty = 1$, se obtiene el resultado deseado. □

Las direcciones ortantes nos permiten caracterizar los ortantes de la siguiente manera:

$$\Omega_k := \{d : \text{sign}(d_i) = \text{sign}(z_i^k), \forall i = 1, \ldots, n\}, \qquad \text{para } k \in \mathbb{N}.$$

y la proyección sobre el ortante:

$$\mathcal{P}(x)_i = \begin{cases} x_i & \text{si sign}(x_i) = \text{sign}(z_i), \\ 0 & \text{caso contrario.} \end{cases} \tag{2.5}$$

El esquema general de los algoritmos de descenso utiliza dos elementos: las direcciones de descenso y el paso de descenso. Los elementos presentados anteriormente permiten establecer una dirección de descenso. Sin embargo, es necesario definir una regla de decrecimiento suficiente para el problema (1.3), utilizando la información proporcionada por la dirección tipo descenso más profundo descrita en (2.1).
Se propone la siguiente regla de decrecimiento suficiente, análoga a la Regla de *Armijo*:

$$\varphi[\mathcal{P}(x^k + s_k d^k)] \leq \varphi(x^k) + \widetilde{\nabla}\varphi(x^k)^T [\mathcal{P}(x^k + s_k d^k) - x^k], \tag{2.6}$$

se escoge el valor del s_k más grande que satisface la condición anterior. Para encontrar el valor de s se utiliza el procedimiento de *Backtraking* [28] descrito en el siguiente algoritmo:

Algorithm 1 Búsqueda Lineal Proyectada

1: Inicializar s_0, c_1.
2: Calcular $\widetilde{\nabla}\varphi(x^k)$, $\varphi(x^k)$, $\mathcal{P}[x^k + s_0 d^k]$
3: **while** $\varphi\left(\mathcal{P}[x^k + s_i d^k]\right) > \varphi(x^k) + c_1 \widetilde{\nabla}\varphi(x^k)^T(\mathcal{P}[x^k + s_i d^k] - x^k)$ **do**
4: $\quad s_i \leftarrow \dfrac{s_i}{2}$.
5: $\quad$ Calcular $\mathcal{P}[x^k + s_i d^k]$
6: $\quad i \leftarrow i + 1$.
7: **end while**

Dado que utilizamos información de segundo orden para la construcción de la dirección de descenso, utilizamos el algoritmo BFGS que describimos a continuación. El algoritmo BFGS *c.f.* [28] es el algoritmo tipo Quasi–Newton más popular; este recibe el nombre de sus creadores, Broyden, Fletcher, Goldfarb y Shanno. De la teoría de optimización sabemos que para resolver un problema de la forma

$$\min_{x \in \mathbb{R}^n} g(x)$$

una dirección de descenso debe satisfacer:

$$p^k \cdot \nabla g_k < 0,$$

ya que esta apunta en sentido contrario al gradiente, y que puede ser escrita como

$$p^k = A^{-1} \nabla g_k,$$

donde A es una matriz invertible y definida positiva, que representa a la matriz Hessiana en el caso de los algoritmos de Newton o que aproxima la Hessiana de f para el caso de los algoritmos Quasi-Newton. La idea del BFGS es aproximar el valor de la matriz A, de tal manera que conserve ciertas características de la segunda derivada de f, por ejemplo la simetría. La aproximación de la matriz Hessiana se realiza de acuerdo a la fórmula recurrente

$$B_{k+1} = B_k - \frac{B_k r_k r_k^T B_k}{r_k^T B_k r_k} + \frac{y_k y_k^T}{y_k^T r_k}. \tag{2.7}$$

ver c.f. [28]

Realizamos un estudio más a fondo sobre las técnicas aplicadas en los algoritmos que utilizan información del subgradiente. Entre los principales algoritmos de este tipo son NW–CG [3] y OWL [13], los cuáles utilizan información de segundo orden

correspondiente a la parte regular de la función objetivo.

2. Algoritmo NW–CG

El algoritmo NW–CG, propuesto por Nocedal et. al. en [3], es un algoritmo de dos fases: una de predicción de los conjuntos activos y una fase de minimización.

Fase de Predicción de Conjuntos Activos En esta fase se estiman las variables de la solución que se anulan a través de la identificación del *ortante* en el que se ejecuta el proceso de optimización. Para dicha identificación es necesario el uso de las llamadas *direcciones ortantes*. Estas direcciones son las que caracterizan el ortante donde es posible disminuir el valor de la función objetivo y las componentes de la solución que se anulan, y por tanto los conjuntos activos. Nótese que la restricción de la función objetivo en el ortante identificado es diferenciable.

Fase de Minimización Una vez identificadas las variables de la solución aproximada que se anulan se estima los valores de las variables libres a través del algoritmo de Gradiente Conjugado c.f. [28], considerando para su ejecución la información de segundo orden asociada a la parte regular.

El paso principal del algoritmo NW–CG puede ser descrito mediante la siguiente expresión

$$x^{k+1} \leftarrow \mathcal{P}\left[x^k - s_k\left[Y^k\left((Y^k)^T B_k(Y^k)\right)^{-1}(Y^k)^T\widetilde{\nabla}\varphi(x^k)\right]\right]$$

donde Y_k es una matriz que genera el espacio de las variables libres. $\mathcal{P}$ representa la proyección sobre el ortante, la cuál garantiza que la iteración $k+1$–ésima permanezca en el ortante identificado en la fase anterior. Gracias a las direcciones ortantes se pueden definir conjuntos activos de la siguiente manera:

$$\mathcal{A}_k := \{i\colon z_i^k = 0\},$$

cuya principal función es la identificación de las componentes de la solución aproximada que serán cero. Las variables libres corresponden justamente a las componentes de la dirección de descenso que no estén en los conjuntos activos. La idea principal de este algoritmo es fijar el valor de las componentes de la dirección de descenso que pertenecen a los conjuntos activos en cero, y la matriz Y^k cumple ese rol. La dirección de descenso para este algoritmo satisface el sistema

$$\left[Y_k^T\nabla^2 f(x^k)Y_k\right](d^k)^Y = -Y_k^T\widetilde{\nabla}\varphi(x^k). \tag{2.8}$$

El método se resume en el siguiente algoritmo:

Algorithm 2 NW–CG

1: Inicializar x^0 y calcular $\nabla^2 f(x^0)$.
2: **repeat**
3: Determinar $\mathcal{A}_k$ caracterizado por z^k, la dirección ortante.
4: Calcular la dirección $\widetilde{\nabla}\varphi(x^k)$. de acuerdo a (2.1)
5: Resolver el sistema (2.8) con el algoritmo del Gradiente Conjugado.
6: Calcular $d^k = Y_k(d^k)^Y$.
7: Calcular s_k mediante un procedimiento de búsqueda lineal proyectada.
8: Fijar $x^{k+1} = \mathcal{P}\left(x^k + s_k d^k\right)$.
9: **until** el criterio de parada sea satisfecho

Al fijar los valores de la dirección a cero en esas componentes el algoritmo garantiza que en la siguiente iteración la solución continua siendo cero en dicha componente. Sin embargo, este algoritmo no puede garantizar la monotonía de los conjuntos activos en todas las iteraciones, sino solo en una vecindad de la solución.

3. Algoritmo OWL

Este algoritmo fue desarrollado por Andrew y Gao en [13]. De manera similar se ejecutan dos fases: la primera para escoger el ortante en el que se va a trabajar, y una segunda fase de minimización sobre el ortante identificado. Se define el *pseudo–gradiente*, el cuál coincide con la dirección tipo descenso más profundo del algoritmo NW–CG. El cálculo de la dirección de descenso está basado en el algoritmo L–BFGS, principalmente por la gran cantidad de datos que se manejan en los problemas para los cuales fue diseñado este algoritmo. Se puede utilizar esquemas tipo BFGS sin comprometer la eficiencia del algoritmo, si el número de variables es reducido. Vale la pena recalcar que el esquema BFGS aproxima la Hessiana de la parte regular de la función objetivo. La principal diferencia entre este algoritmo y el NW–CG radica en que para utilizar la dirección de descenso se realiza una proyección adicional en la dirección contraria al *pseudo–gradiente*, lo cuál nos garantiza que los signos de ambos coinciden. Este hecho se justifica por la necesidad de que el proceso de búsqueda lineal no se desvíe demasiado de la dirección del descenso más profundo, y esta característica es necesaria para garantizar la convergencia del algoritmo [13]. La iteración de este algoritmo se puede

expresar de la siguiente manera.

$$x^{k+1} \leftarrow \mathcal{P}\left(x^k + s_k \mathcal{P}_{[-\tilde{\nabla}\varphi]}(d^k)\right).$$

donde $\mathcal{P}_{[-\tilde{\nabla}\varphi]}$ es la proyección en la dirección contraria a la del *pseudo–gradiente*.
El algoritmo queda descrito a través de los siguientes pasos:

Algorithm 3 OWL

1: Inicializar x^0, H_0, $\widetilde{\nabla}\varphi(0)$.
2: **for** $k = 0$ **to** MaxIter **do**
3: Calcular $v^k = -\widetilde{\nabla}\varphi(x^k)$.
4: Resolver el sistema $d^k = H_k v^k$.
5: Proyectar d^k en $-\left(\widetilde{\nabla}\varphi(x^k)\right)$.
6: Calcular s_k mediante un procedimiento de búsqueda lineal proyectado.
7: Fijar $x^{k+1} = \mathcal{P}\left(x^k + s_k d^k\right)$.
8: **end for**

Donde H_k corresponde a la matriz generada por el algoritmo **L–BFGS** o BFGS, la cual aproxima el valor de la inversa de la matriz Hessiana.

Capítulo 3

Algoritmos con información de segundo orden de la parte no diferenciable

En este capítulo proponemos un nuevo algoritmo de descenso para la resolución de problemas dispersos. Es decir, nuestro algoritmo se basa en un mejoramiento de la función objetivo a lo largo de una dirección y un paso calculado de manera que se produzca una disminución en el valor de la función que estamos minimizando. En las siguientes secciones describimos como calcular estas cantidades. Las ideas principales que utilizamos para el diseño de nuestros algoritmos se resumen en tres características principales:

- Utilización de direcciones ortantes para el reconocimiento de los conjuntos activos.
- Incorporación de información de segundo orden asociada a la norma ℓ_1 a partir de su regularización de Huber.
- Búsqueda lineal proyectada.

Nos referiremos a este algoritmo como **OESOM** por sus siglas en inglés (Orthant–wise Enriched Second Order Method), cuyo equivalente en español es un algoritmo de segundo orden con información enriquecida, o completa, que trabaja de ortante en ortante. La información enriquecida proviene de la información de segundo orden asociada a la norma ℓ_1, la cuál es obtenida de una regularización tipo Huber.

El algoritmo opera en un esquema de los métodos tipo Quasi–Newton. Es decir, métodos que incluyen información de segundo orden aproximando la segunda derivada de la función objetivo. Como se observó antes, la información de segundo orden

asociada a la parte regular se la realiza mediante el algoritmo BFGS u otras técnicas conocidas. La novedad de los métodos desarrollados en este trabajo es la inclusión de la información de segundo orden asociada a la norma ℓ_1; que por supuesto no está dada en términos de la segunda derivada, sino que se realiza una regularización de la norma ℓ_1 que nos permita obtener la información de segundo orden en términos del subdiferencial de la primera derivada de la regularización de la norma ℓ_1.

La primera característica importante de nuestro algoritmo corresponde al uso de direcciones ortantes las cuáles como se mencionó en el capítulo previo nos permite identificar rápidamente los conjuntos activos (las entradas donde se anula la solución). Restringiendo el proceso de optimización a los ortantes donde la función es suave. Además, garantizamos el descenso de la función objetivo en cada iteración del algoritmo, a través del uso de la dirección tipo descenso más profundo. Las direcciones ortantes junto con la dirección tipo descenso más profundo están dadas por las siguientes expresiones:

Dirección tipo descenso más profundo

$$\widetilde{\nabla}_i \varphi(x) = \begin{cases} \nabla_i f(x) + \beta \operatorname{sign}(x_i) & \text{si } x_i \neq 0, \\ \nabla_i f(x) + \beta & \text{si } x_i = 0 \text{ y } \nabla_i f(x) < -\beta, \\ \nabla_i f(x) - \beta & \text{si } x_i = 0 \text{ y } \nabla_i f(x) > \beta, \\ 0 & \text{caso contrario.} \end{cases} \tag{3.1}$$

está dirección nos garantiza el decrecimiento de la función objetivo en cada iteración, ya que utiliza la información del gradiente cuando este existe ($x \neq 0$). Cuando la función no es diferenciable, toma en cuenta el valor pendiente de la parte regular para escoger la dirección que nos garantiza el decrecimiento. Si el valor de la pendiente no es significativo, se fija el valor de la dirección a cero, para de esta forma garantizar el valor de la función objetivo no será mayor en la siguiente iteración.

Dirección ortante

$$z_i = \begin{cases} \operatorname{sign}(x_i) & \text{si } x_i \neq 0, \\ 1 & \text{si } x_i = 0 \text{ y } \nabla_i f(x) < -\beta, \\ -1 & \text{si } x_i = 0 \text{ y } \nabla_i f(x) > \beta, \\ 0 & \text{caso contrario.} \end{cases} \tag{3.2}$$

En los capítulos previos se presentaron ejemplos los cuales permitieron entender el funcionamiento de estos elementos.

La información de segundo orden que se incluye en el cálculo de las direcciones de descenso fue calculada en los capítulos anteriores y está dada por la siguiente expre-

sión:

$$(\Gamma)_{ii} = \begin{cases} \gamma & \text{si } \gamma|x_i| \leq 1, \\ 0 & \text{caso contrario.} \end{cases} \tag{3.3}$$

Puesto que $\gamma > 0$, esta matriz es semi–definida positiva. Esta característica es importante puesto que junto con la matriz correspondiente a la información de segundo orden de la parte regular, la cuál será definida positiva, nos garantiza la existencia de las direcciones de descenso.

El tercer elemento importante en nuestro algoritmo es el procedimiento de búsqueda lineal proyectada, el cuál nos permite encontrar un paso de descenso que nos garantice permanecer en el mismo ortante en la siguiente iteración. La condición de decrecimiento suficiente en este caso está dada por:

$$\varphi[\mathcal{P}(x^k + s_k d^k)] \leq \varphi(x^k) + \widetilde{\nabla}\varphi(x^k)^T[\mathcal{P}(x^k + s_k d^k) - x^k], \tag{3.4}$$

donde la proyección sobre el ortante corresponde a la siguiente expresión:

$$\mathcal{P}(x)_i = \begin{cases} x_i & \text{si } \operatorname{sign}(x_i) = \operatorname{sign}(z_i), \\ 0 & \text{caso contrario.} \end{cases} \tag{3.5}$$

Y el algoritmo queda descrito a través de los siguientes pasos:

Algorithm 4 Búsqueda Lineal Proyectada

1: Inicializar s_0, c_1.
2: Calcular $\widetilde{\nabla}\varphi(x^k)$, $\varphi(x^k)$, $\mathcal{P}[x^k + s_0 d^k]$
3: **while** $\varphi\left(\mathcal{P}[x^k + s_i d^k]\right) > \varphi(x^k) + c_1\widetilde{\nabla}\varphi(x^k)^T(\mathcal{P}[x^k + s_i d^k] - x^k)$ **do**
4: $\quad s_i \leftarrow \dfrac{s_i}{2}$.
5: $\quad$ Calcular $\mathcal{P}[x^k + s_i d^k]$
6: $\quad i \leftarrow i + 1$.
7: **end while**

Previo a la presentación del algoritmo **OESOM** mostramos un esquema más natural de resolución de problemas dispersos el cuál se basa principalmente en incluir la regularización de tipo Huber al funcional de costo. Así se obtiene una función de costo diferenciable y es posible aplicar algoritmos clásicos de optimización diferenciable.

1. Algoritmo basado en la regularización del problema disperso

En las secciones anteriores se formuló un problema regularizado, el cuál se obtuvo mediante el reemplazo de la norma ℓ_1 por su regularización tipo Huber. Este es un problema diferenciable, por lo tanto se puede aplicar un algoritmo de optimización tradicional, como el BFGS. Utilizando el gradiente de la función objetivo descrito en la siguiente expresión.

$$\nabla \varphi_\gamma(x) = \nabla f(x) + \beta \nabla \mathcal{G}(x), \tag{3.6}$$

donde $\nabla \mathcal{G}(x)$ corresponde al gradiente de la regularización de Huber, y está dado por la siguiente expresión:

$$\nabla \mathcal{G}(x) = \left(\frac{\gamma x_i}{\text{máx}\{1, \gamma |x_i|\}} \right)_{i=1,\dots,n}.$$

En lugar de aplicar directamente el algoritmo BFGS al problema regularizado aplicamos este esquema solo a la parte regular, mientras que la información de segundo orden correspondiente a la parte de la regularización se aproxima usando la ecuación (3.3). Resolviendo el siguiente sistema:

$$(B_k + \Gamma_k) d^k = -\nabla \varphi_\gamma(x^k). \tag{3.7}$$

con $\nabla \varphi_\gamma(x)$ dado en (3.6).

Dado que el problema es diferenciable, aplicamos métodos de búsqueda lineal clásicos de la optimización diferenciable. Se utiliza el procedimiento de Backtraking para el cálculo del paso de descenso. El algoritmo queda descrito de la siguiente manera:

Algorithm 5 BFGS–Regularizado

1: Inicializar x^0, B^1, γ.
2: **repeat**
3: Calcular Γ_k dado en (3.3).
4: Calcular la dirección de descenso d^k resolviendo el sistema lineal (3.7).
5: Calcular el paso de descenso s_k con un procedimiento de Backtraking.
6: Calcular $x^{k+1} = x^k + s_k\, d^k$.
7: Actualizar B_k dada por (2.7).
8: $k \leftarrow k + 1$.
9: **until** El criterio de parada sea satisfecho

2. Algoritmo OESOM

En los capítulos anteriores se presentaron los algoritmos OWL y NW–CG que utilizan direcciones ortantes para el cálculo de las direcciones de descenso, y para la predicción de los subespacios en donde se realizará el proceso de optimización. La pregunta natural que surge es; ¿ existe la posibilidad de mejorar el desempeño de estos algoritmos utilizando la información proporcionada por las direcciones ortantes?. Y de ser así, ¿ qué información es necesaria para obtener los resultados deseados?. La respuesta a la primera pregunta es afirmativa. Es decir, es posible mejorar el desempeño de estos algoritmo. Y estos resultados se los obtiene utilizando la información de segundo orden asociada a la norma ℓ_1, a través de una regularización local, que depende de un parámetro de aproximación $\gamma > 0$, como se realizó en el algoritmo basado en la regularización del problema.
En sentido estricto no existen información de primer orden para la norma ℓ_1, sin embargo, podemos obtener dicha información a través del uso de los subgradientes. Los elementos del subdiferencial, además de proporcionar información de primer orden, nos dan un indicio de la existencia de información de la curvatura de la norma ℓ_1. Esta característica es precisamente la novedad del algoritmo **OESOM** respecto de los algoritmos del mismo tipo. En capítulos anteriores se ha indicado que forma tendrá la información de segundo orden, sin embargo no hemos especificado su implementación. Por tanto debemos indicar el papel que cumple esta matriz y qué la diferencia de la información utilizada en otro algoritmos. En los siguientes párrafos explicamos que su rol es crucial para el correcto desempeño de los algoritmos.Los algoritmos como el NW–CG y OWL necesitan realizar un cierto tipo de proyecciones (fijar ciertas componentes de la dirección a cero) para poder obtener un buen comportamiento numérico. Por ejemplo, como se mencionó antes el algoritmo NW–CG, después de haber identificado el ortante activo, fija las componentes de la dirección a cero y resuelve un problema de minimización sobre un subespacio. El ortante activo queda definido a través de las direcciones ortantes (3.2) por la siguiente expresión:

$$\Omega_k := \{d \colon \operatorname{sign}(d_i) = \operatorname{sign}(z_i^k), \forall i = 1, \ldots, n\}, \qquad \text{para } k \in \mathbb{N}.$$

Por otro lado, el algoritmo OWL, realiza una proyección con respecto a la dirección tipo descenso más profundo (3.1), fijando a cero las componentes de la dirección que tengan signo distinto de ésta. En el caso del algoritmo **OESOM**, la matriz asociada a la información de segundo orden de la norma ℓ_1, tiene un efecto de escalamiento sobre aquellas componentes d_i de la dirección que satisfacen $|x_i| < 1/\gamma$, es decir aquellas que están suficientemente cercanas a cero. Se puede entender a este comportamiento como una relajación de las condiciones utilizadas por los algoritmos NW–CG y OWL. Ade-

más, hay que tener en cuenta que la cantidad de componentes escaladas es mayor que en el caso de los otros algoritmos, lo cuál se traduce en unas direcciones de descenso menores en norma, y favorecen a las propiedades de monotonía de los conjuntos activos.

Inspirados en los algoritmos tipo Quasi-Newton proponemos el siguiente sistema para el cálculo de las direcciones de descenso:

$$(B_k + \Gamma_k)d^k = -\widetilde{\nabla}\varphi(x^k) \tag{3.8}$$

donde B_k es la matriz resultante del algoritmo BFGS que aproxima a la Hessiana de la parte regular, Γ_k es la información de segundo orden que aporta la norma ℓ_1 de la ecuación (3.3) y $\widetilde{\nabla}\varphi(x)$ es la dirección tipo descenso más profundo (3.1), la cuál nos garantiza decrecimiento de la función objetivo en cada iteración. Es importante notar que la dirección generada es una dirección de descenso, ver [8].
Una vez presentados todos los elementos del algoritmo se procede a la presentación formal del mismo.

Algorithm 6 Algoritmo OESOM

1: Inicializar x^0, B^1.
2: **repeat**
3: Calcular Γ_k dado en (3.3) .
4: Calcular la dirección ortante correspondiente z^k dada en la ecuación (3.2).
5: Calcular la dirección de descenso d^k usando (3.8).
6: Calcular s_k con el Algoritmo 4.
7: Calcular $x^{k+1} = \mathcal{P}\left[x^k + s_k d^k\right]$ usando (3.5)
8: Actualizar la matriz BFGS.
9: $k \leftarrow k + 1$.
10: **until** El criterio de parada sea satisfecho

3. Propiedades del algoritmo OESOM

Una vez presentado el algoritmo es necesario analizar cuales son las propiedades que permiten que el algoritmo **OESOM** tenga un mejor desempeño en comparación con otros algoritmos como 2 y 3. Las principales propiedades teóricas correspondientes a las direcciones de descenso generadas por el algoritmo y monotonía de los conjuntos activos son presentadas a continuación.

Los conjuntos activos están definidos por:

$$\mathcal{A}_k := \{i \colon z_i^k = 0\}. \tag{3.9}$$

Es claro notar que con esta definición de los conjuntos activos, se garantiza que las componentes i–ésimas con $i \in \mathcal{A}_k$ de la solución en la iteración k-ésima del algoritmo también son cero, es decir, se tiene que si $i \in \mathcal{A}_k$ y $z_i^k = 0$ entonces $x_i^k = 0$.

Además, teniendo en cuenta la definición de la proyección sobre el ortante dada en la ecuación (2.5), se definen los siguientes conjuntos:

$$\mathbb{H}^k := \{i \colon \text{sign}(x_i^k + s\, d_i^k) = \text{sign}(z_i^k)\}, \tag{3.10}$$

los cuales representan las componentes de la solución que permanecen en el ortante antes de la proyección y además nos permiten expresar la iteración $(k+1)$–ésima del Algoritmo 6 de la siguiente manera:

$$x^{k+1} = \mathcal{P}[x^k + sd^k] = x^k + \tilde{d}^k \tag{3.11}$$

con:

$$\tilde{d}_i^k = \begin{cases} sd_i^k, & \text{si } i \in \mathbb{H}_k, \\ -x_i^k, & \text{caso contrario.} \end{cases} \tag{3.12}$$

Otra propiedad importante de la proyección sobre el ortante se presenta en la siguiente proposición:

PROPOSICIÓN 3.1. *En todas las iteraciones del algoritmo **OESOM** se satisface que*

$$|x_i^k| \geq -\tilde{d}_i^k z_i^k \qquad \forall i \in \{1, \ldots, n\}.$$

Demostración. De (2.5) se tiene que $x_i^{k+1}\, z_i^k \geq 0$. De (3.11), tenemos que $(x_i^k + s\tilde{d}_i^k)\, z_i^k \geq 0$ para todo $i = 1, \ldots, n$, de lo cual obtenemos

$$x_i^k z_i^k \geq -\tilde{d}_i^k z_i^k.$$

Por otro lado, de la definición de las direcciones ortantes podemos concluir que $x_i^k z_i^k = |x_i^k|$ y en consecuencia obtenemos el resultado deseado. □

Una versión más fuerte de la proposición 3.1 dice

PROPOSICIÓN 3.2. *Para cada iteración del algoritmo **OESOM** se tiene que*

$$\| x \|_1 \geq -(\tilde{d}^k, z^k).$$

Demostración. De la proposición 3.1, sabemos que para cada componente i de los vectores, se tiene que $|x_i| + \tilde{d}_i^k \, z_i^k \geq 0$, por lo tanto

$$\sum_{i=1}^{n} \left(|x_i| + \tilde{d}_i^k \, z_i^k \right) = \| \, x \, \|_1 + \left(d^k, z^k \right) \geq 0.$$

Así, podemos concluir que

$$\| \, x \, \|_1 \geq -(d^k, z^k). \qquad \square$$

En la presentación del algoritmo se habló acerca del efecto de escalamiento que tiene la matriz Γ_k sobre algunas componentes de la dirección. La forma de la cota para estos valores, se presenta a continuación.

De las hipótesis impuestas sobre la parte regular del problema, $f(x)$ en (1.3), se tiene que tanto la matriz que aproxima su segunda derivada B^k, como el valor de la primera derivada $\nabla f(x)$ son acotadas. Por lo tanto, usando la ecuación (3.8) tenemos que existe una constante $C > 0$ tal que

$$|d_j^k| = \left| \frac{1}{B_{jj}^k + \gamma} \left(\sum_{\ell \neq j} B_{j\ell}^k d_\ell^k + \widetilde{\nabla} \varphi(x^k) \right) \right| \leq \frac{C}{\gamma} \tag{3.13}$$

para todas las componentes j tal que $|x_j^k| \leq 1/\gamma$.

Los siguientes resultados fueron desarrollados en [8] donde se pueden encontrar las demostraciones correspondientes a continuación se presentarán esbozos de las demostraciones más importantes.

3.1. Dirección de descenso

Lema 7. *Sea $x = x^k$ la k–ésima iteración del algoritmo **OESOM** y $z = z^k$ la dirección ortante asociada. Sea además γ suficientemente grande y s suficientemente pequeño tal que*

$$sign(x_i + s d_i) = sign(x_i) \qquad \text{para todo } i\colon |x_i| \geq \frac{1}{\gamma}. \tag{3.14}$$

Entonces,

$$\left(\nabla f(x)^T + \beta z \right) \tilde{d} < 0,$$

donde $z \in \partial \| \cdot \|_1$ definida en (2.2).

Lema 8. *Para la dirección $\tilde{d}$ tenemos que*

$$z^T \tilde{d} = \| \, x + \tilde{d} \, \|_1 - \| \, x \, \|_1 \, .$$

Estas propiedades implican el siguiente resultado:

Teorema 2 (Dirección de descenso)**.** *La dirección $\tilde{d}$ definida por* (3.12) *es una dirección de descenso para φ en x, es decir*

$$\varphi(x + s\tilde{d}) < \varphi(x)$$

para un valor de s suficientemente pequeño.

La demostración de este teorema se encuentra de manera detallada en [8], sin embargo a continuación se presenta un esquema general de la misma. La idea principal de la demostración es aplicar los resultados obtenidos en los Lemas 7 y 8, y utilizando una expansión de Taylor de f alrededor de x. Se obtiene el siguiente resultado:

$$\varphi(x + \tilde{d}) < \varphi(x) + o\left(\| \tilde{d} \|_2\right).$$

Teniendo en cuenta la definición de $\tilde{d}$ dada en (3.12) y que ésta es acotada, podemos decir que $o\left(\| \tilde{d} \|_2\right) = o(s)$. De donde se puede concluir que $\varphi(x + \tilde{d}) < \varphi(x)$, y así se obtiene el resultado deseado.

3.2. Propiedades de monotonía de conjuntos activos

El siguiente Teorema muestra que las iteraciones generadas por el algoritmo **OESOM** garantizan la monotonía de los conjuntos activos en una vecindad de $\bar{x}$ la solución de (1.3). De las propiedades de los algoritmos 2 y 3 también se puede garantizar la monotonía de los conjuntos activos, sin embargo el Teorema muestra que la vecindad para nuestro algoritmo es más grande que en los otros casos, y que esta vecindad depende del parámetro γ.

Teorema 3 (Monotonía de conjuntos activos)**.** *Asumamos que la sucesión $\{x^k\}_{k\in\mathbb{N}}$ generada por el algoritmo **OESOM** converge a la solución $\bar{x}$. Si se cumple la condición de complementariedad estricta en x^k, es decir, el conjunto $\{i\colon \nabla_i f(x^k) = \beta \text{ y } x_i^k = 0\}$ es vacío, entonces*

$$\mathcal{A}_k \subseteq \mathcal{A}_{k+1},$$

para k suficientemente grande.

Demostración. Sea $i \in \mathcal{A}_k$, es decir $z_i^k = 0$, entonces debemos demostrar que $z_i^{k+1} = 0$. De la definición de direcciones ortantes basta mostrar:

i) $x_i^{k+1} = 0$,

ii) $|\nabla_i f(x^{k+1})| \leq \beta$.

La condición $i)$ se sigue automáticamente de la proyección sobre el ortante. Para la demostración de la condición $ii)$ vamos a realizar una expansión de Taylor de $\nabla f(x^{k+1})$ al rededor de x^{k+1}, teniendo en cuenta que de (3.11) $x^{k+1} = x^k + \tilde{d}^k$. Entonces:

$$\nabla f(x^{k+1}) = \nabla f(x^k) + \nabla^2 f\left(x^k + (1-\theta)\tilde{d}^k\right)^T \tilde{d}^k$$

para $\theta \in (0,1)$. De la hipótesis de complementariedad estricta y puesto que $z_i^k = 0$ tenemos que:

$$|\nabla_i f(x^{k+1})| < \beta + \left|\sum_i \left[\nabla^2 f(x^k + (1-\theta)\tilde{d}^k)\right]_{ij} \tilde{d}_j^k\right| \tag{3.15}$$

Analizando el último término de (3.15), y notando por $H^k = \left[\nabla^2 f(x^k + (1-\theta)\tilde{d}^k)\right]$ únicamente por simplicidad en la notación, obtenemos la estimación:

$$\left|\sum_j H_{ij}\tilde{d}_j\right| \leq \| H \|_2 \sum_{j\neq i} |\tilde{d}_j|,$$

pues $\tilde{d}_i = 0$ ya que $i \in \mathcal{A}_k$. Analizando los casos en que $|x_j| \leq 1/\gamma$ y su complemento se sigue que

$$\left|\sum_j H_{ij}\tilde{d}_j\right| \leq \| H \|_2 \left[\sum_{\substack{j\neq i \\ j:\ |x_j|<1/\gamma}} |\tilde{d}_j| + \sum_{\substack{j\neq i \\ j:\ |x_j|\geq 1/\gamma}} |\tilde{d}_j|\right]. \tag{3.16}$$

De (3.13) sabemos que el primer término de (3.16) está acotado por C/γ con $C > 0$, para $j \in \mathbb{H}^k$, donde $\mathbb{H}^k$ es el conjunto definido en (3.10). En el caso en que $j \notin \mathbb{H}^k$ se tiene que $\tilde{d}_j = x_j < 1/\gamma$. Así, podemos concluir que existe $C > 0$ tal que:

$$\left|\sum_j H_{ij}\tilde{d}_j\right| \leq \| H \|_2 \left[\frac{C}{\gamma} + \sum_{\substack{j\neq i \\ j:\ |x_j|\geq 1/\gamma}} |\tilde{d}_j|\right]. \tag{3.17}$$

Por otro lado, puesto que $x^k \to \bar{x}$ cuando $k \to \infty$, de donde se sigue que $\tilde{d}^k = x^{k+1} - x^k \to 0$ cuando $k \to \infty$. En consecuencia, para cualquier $\varepsilon > 0$ existe k_0, tal que para todo $k \geq k_0$, podemos concluir que

$$|\nabla_i f(x^{k+1})| \leq \beta + \varepsilon \| H^k \|_2 .$$

Puesto que ε es arbitrario y tomando γ suficientemente grande, la condición $ii)$ se

satisface y junto con $i)$ obtenemos el resultado deseado. □

Otra de las propiedades importantes del algoritmo **OESOM** es la equivalencia con el algoritmo Newton Semi–smooth, está propiedad se refiere a que bajo ciertos valores del parámetro de regularización γ, las iteraciones del algoritmo **OESOM** son equivalentes a las del método antes mencionado. El análisis completo de esta estrategia puede ser estudiado en [8]. Una de las principales consecuencias de esta equivalencia es que el algoritmo **OESOM** hereda las propiedades de convergencia del algoritmo Newton Semi–smooth; es decir, se tiene una convergencia local y más rápida que la de sus competidores. Además, se pudo establecer una estrategia adaptiva para escoger el parámetro de regularización. La propiedad que se debe satisfacer es la siguiente:

$$\gamma = \max_{i:\, x_i^k \neq 0} \left(\frac{|\nabla_i f(x^k) + \beta \text{sign}(x_i^k)|}{|x_i^k|} \right). \tag{3.18}$$

Este hecho nos permite establecer una versión adaptiva del algoritmo **OESOM**, la cuál viene dada por los siguientes pasos:

Algorithm 7 Algoritmo OESOM con regularización automática

1: Inicializar x^0, B^1, $k = 1$.
2: **repeat**
3: Calcular γ_k usando (3.18).
4: Calcular Γ_k dado en (1.8) .
5: Calcular la dirección ortante correspondiente z^k dada en la ecuación (2.2).
6: Calcular la dirección de descenso d^k usando (3.8).
7: Calcular s_k con el Algoritmo 1.
8: Calcular $x^{k+1} = x^k + s_k\, d^k$ y proyectar sobre el ortante según la ecuación 2.5
9: Actualizar la matriz BFGS.
10: $k \leftarrow k + 1$.
11: **until** El criterio de parada sea satisfecho

Comparamos el desempeño de esta estrategia con el algoritmo **OESOM** sin regularización adaptiva en los test numéricos.

3.3. Soluciones nulas

Estudiamos el caso de las soluciones nulas ya que corresponde al caso extremo de dispersión y exhibe propiedades interesantes. Empezamos con la discusión sobre las

condiciones suficientes sobre el parámetro de dispersión β en el problema (1.3) para que la solución sea exactamente nula.

Teorema 4. *Si* $\beta \geq \beta_0 = \| \nabla f(0) \|_\infty$, *entonces la solución del problema* (1.3) *es* $\bar{x} \equiv 0$.

Demostración. Basta probar que

$$\varphi(x) - \varphi(0) \geq 0, \qquad \text{para todo } x \in \mathbb{R}^n.$$

Se tiene que:

$$\begin{aligned} \varphi(x) - \varphi(0) &= f(x) - f(0) + \beta \| x \|_1 \\ &= \nabla f(\xi)^T x + \beta \| x \|_1 . \end{aligned}$$

para $\xi \in (0, x)$. La última igualdad se la obtiene utilizando el Teorema del valor medio. Usando la desigualdad de Hölder podemos concluir que

$$\begin{aligned} \varphi(x) - \varphi(0) &\geq - \| \nabla f(\xi) \|_\infty \| x \|_1 + \beta \| x \|_1, \\ &\geq (\beta - \| \nabla f(\xi) \|_\infty) \| x \|_1, \end{aligned}$$

y puesto que f es continuamente diferenciable, se tiene que:

$$\beta \geq \beta_0 = \| \nabla f(0) \|_\infty \quad \text{implica que} \quad \beta \geq \| \nabla f(\xi) \|_\infty, \tag{3.19}$$

con $\xi \in (0, x)$. Así:

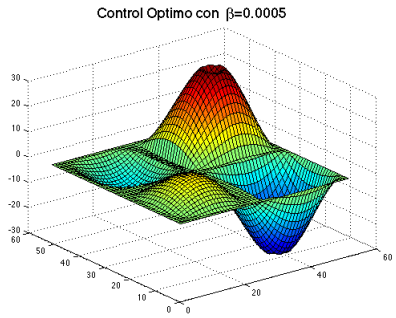
$$\varphi(x) - \varphi(0) \geq 0,$$

con lo cual se demuestra el enunciado del Teorema. □

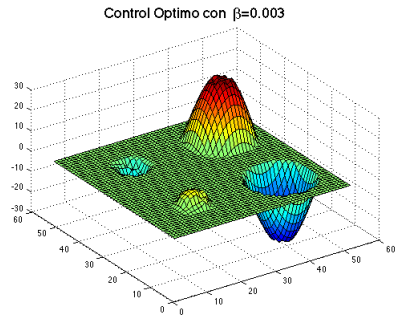
Para ilustrar el efecto del parámetro de dispersión β resolvemos el Experimento 1 presentado en [34]; sin considerar las restricciones sobre el control, es decir, fijamos $a = -\infty$ y $b = \infty$. Resolvemos el siguiente problema:

$$(P) \begin{cases} \min\limits_{(y,u)} \dfrac{1}{2} \| y - y_d \|^2_{L^2(D)} + \dfrac{\alpha}{2} \| u \|^2_{L^2(D)} + \beta \| u \|_{L^1(D)}, \\ \text{s.a.} \\ \qquad -\nu \Delta y = u + f, \qquad \text{en } D, \\ \qquad y = 0, \qquad \text{sobre } \partial D, \end{cases}$$

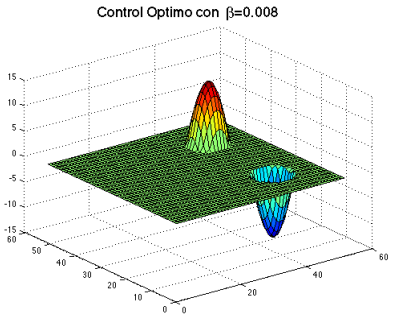
con las funciones $y_d = \sin(2\pi x) \sin(2\pi y) \exp(2x)/6$ y $f = 0$. Fijamos además el parámetro $\alpha = 1e-5$.



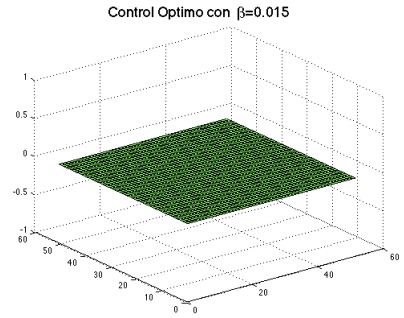
(a) Solución con $\beta = 0{,}0005$



(b) Solución con $\beta = 0{,}003$



(c) Solución con $\beta = 0{,}008$



(d) Solución con $\beta = 0{,}015$

Figura 1. Control óptimo obtenido con el algoritmo **OESOM** para distintos valores de β.

En la Figura 1 se aprecia la estructura dispersa de las soluciones para diversos valores de β sabiendo que $\beta_0 = 0{,}01$.

Denominamos a β_0 como el umbral de dispersión; pues, para valores mayores a este la solución es nula.

Capítulo 4

Experimentos numéricos y aplicaciones

En este capítulo, estudiamos numéricamente el desempeño de nuestros algoritmos realizando test numéricos de comparación con los algoritmos OWL [13], NW–CG [3]. Para esto, consideramos varios problemas de control óptimo gobernados por ecuaciones diferenciales parciales, problemas cuadráticos lineales en dimensión finita generados aleatoriamente, entre otros. Para el caso de problemas de control óptimo, como método de discretización de las ecuaciones diferenciales parciales se utiliza el método de diferencias finitas en el caso elíptico; mientras que en el caso parabólico, se utilizan diferencias finitas para aproximar la variable espacial y un esquema de Euler implícito para resolver el sistema de EDO resultante. Los problemas generados aleatoriamente utilizan funciones de MATLAB, que nos permiten obtener las características deseadas para cada uno de los parámetros.

En todos estos ejemplos se evidencia que el Algoritmo **OESOM**, desarrollado en este trabajo, es capaz de acelerar la convergencia con respecto a los algoritmos NW–CG y OWL. En este trabajo hemos realizado un ligero cambio a los algoritmos con respecto a los métodos utilizados para la resolución del sistema del cuál se obtiene d^k. Para el caso del algoritmo OWL utilizamos la aproximación de la segunda derivada proporcionada por el algoritmo BFGS y en el caso del algoritmo NW–CG utilizamos un método directo para la resolución del sistema y la matriz correspondiente a la Hessiana de la parte regular se la aproxima con el método BFGS. El capítulo está divido en dos partes en las que se presentan diferentes tipos de experimentos numéricos. En la primera parte, se presentan experimentos numéricos con el objetivo de comparar el desempeño de los algoritmos que hemos propuesto en este trabajo, con los algoritmos NW–CG y OWL que consideramos nuestros principales competidores. En la segunda parte, nos concentramos en mostrar la aplicabilidad y alcance de nuestro algoritmo en la resolución numérica de varios problemas prácticos.

En cada experimento especificamos sus objetivos, así como también los paráme-

tros y las funciones utilizadas. La Tabla 1 muestra un resumen de los experimentos numéricos presentados en este capítulo.

Experimento	Descripción
1	Problemas cuadráticos en dimensión finita generados aleatoriamente.
2	Problema de control óptimo elíptico lineal bi–dimensional.
3	Problema de control óptimo elíptico no lineal.
4	Problema de control óptimo elíptico lineal con restricciones sobre el control.
5	Aplicación a un problema de control óptimo de dinámica poblacional espacial.

Tabla 1. Resumen de experimentos numéricos

Para los problemas de control óptimo tomamos un funcional de costo cuadrático tipo "tracking" para la parte regular del funcional. Notamos al dominio espacial como D, el cual será especificado en cada experimento.

1. Resolución numérica de problemas lineales cuadráticos generados aleatoriamente

En [5] se propone resolver 1000 experimentos de un problema cuadrático, que tiene la siguiente forma:

$$(\mathcal{P}) \begin{cases} \min\limits_{u \in \mathbb{R}^n} & \frac{1}{2} u^T Q u + q^T u + \beta \parallel u \parallel_1, \end{cases}$$

donde Q es generada aleatoriamente con la función `sprandsym` de MATLAB, la cuál genera una matriz con un cierto grado de dispersión (d). Es decir, la cantidad de entradas distintas de cero en la matriz de tamaño n será dn^2 y su número de condición es igual al reciproco de rc. Esta matriz es generada a través de un método aleatorio de rotación de Jacobi para una matriz diagonal definida positiva. Para la construcción de Q se utilizan los siguiente parámetros: $d = 25\,\%$ con un número de condición igual a 10^4 y 10^7. Además q es un vector cuyas componentes tienen magnitud parecida a los valores propios de Q y tiene aproximadamente la mitad de componentes negativas y la otra positiva. De los experimentos realizados se evidenció que el valor de $\parallel q \parallel_\infty$ es aproximadamente 1,8. Por lo tanto, el parámetro β es generado aleatoriamente en el intervalo $[1{,}8\ , n/3]$. Para garantizar que las soluciones sean nulas, se verifica si se satisface la condición del Teorema 4. El objetivo principal del experimento es verificar la

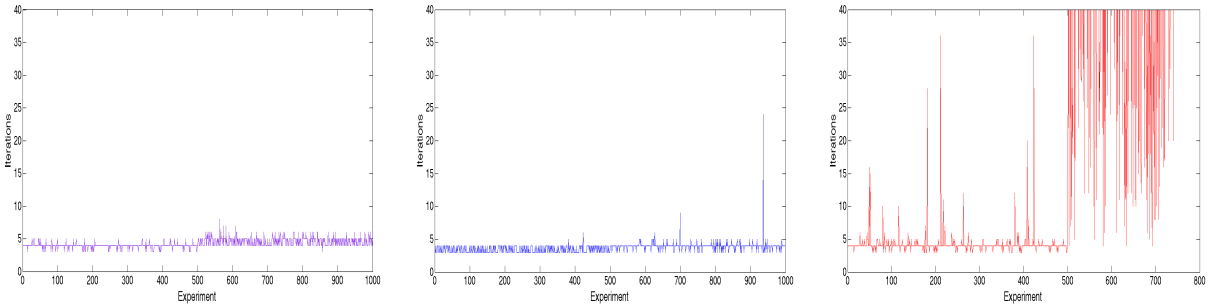


Figura 1. Número de iteraciones: OESOM (izquierda), OWL (centro), NW–CG (derecha) para 1000 experimentos aleatorios.

cantidad de veces que cada algoritmo falla en resolver el problema. Se considera que un algoritmo falla en la resolución de un problema cuando no ha alcanzado la solución nula hasta la iteración 6000. Los resultados obtenidos se muestran en la Tabla 2.

Número de condición de Q	Algoritmos		
	OESOM	NW–CG	OWL
	Número de Fallas		
Moderado	0	0	0
Alto	0	260	2
Total	0	260	2

Tabla 2. Resumen experimento 1

La Figura 1 muestra la cantidad de iteraciones requeridas por cada algoritmo para resolver cada uno de los problemas. Se puede observar que el número de iteraciones realizadas por el algoritmo **OESOM** es significativamente más pequeño que en el caso del algoritmo NW–CG, lo cuál indica un mejor desempeño de nuestro algoritmo para este tipo específico de problemas. Cabe recalcar que las fallas del Algoritmo NW–CG se deben principalmente a que 6000 iteraciones no son suficientes para la convergencia del mismo, sin embargo el algoritmo podría converger a la solución nula en un número de iteraciones mayor.

En la Tabla 3, se muestra la media y varianza del número de iteraciones para los experimentos terminados exitosamente por cada uno de los algoritmos. Podemos observar en los resultados que a pesar de que la media de iteraciones para el algoritmo OWL es menor que para el algoritmo **OESOM**, la varianza es mayor, lo que se traduce en el que comportamiento del algoritmo OWL es más oscilatorio; es decir, existen experimentos para los cuales el algoritmo OWL necesita muchas más iteraciones. Para el algoritmo OESOM la varianza es menor, y se interpreta como un comportamiento más homogéneo.

Algoritmo	Media	Varianza
OESOM	4.2970	0.4252
NW–CG	69.3149	9.64 e+04
OWL	3.7154	0.7394

Tabla 3. Resumen estadístico

2. Resolución de un problema de control óptimo elíptico lineal

El siguiente problema es una modificación del problema 3 de [34]

$$(P)\begin{cases} \min\limits_{(y,u)} \dfrac{1}{2} \parallel y - y_d \parallel^2_{L^2(D)} + \dfrac{\alpha}{2} \parallel u \parallel^2_{L^2(D)} + \beta \parallel u \parallel_{L^1(D)}, \\ \text{s.a.} \\ \qquad -\nu\Delta y = u + f, \qquad \text{en } D, \\ \qquad\quad y = 0, \qquad\quad \text{sobre } \partial D. \end{cases}$$

con $D = (0,1) \times (0,1)$, y los siguientes parámetros: $\nu = 0{,}1$, $\gamma = 10e4$, $\alpha = 0{,}0002$, $\beta = 0{,}0094$, un paso de discretización $h = \frac{1}{51}$. Además tomamos las funciones $y_d = \sin(4\pi x)\cos(8\pi y)\exp(2\pi x)$ y $f = 0$. Usamos como criterio de parada la siguiente condición:

$$\parallel x^{k+1} - x^k \parallel + \frac{|\varphi(x^k) - \varphi(x^{k-1})|}{\varphi(x^k)} < \eta,$$

donde $\eta = 10e - 5$. Se utilizó este criterio de parada puesto que nos permite medir la variación de la solución con respecto a la iteración anterior, y además la variación del valor de la función objetivo. Es decir, el algoritmo para cuando la variación de la solución y del valor de la función objetivo no sean significativos. Para la resolución numérica de este problema primero discretizamos usando el método de diferencias finitas y lo transformamos en un problema de optimización en un espacio de dimensión finita. Para la discretización del problema se utiliza las siguientes herramientas.

La norma L^2 está descrita en términos de integrales, utilizando la regla del punto medio para la aproximación de las integrales con $D \subset \mathbb{R}^2$, obtenemos que

$$\int_D y^2(x)dx \approx h^2 \sum_i v_i (y_i)^2.$$

donde v_i corresponden a los coeficientes resultantes de la regla del punto medio, y h es

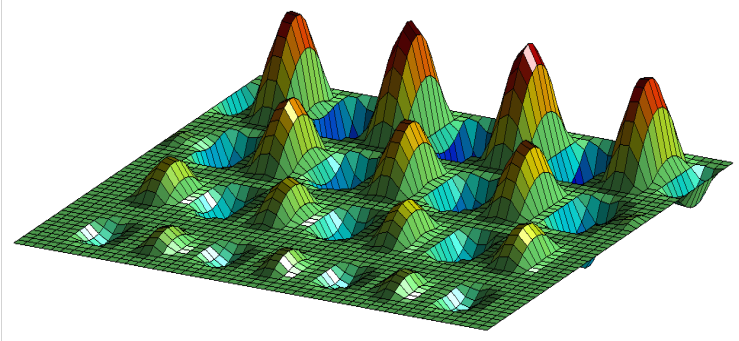


Figura 2. Control óptimo disperso u.

el paso de discretización. Considerando los errores de redondeo, se optó por realizar la multiplicación por el paso de discretización al final de la aproximación. Definimos además, el operador control estado S en dimensión finita por:

$$\begin{aligned} S : \mathbb{R}^n &\to \mathbb{R}^n \\ u &\mapsto Su = A^{-1}(u + f), \end{aligned}$$

donde A es la matriz del Laplaciano discreto, utilizando diferencias finitas. Con estos elementos podemos escribir la función objetivo discretizada de la siguiente manera:

$$\varphi(u) = \frac{h^2}{2} \sum_{i=1}^{n} v_i((Su)_i - y_{di})^2 + \frac{\alpha\, h^2}{2} \sum_{i=1}^{n} v_i(u_i)^2 + \beta\, h^2 \sum_{i=1}^{n} v_i|u_i|.$$

Puesto que todos los coeficientes de la aproximación de la integral v_i son positivos, podemos definir la variable $w_i = v_i\, u_i$. De la linealidad del operador control estado, y notando que $u_i = w_i/v_i$, realizamos un cambio de variable en la función objetivo, obteniendo la siguiente expresión:

$$\varphi_h(w) = \frac{h^2}{2} \sum_{i=1}^{n} \frac{1}{v_i}((Sw)_i - v_i\, y_{di})^2 + \frac{\alpha\, h^2}{2} \sum_{i=1}^{n} \frac{1}{v_i}(w_i)^2 + \beta\, h^2 \sum_{i=1}^{n} |w_i|.$$

Por lo tanto, las dirección ortantes y la dirección tipo descenso más profundo son calculadas en función de la nueva variable w. Este esquema de discretización será utilizado para todos los experimentos de control óptimo.

En la figura 2 se muestra el control óptimo calculado por todos los algoritmos. El principal objetivo de esta figura es mostrar la estructura que tienen los controles y la acción localizada de los mismos en subdominios compactos.

En la Tabla 4, podemos comparar el desempeño de los algoritmos mediante el número de iteraciones requeridas para la convergencia.

Algoritmo	**Número de iteraciones**	**Función objetivo**
OESOM	22	1.23648
NW–CG	48	1.23648
OWL	420	1.23649

Tabla 4. Resumen del experimento 2

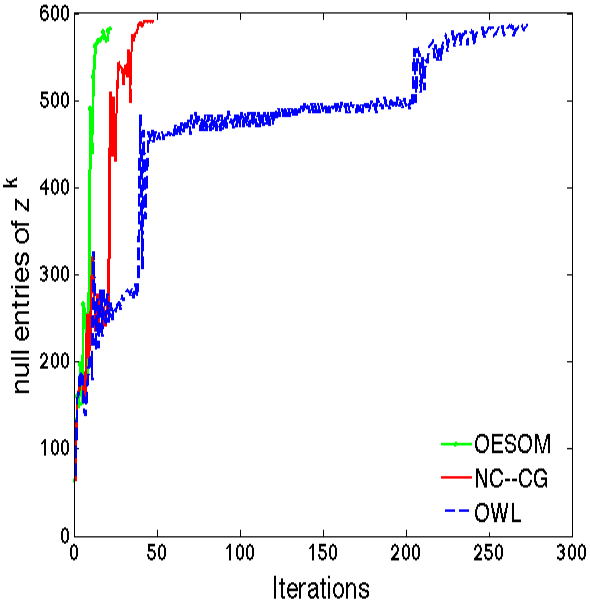
En la Figura 3 se presenta el decrecimiento de la función objetivo en cada iteración y la cardinalidad de los conjuntos activos asociados a z^k y x^k, y el valor de la norma euclidea de la dirección.

Los resultados del mismo experimento pero utilizando como criterio de parada un número máximo de iteraciones se muestran en la Tabla 5. Este criterio de parada nos permite verificar el desempeño de los algoritmos de una manera más acertada, debido a que es en las primeras iteraciones donde ocurren los decrecimientos más significativos en la función objetivo y también la identificación de las conjuntos activos. Se utilizan como número máximo de iteraciones $MaxIter = 40$.

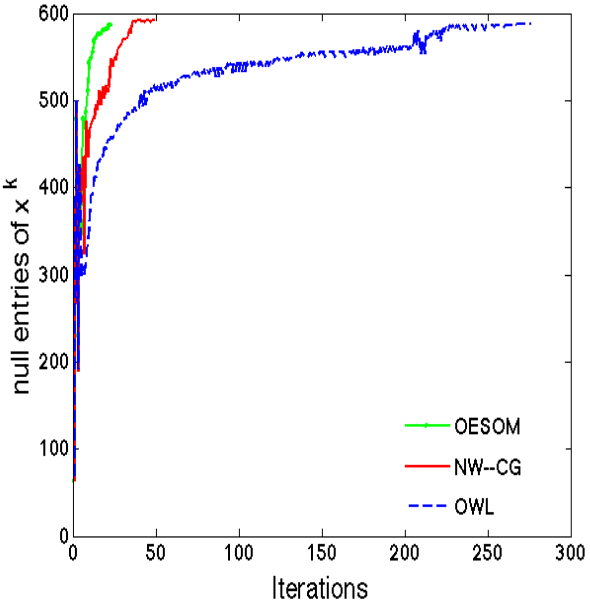
Algoritmo	**Función objetivo**
OESOM	1.2364
OWL	1.2558
NW–CG	1.2365

Tabla 5. Resumen del experimento 2, con $MaxIter = 40$

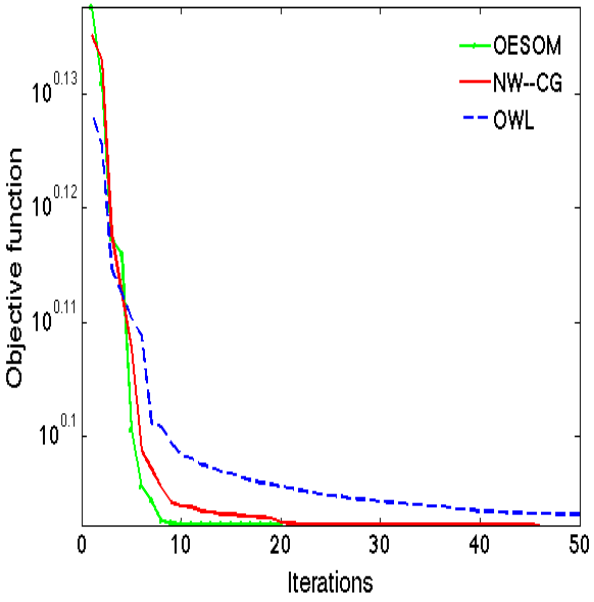
Finalmente, las figuras 4a y 4b muestran el decrecimiento de la función objetivo en cada iteración y la cardinalidad de los conjuntos activos, respectivamente, donde se observa que las direcciones generadas por el algoritmo OESOM producen un decrecimiento más significativo en la función objetivo, y además una identificación más temprana de los conjuntos activos.



(a) Tamaño de los conjuntos activos $\mathcal{A}_k$.

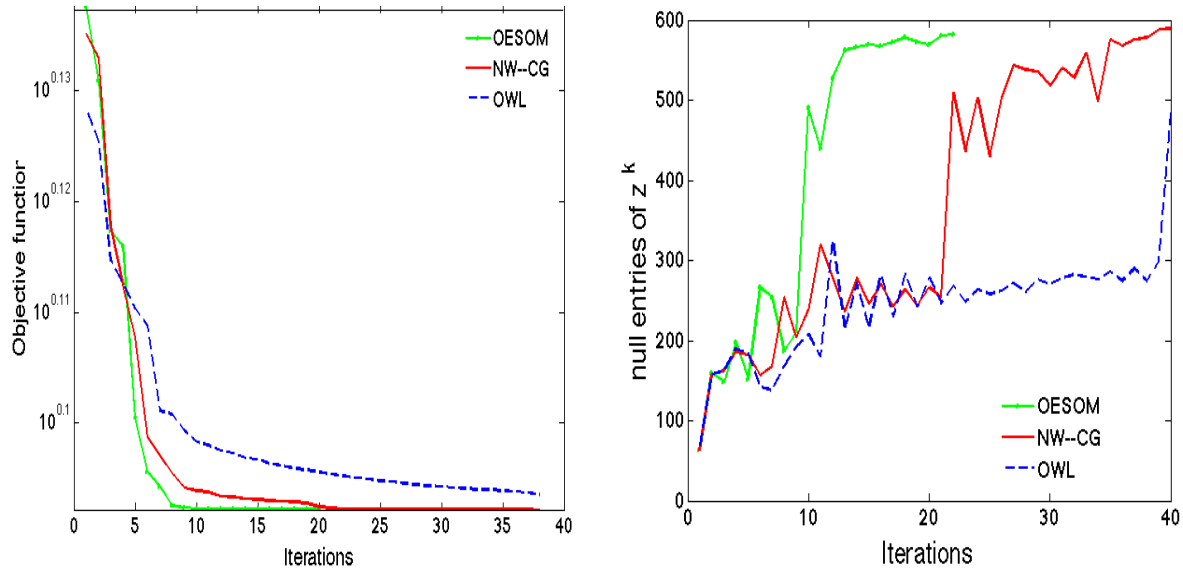


(b) Componentes nulas de x^k.



(c) Decrecimiento de la función objetivo.

Figura 3. Desempeño de los algorimtos en la resolución de problemas de control óptimo para el Experimento 2.



(a) Decrecimiento de la función objetivo con $MaxIter = 40$

(b) Cardinalidad de los conjuntos activos con $MaxIter = 40$

Figura 4. Resumen del experimento 2 con $MaxIter = 40$ como criterio de parada

En la Figura 4 se puede observar de una manera más clara la identificación temprana de los conjuntos activos desempeñada por el algoritmo **OESOM** con respecto a los otros algoritmos. Se puede justificar este comportamiento por el valor de la norma de las direcciones de descenso utilizadas en cada iteración. En la Figura (5) se presenta el gráfico del valor de la norma de la dirección de descenso en cada iteración. Los valores elevados en las primeras iteraciones pueden ser explicados por la distancia de las soluciones calculadas con respecto a la solución. Es decir, se requieren pasos más grandes para obtener un decrecimiento significativo en la función objetivo. Este hecho se ve reflejado en el pronunciado decrecimiento que experimenta la función objetivo en la primeras iteraciones, ver Figura (4a).

De las propiedades del algoritmo sabemos que este tendrá mejor desempeño cuando el valor de γ es suficientemente grande. Sin embargo, este valor no puede ser demasiado grande ya que esto conlleva a un mal condicionamiento del problema. La idea es encontrar un valor de este parámetro el cuál nos garantice un correcto funcionamiento del algoritmo pero sin afectar en el condicionamiento del problema. Esta parte del experimento busca mostrar el desempeño del algoritmo para distintos valores de γ además de compararlo con la estrategia adaptiva mostrada en el Algoritmo 7. En este experimento resolvemos el mismo problema elíptico lineal que en el caso anterior, pero variamos el valor del parámetro de regularización con el fin de entender el efecto del valor γ en la resolución numérica del problema. Uno de los principales objetivos de este experimento es mostrar la función del parámetro γ con respecto al comporta-

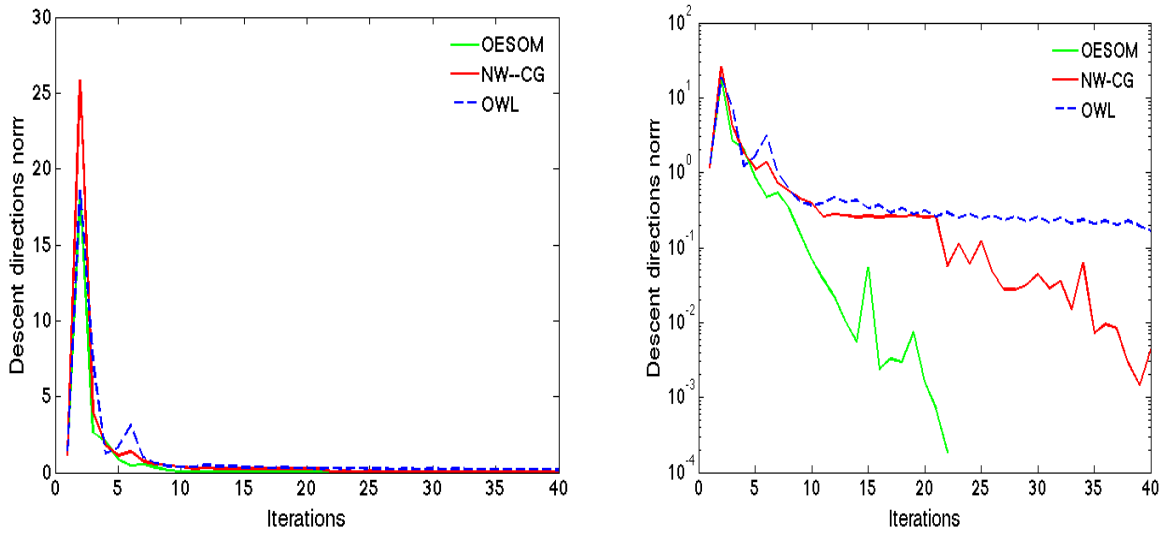


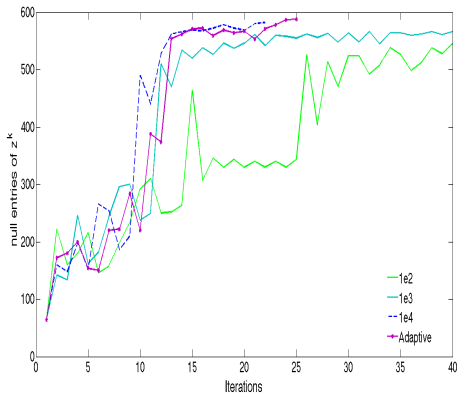
Figura 5. Norma de las direcciones de descenso en escala normal (izquierda) y escala semi–logarítmica (derecha) para el Experimento 2.

miento monótono de los conjuntos activos para un valor de x^k suficientemente cercano a la solución.

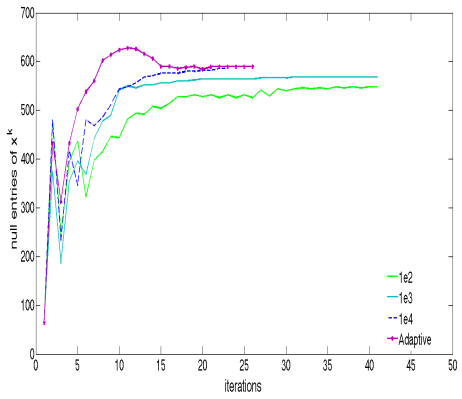
La figura 6 muestra el resumen del experimento al resolver el problema con el algoritmo **OESOM** para distintos valores del parámetro de regularización γ. Recordemos además que $\|\cdot\|_\gamma \to \|\cdot\|_1$ si $\gamma \to \infty$.

Los experimentos presentados anteriormente fueron desarrollados para mostrar el desempeño del algoritmo OESOM con respecto a los otros algoritmos. Sin embargo, es claro que ningún algoritmo puede tener el mismo comportamiento en todos los problemas. Por esto, se propone resolver el mismo problema elíptico lineal pero variando los parámetros más sensibles. El estudio de este problema nos permite evidenciar la importancia que tienen los parámetros β (parámetro de dispersión) y ν (parámetro de difusión asociado al Laplaciano) en el desempeño de los diferentes algoritmos.

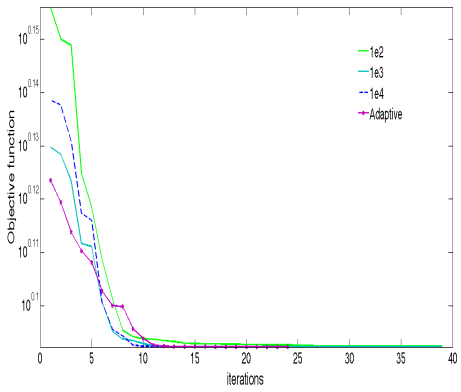
La Figura 7 muestra la cantidad de iteraciones que son necesarias para la convergencia de cada uno de los algoritmos, variando únicamente el valor de parámetro de dispersión β, cuyos valores son tomados de manera homogénea con un incremento de $1e-3$, a partir del valor $1e-3$. Tomamos un número máximo de iteraciones igual a $MaxIter = 100$. Adicionalmente, se conoce que el problema tiene asociado el valor de $\beta_0 = 0{,}09$ como el umbral de dispersión, a partir del cuál las soluciones son nulas. Se puede observar claramente que todos los algoritmos tienen un comportamiento parecido para valores mayores a este umbral.



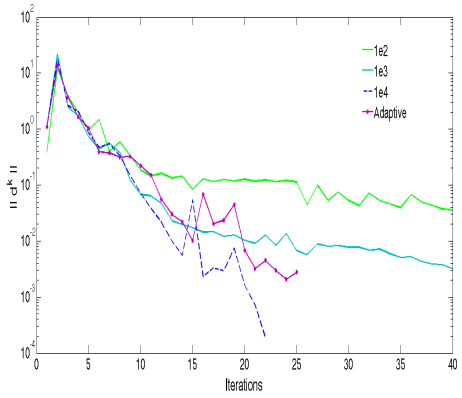
(a) Tamaño de los conjuntos activos $\mathcal{A}_k$.



(b) Componentes nulas de x^k.



(c) Decrecimiento de la función objetivo.



(d) $||d^k||$.

Figura 6. Comportamiento del algoritmo OESOM para diferentes elecciones del parámetro de regularización γ.

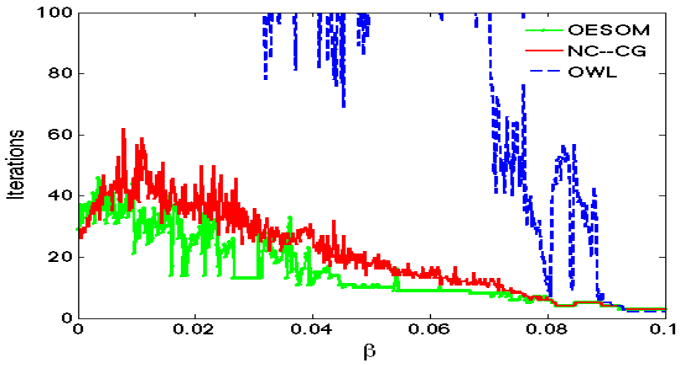


Figura 7. Número de iteraciones por experimento.

Con el fin de mostrar de manera más global el desempeño de cada uno de los algoritmos se propone resolver el problema elíptico, esta vez variando los valores de β y ν. En la Figura 8 se muestran las regiones en la cuáles cada algoritmo necesitó menos iteraciones para la convergencia a la solución del problema. Para la elección de los valores de ν se utilizó una discretización logarítmica para garantizar una mayor cantidad de valores cercanos a cero. La región roja corresponde a los valores en donde el algoritmo OESOM necesita menos iteraciones para la convergencia, la morada es del algoritmo NW–CG, la región amarilla el algoritmo OWL, y la región blanca corresponde a aquellos valores en los cuales alguno de los dos algoritmos necesita el mismo número de iteraciones que el algoritmo **OESOM** para la convergencia. Además se muestra en la línea negra el valor de β_0 en función de ν.

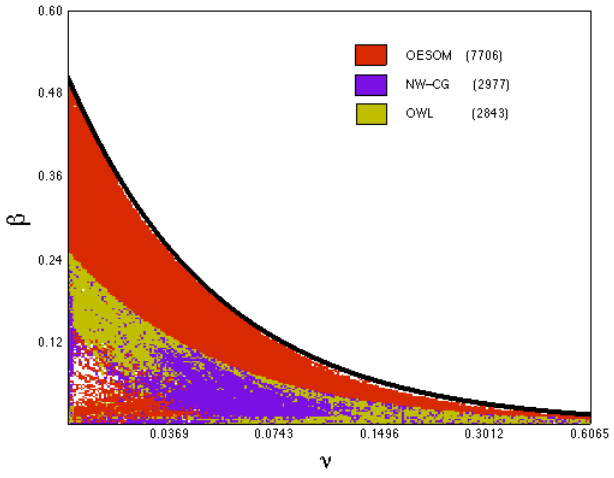


Figura 8. Región de mejor desempeño de los algoritmos para la combinación de parámetros (β, ν).

A continuación se presentan dos experimentos los cuales nos permiten extender la aplicación de nuestro algoritmo a problemas que no necesariamente son de la forma

de los problemas planteados originalmente.

3. Problema de control óptimo con restricciones sobre el control

A continuación se utiliza el ejemplo presentado en [34] con restricciones sobre el control, descrito por:

$$(P)\begin{cases} \min\limits_{y,u} \dfrac{1}{2} \parallel y - y_d \parallel^2_{L^2(D)} + \dfrac{\alpha}{2} \parallel u \parallel^2_{L^2(D)} + \beta \parallel u \parallel_{L^1(D)}, \\ \text{s.a.} \\ \qquad -\nu\Delta y = u + f, \qquad \text{en } D, \\ \qquad\quad y = 0, \qquad\quad \text{sobre } \partial D, \\ \qquad a \leq u(x) \leq b, \quad \text{c.t.p. de } D. \end{cases}$$

con los siguientes parámetros: $D = (0,1) \times (0,1)$, $\nu = 1$, $\gamma = 1000$, $\alpha = 0{,}0002$, $\beta = 0{,}002$, además $y_d = \sin(4\pi x)\cos(8\pi y)\exp(2\pi x)$ y $f = 10\cos(8\pi x)\cos(8\pi y)$, y las restricciones son $a = -10$ y

$$b = \begin{cases} 0, & \text{para } (x,y) \in [0, 1/4] \times [0,1], \\ \\ -5 + 20x, & \text{para } (x,y) \in [1/4, 1] \times [0,1]. \end{cases}$$

Debido a que se tienen restricciones sobre el control es necesario utilizar una proyección extra para garantizar que estas se cumplan.
Se define la proyección sobre el conjunto $\mathcal{U}_{ad} := \{u \colon a \leq u \leq b\}$, de la siguiente manera:

$$\mathbb{P}[u(x)] = \begin{cases} a, & \text{si } u(x) < a, \\ u(x), & \text{si } a \leq u(x) \leq b, \\ b, & \text{si } u(x) > b. \end{cases}$$

Esta proyección se debe realizarla antes de la proyección sobre el ortante definida en la ecuación (2.5). En términos numéricos la proyección está dada por la siguiente expresión

$$u = \text{máx}(\text{mín}(u, b), a).$$

El resultado obtenido con el Algoritmo OESOM tras 87 iteraciones se muestra en

la figura 9

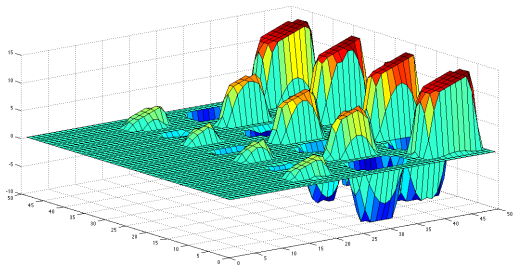


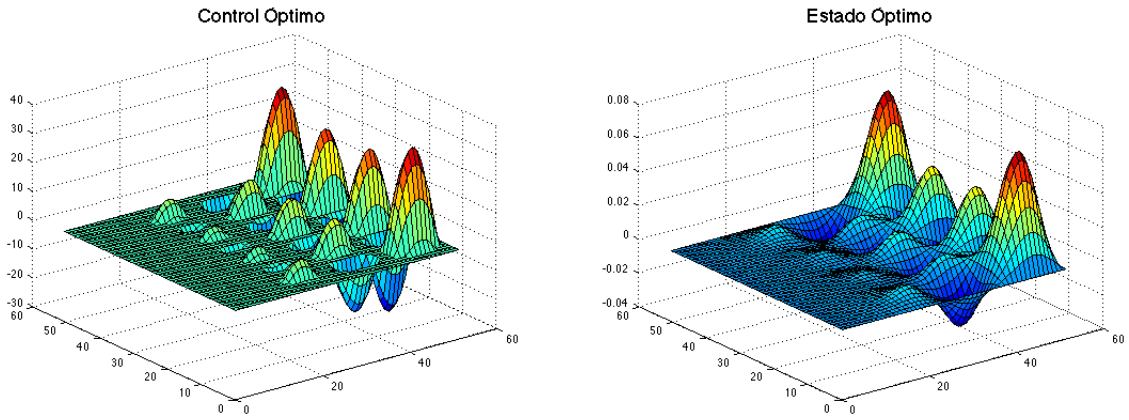
Figura 9. Control óptimo

4. Problema de control óptimo con ecuación de estado no lineal

A continuación se presenta un ejemplo de control óptimo elíptico no lineal, lo que implica que la parte regular del funcional de costo no necesariamente es convexa. El problema a resolverse es el siguiente:

$$(P)\begin{cases} \min\limits_{(y,u)} \dfrac{1}{2} \parallel y - y_d \parallel^2_{L^2(D)} + \dfrac{\alpha}{2} \parallel u \parallel^2_{L^2(D)} + \beta \parallel u \parallel_{L^1(D)}, \\ \text{s.a.} \\ \qquad -\Delta y + y^3 = u + f, \qquad \text{en } D, \\ \qquad\qquad y = 0, \qquad\quad \text{sobre } \partial D. \end{cases}$$

El algoritmo implementado para este experimento es similar al algoritmo OESOM. El único cambio que se realiza es con respecto a la solución de la ecuación de estado, y del estado adjunto. Utilizamos los siguiente parámetros: $\alpha = 0{,}0002$, $\beta = 0{,}002$, $\gamma = 1000$ y las funciones $y_d = \sin(4\pi x)\cos(8\pi y)\exp(2x)$ y $f = 10\cos(8\pi x)\cos(8\pi y)$. El algoritmo necesita 56 iteraciones para la convergencia y el control resultante se muestra en la figura 10a y el estado óptimo asociado en la figura 10b



(a) Control óptimo **(b)** Estado óptimo asociado.

Figura 10. Resultados obtenidos con el algoritmo OESOM para un problema de control óptimo semi–lineal

En la Figura 10 se observa el control óptimo el cuál tiene claramente una estructura dispersa, y su estado óptimo asociado.

5. Aplicación al control óptimo de dinámica poblacional espacial

Está sección está dedica a la aplicación de los algoritmos desarrollados en los capítulos anteriores en un problema de control óptimo de dinámica poblacional, específicamente con el modelo de Fisher.

Una dinámica poblacional simplificada de una especie en un espacio bidimensional se puede modelar a través de la ecuación de Fisher. Este modelo considera la dispersión del ente biológico mediante un proceso de difusión espacial, mientras que la evolución está gobernada por el crecimiento logístico. El modelo está dado de la siguiente manera:

$$\frac{\partial y}{\partial t} - \nu \Delta y + uy = ry\left(1 - \frac{y}{\kappa}\right), \tag{4.1}$$

$$\frac{\partial y}{\partial \vec{n}} = 0, \tag{4.2}$$

$$y(x, 0) = y_0. \tag{4.3}$$

donde:

ν : velocidad de difusión	u : tasa de mortalidad a ser controlada
r : tasa de creciemiento	κ : capacidad del medio ambiente
y_0 : condición inicial	

El objetivo de este ejemplo consiste en controlar el crecimiento de esta población a través de un proceso de optimización de la Ecuación de Fisher, donde la variable a optimizar es la tasa de mortalidad u. En otras palabras, el problema consiste en encontrar la estrategia óptima para eliminar la población de tal manera que al final del proceso obtengamos un estado deseado y_d utilizando la menor cantidad de recursos. Formulamos el problema de control óptimo como:

$$\begin{cases} \min\limits_{(u,y)} \ \dfrac{1}{2}\displaystyle\int_0^T \int_\Omega (y(x,t) - y_d(x,t))^2 \, dxdt + \frac{\alpha}{2}\int_\Omega (u(x))^2 \, dx + \beta \int_\Omega |u(x)| dx. \\ \\ \text{sujeto a:} \\ \\ \text{la ecuación (4.1).} \end{cases} \tag{4.4}$$

De las secciones anteriores sabemos que el resultado será un control sparse, el cuál garantizará encontrar una estrategia más real para la eliminación de cierta especie. En términos prácticos significa que esta estrategia tiene que ser ubicada solo en ciertas regiones del dominio. Es decir en un problema de agricultura, para controlar una cierta enfermedad es necesario aplicar el insecticida en zonas específicas lo que se traduce en menos trabajo por parte de las personas que aplican la estrategia y en una cantidad menor de insecticida a utilizarse.

5.1. Condiciones de optimalidad de primer orden

La teoría de optimización nos permite establecer un sistema de optimalidad el cuál caracteriza las soluciones del problema (4.4).

PROPOSICIÓN 4.1. *Sea $\bar{u}$ solución de* (4.4) *existen $\bar{y}$ y $\bar{p}$ el estado óptimo y el estado*

adjunto asociados tal que satisfacen el siguiente sistema de optimalidad.

$$
\begin{array}{ll}
\textit{(Ecuación de Estado)} & \begin{cases} \frac{\partial \bar{y}}{\partial t} - \nu \Delta \bar{y} + u\bar{y} = r\bar{y}\left(1 - \frac{\bar{y}}{\kappa}\right) & \textit{en } D \times [0,T], \\ \dfrac{\partial \bar{y}}{\partial \vec{n}} = 0 & \textit{sobre } \partial D \times [0,T], \\ \bar{y}(0,x) = y_0 & \textit{en } D. \end{cases} \\
\textit{(Ecuación Adjunta)} & \begin{cases} -\frac{\partial \bar{p}}{\partial t} - \nu \Delta \bar{p} - r\bar{y} + \dfrac{2r}{\kappa}\bar{y}\bar{p} = y_d - y & \textit{en } D \times [0,T], \\ \dfrac{\partial \bar{p}}{\partial \vec{n}} = 0 & \textit{sobre } \partial D \times [0,T], \\ \bar{p}(T,x) = 0 & \textit{en } D. \end{cases} \\
\textit{(Ecuación del Gradiente)} & \alpha \bar{u} - \displaystyle\int_0^T \bar{y}\bar{p} + \bar{q} = 0, \\
\textit{(Ecuaciones de Complementariedad)} & \begin{cases} |\bar{q}(x)| \leq \beta & \textit{si } \bar{u}(x) = 0, \\ \bar{q}(x) = \textit{sign}(u(x)) & \textit{si } \bar{u}(x) \neq 0. \end{cases}
\end{array}
\tag{4.5}
$$

5.2. Resolución numérica

Resolvemos el siguiente problema:

$$
(P) \begin{cases} \min\limits_{(y,u)} & \frac{1}{2} \parallel y - y_d \parallel^2_{L^2(D)} + \dfrac{\alpha}{2} \parallel u \parallel^2_{L^2(D)} + \beta \parallel u \parallel_{L^1(D)} \\ \text{sujeto a:} & \\ & \begin{cases} \frac{\partial y}{\partial t} - \nu \Delta y + uy = ry\left(1 - \frac{y}{\kappa}\right) & \text{en } D \times [0,T], \\ \dfrac{\partial y}{\partial \vec{n}} = 0 & \text{sobre } \partial D \times [0,T], \\ y(x,0) = y_0 & \text{en } D. \end{cases} \end{cases}
$$

donde D es el círculo de centro 0 y radio 1, y ∂D la frontera del D. El estado inicial y_0 es la indicatriz de la bola con centro en cero y radio 0.25, y el estado deseado $y_d = 0{,}2$. Los parámetros $\alpha = 0{,}002$ y $\beta = 0{,}02$. El dominio circular se lo ha escogido basándose en que en la práctica puede representar a una caja petri la cuál contiene un cultivo de una cierta especie bacteriana. Además se consideran los siguientes parámetros para la ecuación de Fisher:

$\nu = 0{,}001$ $\quad$ u : tasa de mortalidad a ser controlada

$r = 0{,}15$ $\quad$ $\kappa = 1$ (Capacidad de carga del territorio)

$$y_0(x) = \begin{cases} 1 & \text{si } x \in B(0, 0{,}25), \\ 0 & \text{caso contrario.} \end{cases}$$

Para la discretización espacial de este problema se utiliza el Método de Elementos Finitos lineales a trozos, ver [29], a diferencia del esquema de diferencias finitas utilizada en experimentos anteriores, principalmente debido a la geometría del Dominio. Para la discretización se uso una malla en coordenadas polares, sobre este espacio $\tilde{D} = [0, R] \times [0, 2\,\pi]$. En la Figura 11 se muestra la estructura de la malla generada.

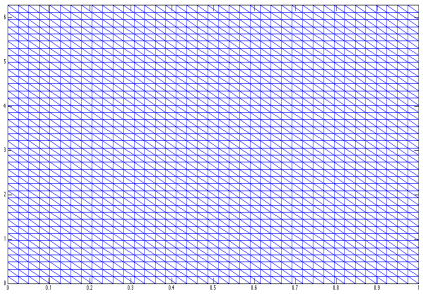


Figura 11. Malla generada en coordenadas polares.

Esta malla nos permite establecer las estructuras necesarias para el cálculo de las matrices de rigidez y de masa necesarias para la discretización del problema. Realizando la transformación correspondiente se puede establecer la malla en coordenadas rectangulares. La Figura 12 muestra la malla resultante de esta transformación sobre el dominio circular D.

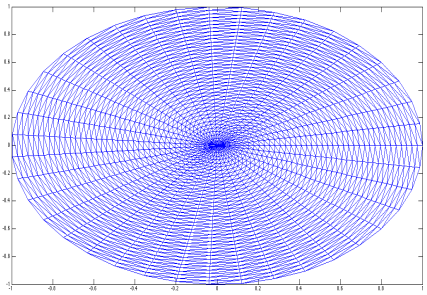
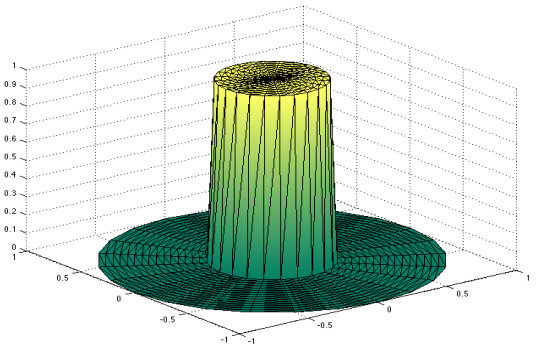


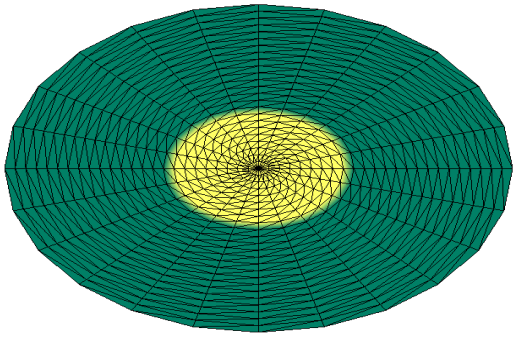
Figura 12. Mallado del dominio ciruclar D.

Las matrices de masa y rigidez son generadas con a ayuda de la función *assema* de MATLAB correspondiente al PDEtoolbox.

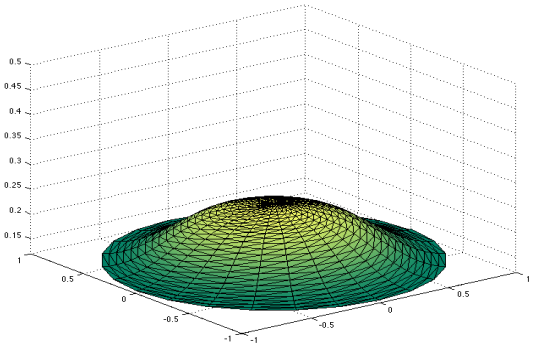
A continuación se presenta los resultados obtenidos para estos parámetros. El control óptimo, el estado óptimo y con fines comparativos el estado no controlado.



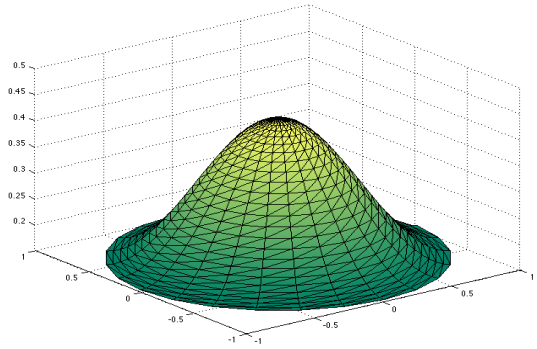
(a) Control óptimo (Tasa de mortalidad óptima localizada)



(b) Control óptimo *Sparse*



(c) Estado Óptimo al tiempo t=0.67



(d) Estado no controlado en el tiempo t=0.67.

Figura 13. Resumen del experimento para un problema de control óptimo de dinámica poblacional

Podemos notar que existe una diferencia significativa entre el estado óptimo de la

Figura 13c y el estado no controlado 13d, teniendo en cuenta que el estado deseado $y_d = 0{,}2$. Con respecto a la estructura del control podemos observar en las Figuras 13a y 13b la acción localizada de la estrategia óptima para eliminación de la especie.

Capítulo 5

Conclusiones

Los problemas estudiados en este trabajo, provienen de una gran variedad de aplicaciones tanto en espacios de dimensión finita como en espacios funcionales. Específicamente, las aplicaciones en espacios funcionales, y en especial problemas de control óptimo pueden ser aplicados a varias ramas de la ingeniería. Por ejemplo, en localización de actuadores con materiales pizo–eléctricos en placas, en modelos de pesca, entre otras. En los primeros capítulos se mencionó varias estrategias numéricas utilizadas actualmente para la resolución numérica de estos problemas, entre las cuales se pueden distinguir tres grandes grupos; los que utilizan información de subgradientes, aquellos que utilizan regularizaciones, y los que proponen resolver problemas con restricciones. Dentro del grupo de los algoritmos que utilizan la información proporcionada por los subgradientes, existen dos algoritmos NW–CG y OWL los cuales utilizan información proporcionada por las direcciones ortantes y definen subespacios para realizar el proceso de optimización. La principal idea de estos algoritmos proviene del hecho de que la norma ℓ_1 restringida a un ortante es diferenciable. Es decir, estos algoritmos buscan un ortante activo donde realizan el proceso de optimización, y que garantice que la función es diferenciable en ese conjunto. Es justamente de este hecho que la noción de dirección ortante se ajusta a las necesidades teóricas y numéricas.

Los algoritmos mencionados anteriormente trabajan dentro del enfoque proporcionado por los algoritmos de optimización los cuales necesitan de una dirección de descenso para su ejecución. Una de las posibles elecciones para esta dirección está asociada al gradiente de la función objetivo la cual es conocida como la dirección del descenso más profundo. Sin embargo, para el problema tratado en este trabajo no es posible utilizar esta información directamente, por lo que se define una dirección tipo descenso más profundo que cumple el objetivo de proporcionar decrecimiento de la función objetivo. Esta está descrita en términos de un elemento del subdiferencial. Las direcciones ortantes, por otro lado, además de caracterizar los ortantes en los que la

función objetivo es diferenciable, también son elemento del subdiferencial de la norma ℓ_1. Estas direcciones corresponden a la contribución de la norma ℓ_1 en el subgradiente de la función objetivo. Utilizar la dirección tipo descenso más profundo en estos algoritmos sería poco eficiente, debido a que resultados de optimización sabemos que la convergencia de este algoritmo es lineal, por esta razón los algoritmos NW–CG y OWL proponen utilizar información de segundo orden. Es decir un método de Newton en el caso de NW–CG, y un esquema Quasi–Newton: L–BFGS en el caso del algoritmo OWL, para obtener un orden de convergencia mayor. A pesar de que resultados de convergencia no han sido demostrados, de los experimentos numéricos sabemos que eso ocurre. Cabe recalcar que la información de segundo orden utilizada por estos algoritmos corresponde únicamente a la parte regular de la función objetivo.

En el caso del paso de descenso, el cuál nos garantiza un decrecimiento suficiente en la función objetivo debe garantizar que las iteraciones subsiguientes permanezcan en el mismo ortante. Por este motivo se propone una regla de decrecimiento que involucra una proyección sobre el ortante, la que nos permita encontrar el valor del paso de descenso tomando en cuenta dicha condición.

La familia de algoritmos propuesta utiliza información de segundo orden asociada a la parte no diferenciable adicional a la información correspondiente a la parte regular. La información de segundo orden asociada a la norma ℓ_1 se obtiene a través del uso de una regularización tipo Huber. Esta regularización permite captar la mayor cantidad de información debido a que no es excesivamente suave. Para obtener dicha información se representa mediante nociones más generales de derivada.

La inclusión de esta información en nuestros algoritmos, se puede justificar notando que la norma ℓ_1, a pesar de no ser diferenciable en el sentido clásico, es dos veces diferenciable en el sentido de las distribuciones. Inclusive su derivada generalizada en cero está dada por la función *delta de Dirac*. Este hecho nos lleva a pensar que la información de segundo orden asociada al punto cero es importante. Además teniendo en cuenta que las soluciones tendrán mucho valores iguales a cero, la información no considerada es aún mayor. Esta información es justamente la que otros algoritmos desprecian, y por lo cuál podemos decir que nuestro algoritmo considera información *completa* o *enriquecida* de segundo orden. Una de las principales consecuencias de utilizar esta información enriquecida para el cálculo de las direcciones de descenso es el efecto de escalamiento sobre las componentes de la dirección. Este efecto se puede entender como una relajación de las condiciones establecidas en los algoritmos OWL y NW–CG. El algoritmo propuesto posee varias propiedades importantes, las cuales nos permiten afirmar que el desempeño de nuestro algoritmo es mejor que los anteriores y además es confirmado en los test numéricos. Estas propiedades son básicamente sobre condiciones de las direcciones de descenso, la monotonía de los conjuntos activos y la

equivalencia con el método Newton Semi–Smooth. Además de tener propiedades teóricas importantes, el algoritmo tiene un mejor desempeño en términos de la resolución numérica de los problemas presentados en el capítulo de test numéricos.

Capítulo 6

Apéndice

1. Resultados de Cálculo subdiferencial

Los resultados presentados a continuación con respecto a las reglas del cálculo subdiferencial fueron tomados de [26]. Y los resultados con respecto a las funciones *semi–suaves* fueron tomados de [36].

Teorema 5. *Sea* $f_i : \mathbb{R} \to \mathbb{R}$, *para* $i = 1, 2$ *Lipschitz continuas alrededor de* $\bar{x}$. *Entonces se tiene*

$$\partial \left(f_1 \cdot f_2\right)(\bar{x}) = \partial \left(f_2(\bar{x}) f_1 + f_1(\bar{x}) f_2\right)(\bar{x}).$$

Si, adicionalmente, una de las funciones (por ejemplo f_1*) es estrictamente diferenciable en* $\bar{x}$, *entonces*

$$\partial \left(f_1 \cdot f_2\right)(\bar{x}) = f_1'(\bar{x}) f_2(\bar{x}) + \partial(f_1(\bar{x}) f_2)(\bar{x}).$$

Teorema 6. *Sea* $f_i : \mathbb{R} \to \mathbb{R}$, *para* $i = 1, 2$ *Lipschitz continuas alrededor de* $\bar{x}$. *Entonces se tiene*

$$\partial \left(\frac{f_1}{f_2}\right)(\bar{x}) = \frac{\partial \left(f_2(\bar{x}) f_1 - f_1(\bar{x}) f_2\right)(\bar{x})}{[f_2(\bar{x})]^2}.$$

Si, adicionalmente, una de las funciones (por ejemplo f_1*) es estrictamente diferenciable en* $\bar{x}$, *entonces*

$$\partial \left(\frac{f_1}{f_2}\right)(\bar{x}) = \frac{f_1'(\bar{x}) f_2(\bar{x}) + \partial(-f_1(\bar{x}) f_2)(\bar{x})}{[f_2(\bar{x})]^2}.$$

En particular, a continuación se presenta el siguiente Corolario

Corolario 1. *Sea* $f : \mathbb{R} \to \mathbb{R}$ *sea Lipschitz continua alrededor de* $\bar{x}$ *con* $f(\bar{x}) \neq 0$. *Entonces se tiene*

$$\partial \left(\frac{1}{f}\right)(\bar{x}) = -\frac{\partial f(\bar{x})}{f^2(\bar{x})}.$$

Teniendo en cuenta que la derivada en el sentido de Newton es un elemento del subdiferencial de una función, podemos utilizar los siguientes resultados, para la demostración de existencia de un elemento del subdiferencial.

DEFINICIÓN 6.1. *Sea $V \subset \mathbb{R}^n$ un conjunto abierto y no vacío. La función $f : V \to \mathbb{R}^m$ es semi–suave en $x \in V$, si esta es Lipschitz continua cerca de x y el siguiente límite existe para todo $s \in \mathbb{R}^n$:*

$$\lim_{\substack{M \in \partial f(x+\tau d) \\ d \to s, \tau \to 0^+}} Md.$$

Si f es semi–suave para todo $x \in V$, se dice que f es semi–suave en V.

Teorema 7. *Sea $f : \mathbb{R} \to \mathbb{R}$ una función lineal a trozos. Entonces f es semi–suave.*

Teorema 8. *Sea $V \subset \mathbb{R}^n$ y $W \subset \mathbb{R}^l$ conjuntos abiertos. Sea $g : V \to W$ una función semi–suave en $x \in V$ y $h : W \to \mathbb{R}^m$ una función semi–suave en $g(x)$ con $g(V) \subset W$. Entonces la función $f = h \circ g : V \to \mathbb{R}^m$ es semi–suave en x. Además se tiene que:*

$$f'(x) = h'(g(x))g'(x).$$

2. Cálculo de subdiferenciales

En los siguientes lemas se muestra el cálculo de los subdiferenciales para las funciones valor absoluto y máximo.

DEFINICIÓN 6.2. *Sea X y Y dos espacios de Banach. Adicionalmente, sea $D \subset X$ un conjuntos abierto y no vacío y $f : D \leftarrow Y$ una función dada. f se dice Newton diferenciable en D si existe $G_f : D \to \mathcal{L}(X, Y)$ tal que*

$$\lim_{|h| \to 0} \frac{\| f(x+h) - f(x) - G_f(x+h) \|_Y}{\| h \|_X} = 0 \tag{6.1}$$

Lema 9. *Demostrar*

$$\partial(|x|) = \begin{cases} -1 & si\ x < 0, \\ 1 & si\ x > 0, \\ \delta & si\ x = 0. \end{cases}$$

con $\delta \in [-1, 1]$.

Demostración. Claramente la función valor absoluto no es diferenciable en 0. Notamos

$$G_\delta(x) = \begin{cases} -1 & \text{si } x < 0, \\ 1 & \text{si } x > 0, \\ \delta & \text{si } x = 0. \end{cases}$$

con $-1 \leq \delta \leq 1$, entonces demostramos que $G_\delta(x)$, satisface la ecuación (6.1),

$$\lim_{|h|\to 0} \frac{\| f(x+h) - f(x) - G_\delta(x+h)h \|_Y}{\| h \|_X} = \lim_{|h|\to 0} \frac{||h| - G_\delta(h) \cdot h|}{|h|}$$

De la definición sabemos que $|h| > 0$, analizamos dos casos.
Suponemos $h < 0$, entonces

$$\lim_{|h|\to 0} \frac{|-h-(-1)h|}{-h} = \lim_{|h|\to 0} \frac{-h+h}{-h} = \lim_{|h|\to 0} 0 = 0$$

De manera similar para el caso en que $h > 0$ tenemos

$$\lim_{|h|\to 0} \frac{|h-(1)h|}{h} = \lim_{|h|\to 0} \frac{|h-h|}{h} = \lim_{|h|\to 0} 0 = 0$$

De esta forma, podemos concluir que la familia de funciones definidas por G_δ representan la derivada de Newton para $f(x) = |x|$. □

Para el análisis tomaremos un elemento de G_δ, entonces

$$\partial(|x|) = \begin{cases} -1 & \text{si } x < 0 \\ 1 & \text{si } x \geq 0 \end{cases}$$

Lema 10. *Demostrar*

$$\partial(\text{máx}\{0, x\}) = \begin{cases} 0 & si\ x < 0, \\ 1 & si\ x > 0, \\ \delta & si\ x = 0. \end{cases}$$

con $\delta \in [0, 1]$.

Demostración. Notamos al conjunto de funciones $H_\delta(x)$

$$H_\delta(x) = \begin{cases} 0 & \text{si } x < 0, \\ 1 & \text{si } x > 0, \\ \delta & \text{si } x = 0. \end{cases}$$

Verificamos que la familia de funciones satisface (6.1)

$$\lim_{|h|\to 0} \frac{\| f(x+h) - f(x) - G(x+h)h \|_Y}{\| h \|_X} = \lim_{|h|\to 0} \frac{|\max(0,h) - G(h)\cdot h|}{|h|}$$

Al igual que para la función valor absoluto, analizamos dos casos. Suponemos $h < 0$, entonces

$$\lim_{|h|\to 0} \frac{|0-(0)h|}{-h} = \lim_{|h|\to 0} \frac{0-0}{-h} = \lim_{|h|\to 0} 0 = 0$$

Para $h > 0$

$$\lim_{|h|\to 0} \frac{|h-(1)h|}{h} = \lim_{|h|\to 0} \frac{h-h}{h} = \lim_{|h|\to 0} 0 = 0$$

Por lo tanto, podemos concluir que la familia definida por $H_\delta(x)$, es la derivada de Newton de $f(x) = \max(0,x)$. □

Es suficiente calcular la derivada para esta función puesto que se tiene la siguiente igualdad:

$$\max(h,g) = \max(0,g-h) \qquad \text{para cualquier par de funciones } h,g.$$

Tomaremos como la derivada de la función máximo, la función definida por:

$$\partial\left(\max(0,x)\right) = \begin{cases} 0 & \text{si } x \leq 0, \\ 1 & \text{si } x > 0. \end{cases}$$

Es claro que es suficiente calcular la derivada para esta función puesto que se tiene la siguiente igualdad:

$$\max(h,g) = \max(0,g-h) \qquad \text{para cualquier par de funciones } h,g.$$

3. Teoría de dualidad de Fénchel

DEFINICIÓN 6.3. *Sean V y Y dos espacios de Banach con espacios duales V^* y Y^*, respectivamente. Sea $\Lambda \in \mathcal{L}(V,Y)$ dada y $\Lambda^* \in \mathcal{L}(Y^*,V^*)$ el operador adjunto asociado. Además, sea $J : V \to \mathbb{R}$ tal que $J = J(x,\Lambda x)$. Se tiene el problema*

$$\inf_{x\in V} J(x,\Lambda x), \tag{6.2}$$

El problema dual asociado a (6.2) *es:*

$$\sup_{q^* \in Y^*} \; -J(\Lambda^* q^*, -q^*). \tag{6.3}$$

El siguiente teorema presenta las hipótesis necesarias para que no exista brecha de dualidad. Es decir que los valores de las funciones objetivo en el valor óptimo correspondiente tanto para el problema primal como para el dual son iguales.

Teorema 9. *Sea J convexo. Asumiendo que el problema* (1.3) *tiene solución y que existe $x_0 \in V$ tal que $J(x_0, \Lambda x_0) < \infty$ y que la función $q \to J(x_0, q)$ es continua en Λx_0, entonces no existe brecha de dualidad, es decir:*

$$\inf_{x \in V} J(x, \Lambda x) = \sup_{q^* \in Y^*} -J(\Lambda^* q^*, -q^*).$$

y el problema dual (6.3) *tiene al menos una solución $\bar{q}^*$.*

Las funciones conjugadas se definen de la siguiente manera:

DEFINICIÓN 6.4. *Sea $f : V \to \mathbb{R} \cup \{-\infty, +\infty\}$ una función convexa. La función convexa conjugada $f^* : V^* \to \mathbb{R} \cup \{-\infty, +\infty\}$ se define por*

$$f^*(x^*) = \sup_{x \in V} \; \{\langle x^*, x \rangle - f(x)\}, \quad \text{para todo } x^* \in V^*.$$

donde $\langle \cdot, \cdot \rangle$ es el producto en dualidad.

Con esta nueva definición se puede reformular el Teorema **??** como:

Teorema 10. *Sean V y Y dos espacios de Banach. Sean $\mathcal{F}$ y $\mathcal{G}$ funciones propias y semicontinuas inferiores, tales que existe $v_0 \in V$ que satisface $\mathcal{F}(v_0), \mathcal{G}(\Lambda v_0) < +\infty$ con $\mathcal{G}$ continua en Λv_0. Si además, se tiene que $\mathcal{F}(x) + \mathcal{G}(\Lambda x) \to +\infty$ cuando $\| x \| \to \infty$ y que V es un espacio reflexivo, entonces los problemas primal y dual admiten (al menos) una solución $\bar{x}$ y $\bar{q}$ respectivamente, tales que*

$$\mathcal{F}(\bar{x}) + \mathcal{G}(\Lambda \bar{x}) = -\mathcal{F}^*(\Lambda^* \bar{q}) - \mathcal{G}^*(\bar{q}),$$

donde $\mathcal{F}^*$ y $\mathcal{G}^*$ son las funciones conjugadas de $\mathcal{F}$ y $\mathcal{G}$ respectivamente.

Teorema 11. *Suponga que las hipótesis del Teorema 10 se satisfacen, es decir, se tiene que*

$$\inf_{x \in V} \; \{\mathcal{F}(x) + \mathcal{G}(\Lambda x)\} = \inf_{q \in V^*} \; \{-\mathcal{F}^*(-\Lambda^* q) + \mathcal{G}^*(q)\}.$$

Si este número es finito, entonces todas las soluciones $\bar{x}$ y $\bar{q}$ de los problemas primal y dual respectivamente, satisfacen las siguientes condiciones extremales.

$$\begin{aligned} -\Lambda^* \bar{q} &\in \partial \mathcal{F}(\bar{x}), \\ -\bar{q} &\in \partial \mathcal{G}(\Lambda \bar{x}). \end{aligned}$$

Bibliografía

[1] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, Singapore, 2006.

[2] R. Byrd, G. Chin, W. Neveitt, y J. Nocedal. On the use of stochastic hessian information in optimization methods for machine learning. *SIAM Journal on Optimization*, 21(3):977–995, 2011.

[3] R. Byrd, G. Chin, J. Nocedal, y Y. Wu. Sample size selection in optimization methods for machine learning. *Mathematical Programming*, 134(1), 2011.

[4] R Byrd, Gillian M Chin, Will Neveitt, y Jorge Nocedal. On the use of stochastic hessian information in unconstrained optimization. *Manuscript, Northwestern University*, 2010.

[5] Richard H Byrd, Gillian M Chin, Jorge Nocedal, y Figen Oztoprak. A family of second-order methods for convex l1-regularized optimization. *Unpublished: Optimization Center: Northwestern University, Tech Report*, 2012.

[6] Eduardo Casas, Christopher Ryll, y Fredi Tröltzsch. Sparse optimal control of the Schlögl and Fitzhugh–Nagumo systems. *Computational Methods in Applied Mathematics*, 13(4):415–442, 2013.

[7] Michael Collins y Terry Koo. Discriminative reranking for natural language parsing. *Computational Linguistics*, 31(1):25–70, 2005.

[8] J.C. De los Reyes, E. Loayza, y P. Merino. Second-order orthant-based methods with enriched Hessian information for sparse ℓ_1-optimization. *arXiv*, (1407.1096), 2014.

[9] Juan Carlos De Los Reyes y Sergio González. Path following methods for steady laminar bingham flow in cylindrical pipes. *ESAIM: Mathematical Modelling and Numerical Analysis*, 43(1):81, 2009.

[10] I. Ekeland y R. Témam. *Convex Analysis and Variational Problems*. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 1999.

[11] Y. Favennec, V. Labbé, y F. Bay. Induction heating processes optimization a general optimal control approach. *Journal of Computational Physics*, 187(1):68 – 94, 2003.

[12] K. Fountoulakis y J. Gonzio. A second-order method for strongly convex ℓ_1-regularization problems. *arXiv*, (1306.5386v3), 2013.

[13] Andrew G. y Gao J. Scalable training of l1-regularized log-linear models. *In Proceedings of the Twenty Fourth Conference on Machine Learning (ICML)*, 2007.

[14] Gene H Golub y Charles F Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[15] Sergio González. *Semismooth Newton and path following methods for the numerical simulation of Bingham fluids*. PhD thesis, Escuela Politécnica Nacional, 2008.

[16] M. Hinze, R. Pinnau, M. Ulbrich, y S. Ulbrich. *Optimization with PDE constraints*, volume 23 of *Mathematical Modelling: Theory and Applications*. Springer, New York, 2009.

[17] Kwangmoo Koh, Seung-Jean Kim, y Stephen Boyd. An Interior-Point Method for Large-Scale L1-Regularized Logistic Regression. *J. Mach. Learn. Res.*, 8:1519–1555, December 2007.

[18] Hang-Chin Lai y Lai-Jiu Lin. Moreau-rockafellar type theorem for convex set functions. *Journal of mathematical analysis and applications*, 132(2):558–571, 1988.

[19] Su I. Lee, Honglak Lee, Pieter Abbeel, y Andrew Y. Ng. Efficient L1 Regularized Logistic Regression. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI-06)*. 2006.

[20] Su-In Lee, Honglak Lee, Pieter Abbeel, y Andrew Y Ng. Efficient ℓ_1 regularized logistic regression. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, pág. 401. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.

[21] Peiyang Li, Peng Xu, Rui Zhang, Lanjin Guo, y Dezhong Yao. L1 norm based common spatial patterns decomposition for scalp eeg bci. *Biomedical engineering online*, 12:77, 2013.

[22] Nicolai Meinshausen y Bin Yu. Lasso-type recovery of sparse representations for high-dimensional data. *The Annals of Statistics*, págs. 246–270, 2009.

[23] Andre Milzarek y Michael Ulbrich. A semismooth newton method with multidimensional filter globalization for l_1-optimization. *SIAM Journal on Optimization*, 24(1):298–333, 2014.

[24] Thomas P Minka. A comparison of numerical optimizers for logistic regression. *Unpublished draft*, 2003.

[25] Tom M. Mitchell. *Machine Learning*. Mc Graw Hill, Boston, Massachusetts, 1997.

[26] Boris S. Mordukhovich y Yongheng Shao. On nonconvex subdifferential calculus in banach spaces, 1995.

[27] Mila Nikolova y Michael Ng. Fast image reconstruction algorithms combining half-quadratic regularization and preconditioning. In *Image Processing, 2001. Proceedings. 2001 International Conference on*, volume 1, págs. 277–280. IEEE, 2001.

[28] J. Nocedal y S. J. Wright. *Numerical Optimization*. Springer-Verlag, New York, 1999.

[29] Alfio Quarteroni. *Numerical models for differential problems*. MS&A. Springer, Milan, Berlin, New York, 2009. Trad. de : Modellistica numerica per problemi differenziali.

[30] Srikanth Ryali, Kaustubh Supekar, Daniel A. Abrams, y Vinod Menon. Sparse logistic regression for whole-brain classification of fmri data. *NeuroImage*, 51(2):752 – 764, 2010.

[31] Mark Schmidt, Glenn Fung, y Rmer Rosales. Fast optimization methods for l1 regularization: A comparative study and two new approaches. In *Machine Learning: ECML 2007*, págs. 286–297. Springer, 2007.

[32] Mark Schmidt, Glenn Fung, y Rómer Rosales. Fast Optimization Methods for L1 Regularization: A Comparative Study and Two New Approaches. In *Machine Learning: ECML 2007*, volume 4701 of *Lecture Notes in Computer Science*, chapter 28, págs. 286–297. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[33] Suvrit Sra, Sebastian Nowozin, y Stephen Wright. *Optimization for Machine Learning*. Massachusetts Institute of Tecnology, Massachusetts, USA, 2012.

[34] Georg Stadler. Elliptic optimal control problems with L^1-control cost and applications for the placement of control devices. *Comput. Optim. Appl.*, 44(2):159–181, 2009.

[35] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, págs. 267–288, 1996.

[36] M. Ulbrich. *Semismooth Newton Methods for Variational Inequalities and Constrained Optimization Problems in Function Spaces*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 2011.

[37] James Hardy Wilkinson. *The algebraic eigenvalue problem*, volume 87. Clarendon Press Oxford, 1965.

[38] G. Yuan, K. Chang, y C. Hsieh, C.and Lin. A comparison of optimization methods and software for large-scale l1-regularized classification. *Journal of Machine Learning Research*, (11):3183–3234, 2010.