PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

ANÁLISIS DEL ALGORITMO FISTA ORIENTADO A MEJORAR LA VELOCIDAD DE CONVERGENCIA

Tesis para obtener el título profesional de Ingeniero Electrónico

AUTOR:

Gabriel Ramirez Orihuela

ASESOR:

Paul Antonio Rodríguez Valderrama

Lima, Febrero, 2022

Resumen

Los problemas lineales inversos existen en numerosas ramas de la ciencia e ingeniería, lo cual genera la necesidad de definir algoritmos de solución eficientes, que requieran poco costo computacional y converjan en el menor número de iteraciones. Se desea recuperar información original a la cual no se tiene acceso lo más similarmente posible y con dimensiones reducidas, produciendo así una disminución en el uso de recursos computacionales y por ende en el tiempo de ejecución. Esto es de particular importancia debido a que el tamaño de las señales se encuentra en constante aumento y su manipulación puede resultar muy costosa.

Se estudia el algoritmo de optimización de primer orden FISTA (*Fast Iterative Shrinkage-Thresholding Algorithm*), el cual es utilizado en problemas inversos cuya solución se resume a la minimización de funciones convexas empleando información de la gradiente y de iteraciones previas. En este contexto, se analizan métodos que buscan la optimización del algoritmo por medio de tamaños de paso adaptativos para delimitar el paso de la gradiente y una mejor solución inicial mediante la reducción de dimensiones a través de las técnicas conocidas como *Screening* y *Warm Start*, produciendo así datos más *sparse*. Además, se comprueba la eficacia de los métodos desarrollados por medio de un algoritmo generalizado, en el cual son evaluados datos aleatorios generados sintéticamente e imágenes, con el fin de obtener la mejor tasa de convergencia.

A mi madre María, por su amor cálido e incondicional, por su brillante naturaleza, vitalidad irradiante y fervor de corazón. Por ser el eje primordial de la unión familiar.

A mi padre Jorge, por inculcar el sentido del deber y la perseverancia en el trabajo, por infundir el sentido de mesura. Por inspirar el amor al conocimiento universal y el aprendizaje continuo.

A mi hermano Jordi, por su dignificante visión del mundo y su virtuosa capacidad de pensamiento, por su inteligencia al balancear juicios. Por su apoyo incondicional a lo largo de mi vida y la unión atemporal e infinita que compartimos desde nacimiento.

Al Profesor Paul Rodríguez, por su visión y dedicación al progreso científico, por ayudarme a moldear la capacidad de pensar y presentar un nuevo universo de interpretación matemática de datos, por impulsar a formar un camino profesional en el mundo académico y poder soñar en el futuro.

Índice

1 Principios de optimización y regularización

La investigación desarrollada busca estudiar el algoritmo FISTA (*Fast Iterative Shrinkage-Thresholding Algorithm*) [1], y está orientada a entender las características que conllevan a mejorar la velocidad de convergencia.

En el Capítulo 1 se desarrollan nociones generales necesarias para comprender los fundamentos y alcances del presente estudio, así como la justificación y objetivos.

1.1 Descripción y Motivación

FISTA es un algoritmo de optimización que busca la minimización de la suma de dos funciones convexas. Se evaluarán estrategias de optimización para FISTA con el fin de generar un algoritmo con el mejor desempeño comparativo.

Debido a que FISTA es la solución a un problema lineal inverso, es imprescindible comprender la configuración de ellos.

1.1.1 Problema lineal inverso

Los problemas lineales inversos surgen en una gran variedad de ramas de la ciencia e ingeniería en las que se desea estimar una señal verdadera desconocida, en base a valores conocidos afectados por ruido.

Se definen los valores observados como aquellos a los que se tiene acceso, son datos modificados por ruido, el cual se puede interpretar como consecuencia del medio de adquisición de los datos originales.

En la Figura 1 se puede apreciar una imagen que sufre un proceso de adición de ruido, este se podría deber a errores de sensores, ruido en el canal, etc. El objetivo es recuperar la imagen original, a la cual no se tiene acceso.

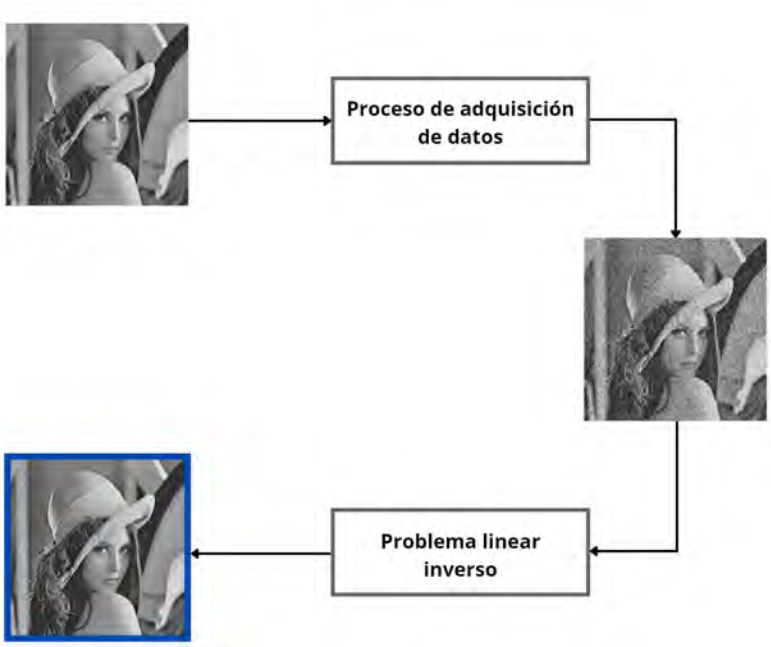


Figura 1 Imagen restaurada en base a datos ruidosos.

La información puede ser representada mediante un conjunto de valores discretos organizados vectorialmente, en este sentido, la naturaleza de la información puede ser variante, tanto una imagen como diversas clases de datos.

Si la información es procesada, el tamaño de datos influye en el tiempo de cómputo, por lo cual son necesarios algoritmos que estimen características importantes o *features* de las señales con fin de mejorar la eficiencia. Esto es posible a través de algoritmos de optimización, tal como el caso de minimización de funciones.

1.1.2 Función de minimización

La solución para un problema lineal inverso puede ser expresada mediante una función de minimización en la que se busca minimizar el tamaño y error de estimación de los datos originales. En particular, en el ámbito de la presente tesis, el objetivo es minimizar una función convexa conformada por la suma de dos funciones:

$$\min_{\mathbf{x}} F(\mathbf{x}) = \min_{\mathbf{x}} f(\mathbf{x}) + g(\mathbf{x}) \tag{1}$$

El primer término $f(\mathbf{x})$ representa la fidelidad o error, el cual garantiza que los valores estimados sean lo más cercano posible a los valores originales. Esta función de error es comprendida por el cuadrado de la norma ℓ_2 de la diferencia entre el vector de datos observados $\mathbf{b}$ y los datos estimados $\mathbf{x}$ convolucionados o transformados a través una matriz A, lo que representa la distancia entre los datos estimados y originales:

$$f(\mathbf{x}) = ||\mathbf{A}\mathbf{x} - \mathbf{b}||_2^2 \tag{2}$$

Se define la norma ℓ_2 de un vector ($||\cdot||_2$) como:

$$||\mathbf{x}||_2 = \sqrt{\left(\sum_i {x_i}^2\right)} = \sqrt{{x_1}^2 + {x_2}^2 + ... + {x_i}^2} \tag{3}$$

Por otro lado, la función de regularización $g(\mathbf{x})$ proporciona información previamente conocida acerca de la naturaleza de los datos para solucionar problemas inversos mal definidos. Particularmente en el caso de estudio, es conformada por la norma ℓ_1 de los datos estimados y busca limitar el tamaño de los datos que se están estimando. Además, el término de regularización λ indica la magnitud del efecto, al ser mayor, se induce un mayor llenado de ceros, con lo que los datos finales son descritos por pocos valores importantes y muchos en cero, esto es denominado *sparse*:

$$g(\mathbf{x}) = \lambda \cdot ||\mathbf{x}||_1 \tag{4}$$

Se define la norma ℓ_1 de un vector ($||\cdot||_1$) como:

$$||\mathbf{x}||_1 = \sum_i |x_i| = |x_1| + |x_2| + ... + |x_i| \tag{5}$$

Ambos términos conforman la función de costo $F(\mathbf{x})$, la cual busca minimizar tanto el error como el tamaño. La minimización toma la siguiente estructura:

$$\min_{\mathbf{x}} F(\mathbf{x}) = ||\mathbf{A}\mathbf{x} - \mathbf{b}||_2^2 + \lambda||\mathbf{x}||_1 \tag{6}$$

Ambas son funciones convexas, por lo que se planea utilizar la gradiente para la minimización. La función de error es convexa suave con la gradiente definida, sin embargo, el segundo término es no suave, lo cual implica un subdiferencial, o un rango de valores en los cuales está definida la gradiente.

Es necesario generar datos estimados que sean una reconstrucción adecuada de la información original, es decir sin ruido y con la menor cantidad de valores. Para lograrlo, es posible el empleo de diversos algoritmos de solución.

1.1.3 Algoritmos de solución

En muchas ocasiones pequeños cambios en la información pueden llevar a resultados muy diversos y alejados de los datos originales, por ello se requieren aproximaciones adecuadas para la resolución del problema, es decir, métodos de regularización eficientes.

Existen diferentes soluciones para el formato particular de la minimización de la suma del error y el tamaño de los valores estimados en base a la norma ℓ_1. Se empleará un método de primer orden, el cual se compone de operaciones simples de cálculo de la gradiente y multiplicaciones de matriz por vector. Los métodos de primer orden son atractivos debido a la poca cantidad de recursos que necesitan.

Es imprescindible generar algoritmos que requieran pocos recursos computacionales, debido a que las bases de datos han experimentado un incremento exponencial en los últimos años [2]. Debido a que FISTA es un algoritmo de primer orden, puede ser utilizado para resolver problemas incluso con datos de gran tamaño.

Además, se emplearán estrategias de optimización con el objetivo de acelerar la tasa de convergencia. Debido a que las funciones a minimizar son convexas, es posible emplear información de la gradiente para la optimización a través de un tamaño de paso actualizable y una reducción inicial del tamaño de los datos en el algoritmo iterativo por medio de métodos de optimización.

1.2 Estado del Arte

1.2.1 Antecedentes

FISTA es formado a partir de un conjunto de algoritmos predecesores.

Gradiente descendiente [3]:

En primer lugar, el método de gradiente descendiente, emplea la gradiente para minimizar la función convexa del error. Se toman pasos en cada iteración en la dirección opuesta a la gradiente, el cual es el paso más descendente con el que se busca llegar al mínimo.

Tikhonov [4]:

El método anterior puede ser efectivo en casos de deconvolución, sin embargo, en otras ocasiones puede llegar a incrementar el nivel de ruido. Tikhonov introduce un término de regularización que busca solucionar este problema: la norma ℓ_2 de los datos estimados (similar a FISTA que emplea la norma ℓ_1), con lo que se llega a obtener una solución única.

ISTA [1]:

El algoritmo ISTA es una extensión del algoritmo de gradiente descendiente, busca la solución de la minimización de la suma del error cuadrático y la función de regularización basada en la norma ℓ_1. Es atractivo debido a su simplicidad ya que utiliza información de la gradiente y valores estimados de una iteración anterior. Se puede emplear incluso para resolver problemas con datos de gran tamaño, sin embargo, este método converge con bastante lentitud, a un ratio de $F(\mathbf{x}_k) - F(\mathbf{x}^*) \simeq \mathcal{O}(1/k)$, donde k es el número de la iteración y $\mathbf{x}^*$es el valor estimado al cual converge la función.

1.2.2 Métodos modernos

FISTA [1]:

Es el algoritmo estudiado en el presente texto, se basa en el algoritmo ISTA y emplea una lógica fundamentada en el algoritmo de gradiente descendiente. Además de utilizar la gradiente para realizar los cálculos, emplea una combinación especifica de valores de dos iteraciones pasadas. Es iterativamente más rápido y conserva la simplicidad computacional de ISTA, con una tasa de convergencia acelerada de $F(\mathbf{x}_k) - F(\mathbf{x}^*) \simeq \mathcal{O}(1/k^2)$.

ADMM [5]:

En este algoritmo se emplean multiplicadores de Lagrange y condicionales para la solución de problemas de optimización como el estudiado. Emplea más pasos que FISTA pero el tiempo de convergencia y costo computacional es similar.

Forward-Backward Splitting [6]:

Resuelve problemas de optimización convexos dividiéndolos en partes más pequeñas, generando secciones reducidas que son sencillas de manipular. Se producen soluciones de pequeños subproblemas locales para encontrar solución a un gran problema global.

En los métodos descritos anteriormente, puede tomar tiempo generar la división matricial [7], y obtener los parámetros de ajuste [8]. FISTA ha ganado popularidad debido a su buena *performance* y poco costo computacional, tiene buen rendimiento incluso en el peor de los casos [8].

1.2.3 Estrategias de optimización

Tamaño de paso actualizable [9]**:**

Se investigan diferentes propuestas de tamaño de paso que modifican el efecto de la gradiente. Si los tamaños de paso son seleccionados manualmente y son

invariantes en cada iteración, puede resultar en un sub- o sobredimensionamiento del efecto de la gradiente.

Los tamaños de paso presentados son actualizados en cada iteración y utilizan información contenida en la gradiente y en iteraciones previas. Se estudian diversos métodos basados en el tamaño de paso de Cauchy, los cuales son sencillos de calcular y garantizan una mejor tasa de convergencia.

Warm Start [10]**:**

Además, existen técnicas para optimizar el algoritmo que buscan una mejor solución inicial al problema. *Warm Start* consiste en el incremento del término de regularización λ, que representa la magnitud de la reducción del tamaño. Se busca un incremento de λ en las primeras iteraciones, tendiendo hacia el valor propuesto inicialmente, lo cual tiene el efecto de generar datos iniciales estimados más *sparse*, con lo que se puede diferenciar valores importantes de irrelevantes.

Screening [11]**:**

Screening elimina entradas de una solución inicial, solo se eliminan valores que son garantizados estar ausentes una vez alcanzada la convergencia en el algoritmo de optimización. Se identifican de forma segura las variables que no determinan características o *features* de la solución óptima.

Screening es un método de optimización con el cual se genera una solución inicial mediante la selección de ciertas columnas de $\mathbf{A}$ y valores estimados al ejecutar el algoritmo iterativo. La complejidad del cálculo de *Screening* es insignificante en comparación con el esfuerzo computacional de ejecutar un algoritmo con mayor número de variables, el costo de *Screening* crece linealmente con el número de *features* [12].

La técnica de *Screening* puede ser aplicada en problemas de optimización convexa suave y no suave, tal como la función de error cuadrático y el caso de la regularización ℓ_1.

1.2.4 Aplicaciones

El algoritmo de optimización FISTA tiene amplias aplicaciones, entre las cuales se describirán los algoritmos de *Total Variation* y *Compressed Sensing*.

Total Variation (TV) [13]**:**

El algoritmo de *Total Variation* (TV) busca suavizar el ruido en regiones planas, así como conservar cambios bruscos en altas frecuencias, los cuales pueden ser interpretados como bordes en una imagen. Con esto, se logra reducir la variación total de los valores estimados, disminuyendo los niveles de error y tamaño.

Para desarrollar el algoritmo de TV, se puede emplear FISTA. Este algoritmo, al igual que FISTA, es computacionalmente poco costoso y tiene una tasa de convergencia acelerada.

Se pueden apreciar sus efectos para *denoising* en la Figura 2.

Figura 2 *Denoising* con *Total Variation*.

Como se observa, se reduce el ruido en regiones planas y además mantiene los bordes, por lo que es aplicado para problemas de *denoising* y *deblurring*.

Compressed Sensing [14]**:**

Para casos más generalizados y señales en el tiempo, en los que además se tenga una baja tasa de muestreo, existe el método denominado *Compressed Sensing*. Los valores originales son recreados en base a los datos extraídos de un diccionario, el cual contiene información de las características de los datos.

Ofrece nuevas posibilidades debido a que puede obtener una señal o imagen con menos muestras que los métodos tradicionales, muestreada incluso por debajo del

doble de la frecuencia máxima, incumpliendo el teorema de Nyquist. Los valores ausentes pueden ser estimados con baja norma y poco error.

La aproximación de los valores faltantes se extrae de mediciones disponibles resolviendo un problema lineal inverso basado en la norma ℓ_1 y es posible desarrollarlo mediante FISTA. La regla de reconstrucción usa optimización convexa y no requiere excesivos recursos computacionales.

Esta técnica posee aplicaciones tales como procesamiento de datos de sensores con tasa de muestreo bajas, recuperación de información faltante en imágenes, corregir errores del medio para señales en el tiempo, etc. Es posible su implementación en señales de cualquier tipo de sensor e información general en las que se tengan relativamente pocas mediciones, y se desee obtener características y detalles ausentes.

1.3 Justificación

Los problemas lineales inversos existen en numerosas ramas de la ciencia e ingeniería y, debido a que el tamaño de los datos se encuentra en constante incremento [2], se genera la necesidad de definir algoritmos de optimización eficientes que requieran poco costo computacional y permitan obtener resultados adecuados, lo más similares posible a los datos originales y con tamaño reducido. Esto es particularmente importante debido a que se dificulta la manipulación de señales de gran tamaño.

FISTA es un algoritmo con buen rendimiento y que requiere poco costo computacional [8]. Mediante la optimización del algoritmo FISTA, se produciría una disminución en el uso de recursos y tiempo de ejecución para los programas que procesen los datos.

1.4 Objetivos

Objetivo general: Evaluación de estrategias de optimización para FISTA con el fin de obtener un algoritmo con el mejor desempeño comparativo.

Objetivos específicos:

- Evaluación de tamaños de paso dinámicos para reducir el error de la gradiente.
- Empleo de *Screening* y *Warm Start* para una mejor solución inicial.
- Desarrollo de algoritmos para comprobar la eficacia de los métodos, por medio de datos aleatorios e imágenes.

Observaciones iniciales

Como se puede apreciar, FISTA es un algoritmo de optimización que soluciona un problema lineal inverso al minimizar el error y promover la reducción del tamaño debido a la penalización de la norma ℓ_1. Además, se pueden aplicar diferentes estrategias de optimización, tales como tamaño de paso automático, *Warm Start* y *Screening*. Por último, FISTA puede ser implementado para diversos tipos de datos en $\mathbb{R}^N$, tales como datos aleatorios e imágenes.

En los próximos capítulos se explicará y evaluará a fondo lo descrito. En el Capítulo 2, se brindan las herramientas matemáticas para entender los algoritmos de minimización iterativos de primer orden. En el Capítulo 3, se demuestra cómo se emplean los métodos de optimización y se implementa un algoritmo generalizado. Finalmente, en el Capítulo 4, se presentan los resultados obtenidos a través de comparadores como una función de costo basada en el error y la norma ℓ_1 de los valores estimados (tamaño), con lo que se generarán conclusiones pertinentes en base a los resultados por medio de simulaciones con el fin de obtener la mejor tasa de convergencia.

2 Marco teórico y fundamentos matemáticos

El presente capítulo tiene como objetivo introducir los conceptos matemáticos necesarios para entender las herramientas de optimización empleadas en el desarrollo de la tesis.

2.1 Conceptos matemáticos para la interpretación y optimización de FISTA

FISTA [1] es la solución a un problema lineal inverso particular. Por lo tanto, es necesario entender la naturaleza de esta clase de enunciados.

2.1.1 Problema lineal inverso

Los problemas lineales inversos se dan en ocasiones en las que, a partir de un conjunto de observaciones o medidas, se debe obtener una aproximación de la información original.

El modelo lineal inverso toma la siguiente estructura:

$$\mathbf{b} = \mathbf{A} \cdot \mathbf{x} \tag{7}$$

La solución tentativa al problema se da con resolver un conjunto de ecuaciones lineales a través de la inversión de la matriz $\mathbf{A}$. Sin embargo, debido a que la matriz $\mathbf{A}$ suele ser de gran tamaño y, en ocasiones es cercana a no ser invertible (*ill-conditioned problem*), se evita este procedimiento.

Por otro lado, los resultados pueden alejarse mucho de los valores originales, es decir, un mismo valor observado puede dar como solución múltiples aproximaciones de datos estimados (*ill-posed problem).* Además, se tiene el caso en el cual es introducido ruido:

$$\mathbf{b} = \mathbf{A} \cdot \mathbf{x} + \eta \tag{8}$$

Con lo que la solución al problema se complica aún más al no poder emplear deconvolución directamente. Por estos motivos, se requieren métodos de regularización para reemplazar el problema original mal condicionado por un

problema bien condicionado y norma delimitada.

2.1.2 Función de minimización

Para plantear una solución adecuada a problemas lineales inversos, se emplea la estrategia de la minimización de funciones convexas, la cual emplea información de la gradiente para hallar un óptimo.

Se plantea la minimización de la función convexa de costo $F(\mathbf{x})$ (Ec. 6), que reduce el error tanto como el tamaño. Esta minimización de suma de funciones convexas se denomina minimización regularizada ℓ_1, siendo posible definir un modelo de solución en base a métodos de optimización de primer orden.

Existen diferentes algoritmos de solución para el problema de minimización de funciones convexas, tales como FISTA.

2.1.3 Operador proximal

Para entender el algoritmo de solución, se definirá primero el método del operador proximal, para luego desarrollar el gradiente proximal [15], los cuales conforman las bases de FISTA.

Partimos del espacio Euclidiano en el que $\forall \mathbf{x} \in \mathbb{E}$. Se define el operador proximal para el caso de minimización con la norma ℓ_1, donde $\mathbf{u}$ es una transformada o función que depende de $\mathbf{x}$:

$$\underset{f}{\text{prox}}(\mathbf{x}) = \underset{\mathbf{u}\in E}{\text{argmin}} \frac{1}{2}||\mathbf{u}-\mathbf{x}||_2^2 + \lambda||\mathbf{u}||_1 \tag{9}$$

Con lo que se define la función convexa:

$$h(\mathbf{u}) = \frac{1}{2}(\mathbf{u}-\mathbf{x})^2 + \lambda|\mathbf{u}| \tag{10}$$

Para hallar el óptimo, se deriva igualando a cero: $h'(\mathbf{u}) = 0$. Se puede demostrar que la solución obtenida es la siguiente [16]:

$$\underset{f}{\text{prox}}(\mathbf{x}) = \begin{cases} 0 & |x| \leq \lambda \\ sign(x)\cdot(|x|-\lambda) & |x| > \lambda \end{cases} = shrink(\mathbf{x},\lambda) \tag{11}$$

Como se aprecia, la solución al sistema se da empleando el operador *shrink*, el cual es un simple comparador de tamaño.

2.1.4 Gradiente proximal

En base al método de gradiente descendiente, los valores de cada iteración dependen de la información de una iteración pasada y de la gradiente [15]:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \nabla f(\mathbf{x}_k) \tag{12}$$

Este caso no considera la minimización de la norma ℓ_1 de los valores estimados, por lo que se debe aplicar el operador proximal, obteniendo el gradiente proximal. Los valores de cada iteración estarán definidos por:

$$\mathbf{x}_{k+1} = prox_g(\mathbf{x}_k - \alpha_k \nabla f(\mathbf{x}_k)) \tag{13}$$

Con lo cual, se tiene una solución sencilla y computacionalmente fácil de calcular para los valores de cada iteración, basados tanto en información de la gradiente como del tamaño.

El gradiente proximal también contiene al operador *shrink* (ver Ec. 11), con lo que se induce el llenado de ceros por medio de una comparación con información de la gradiente.

2.1.5 FISTA

FISTA [1] emplea el gradiente proximal para el cálculo de los valores estimados en cada iteración:

$$\mathbf{x}_k = prox_g(\mathbf{y}_k - \alpha_k \cdot \nabla f(\mathbf{y}_k)) \tag{14}$$

La variable $\mathbf{y}$ representa la actualización de los datos estimados y está comprendida por información de dos iteraciones anteriores. Además, la variable t_k representa un peso que se aplica a la actualización de cada iteración. Las variables son inicializadas como $\mathbf{y}_1 = x_0$ y $t_1 = 1$:

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2} \tag{15}$$

$$\mathbf{y}_{k+1} = \mathbf{x}_k + (\frac{t_k - 1}{t_{k+1}}) \cdot (\mathbf{x}_k - \mathbf{x}_{k-1}) \tag{16}$$

La introducción de dos iteraciones anteriores al gradiente proximal es conocida como gradiente proximal rápido, el cual tiene una tasa de convergencia acelerada.

2.2 Modelos a evaluar

A continuación, se desarrollan modelos mediante los cuales es posible la aplicación del algoritmo FISTA.

Partiendo del caso de la minimización de suma de funciones convexas de error cuadrático y tamaño (ver Ec. 6), se tiene lo siguiente: con el fin de aproximar los datos originales $\mathbf{x}_{orig}$, se tiene la variable $\mathbf{x}$, que es la información estimada en cada iteración. El vector $\mathbf{b}$ representa los datos observados (información a optimizar disponible) y la matriz $\mathbf{A}$, un diccionario o transformación.

Diferentes modelos tienen diversas interpretaciones de los tipos de datos a evaluar.

2.2.1 Modelo de datos aleatorios

Para el caso de la generación sintética de datos aleatorios, la información original a recuperar $\mathbf{x}_{orig}$, es un vector de números aleatorios *sparse*.

La información observada $\mathbf{b}$ es la convolución entre los datos originales y una matriz aleatoria positiva definida $\mathbf{A}$ [17]. Esto asegura que la solución a la función de minimización convexa llegue a un mínimo global, ya que la matriz Hessiana de la función es garantizada ser positiva. Además, se le es añadido ruido aditivo Gaussiano.

Los datos estimados $\mathbf{x}$ pueden ser inicializados en aleatorios, ceros o igualándolos a los datos observados $\mathbf{b}$.

Una interpretación al problema, es la reconstrucción de datos aleatorios por medio de la combinación lineal de un diccionario aleatorio.

2.2.2 Modelo de imágenes mediante la Transformada Wavelet

Para poder describir imágenes por medio de una base ortonormal, se empleará la Transformada Wavelet [18].

En general, las imágenes tienes regiones planas interrumpidas por bordes o cambios bruscos, los cuales son componentes en alta frecuencia, sección de interés para ser evaluada en ciclos de regularización. En el modelo de imágenes, la Transformada Wavelet es empleada para separar los componentes de alta y baja frecuencia mediante una serie de filtros paso altos y paso bajos a través de niveles.

El valor observado $\mathbf{b}$ es la imagen original $\mathbf{x}_{orig}$ a la cual se le ha adherido ruido Gaussiano.

Por otro lado, el vector de coeficientes estimados $c_{\mathbf{x}}$ está dado por la Transformada Wavelet de los datos estimados. Puede ser inicializado en ceros, números aleatorios o igualándolo a la Transformada Wavelet de los datos observados $\mathbf{b}$.

Se busca mantener fijos los coeficientes aproximados (componentes de baja frecuencia) y modificar mediante FISTA los coeficientes de detalle (componentes de altas frecuencias), en los que se encuentra el ruido.

3 Evaluación de estrategias de optimización y desarrollo del algoritmo

Habiendo descrito los fundamentos matemáticos de funciones de minimización de primer orden en capítulos anteriores, es posible generar la lógica de métodos de optimización y desarrollar un algoritmo generalizado con el cual puedan ser evaluados.

3.1 Estrategias de optimización

Existen métodos con diversos enfoques para la optimización de FISTA, los cuales permiten mejorar la velocidad de convergencia.

Debido a que la función objetivo es convexa, es empleada información de la gradiente y de un tamaño de paso actualizable para calcular la dirección de minimización.

Adicionalmente, al ser los datos evaluados *sparse*, se pueden identificar los valores que están asegurados ser ceros finalizando el algoritmo, por medio de una serie de comparaciones.

Así mismo, se puede incrementar el parámetro de regularización con el fin de ampliar el umbral de evaluación. Con esto, en menor número de iteraciones se logra diferenciar valores importantes o *features* de valores prescindibles, los cuales al ser eliminados contribuyen al decremento del error y el tamaño.

3.1.1 Tamaño de paso automático

Al trabajar con una función de minimización convexa, es posible emplear información de la gradiente con el fin lograr una reducción más eficiente de la función de costo en cada iteración.

Partimos del método gradiente descendiente (ver Ec. 12), la actualización de cada iteración lleva dirección opuesta a la gradiente de la función $f(\mathbf{x})$, lo que permite minimizar los datos. Diferentes interpretaciones del algoritmo de gradiente descendiente conducen a diversas opciones de tamaños de paso.

Tamaño de paso de Cauchy

Cauchy [19] propone la solución al siguiente problema:

$$\alpha_k = \underset{\alpha}{\operatorname{argmin}} f(\mathbf{x}_k - \alpha \nabla f(\mathbf{x}_k)) \tag{17}$$

Como se observa, se busca minimizar el tamaño de paso con respecto del algoritmo gradiente descendiente. Con esto, se obtiene el descenso más empinado o *Steepest Descent*, donde $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$:

$$\alpha_k^C = \frac{||\mathbf{g}_k||_2^2}{\mathbf{g}_k^T \mathbf{A} \mathbf{g}_k} \tag{18}$$

El tamaño de paso es elegido evaluando la mejor dirección, es decir el mínimo de la función en α. En la Figura 3, se puede observar que FISTA, con tamaño de paso de Cauchy, converge más rápidamente que en el caso de tamaño de paso constante.

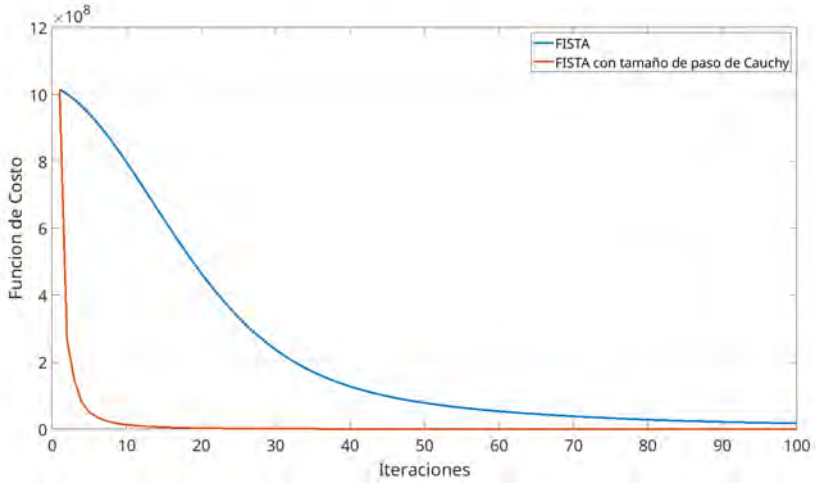


Figura 3 Comparación entre FISTA Regular (azul) y con tamaño de paso de Cauchy (rojo).

Sin embargo, en ocasiones, resulta ser un método que converge lentamente. Para dos dimensiones, las direcciones se encuentran en un subespacio bidimensional constituido por dos vectores propios de $\mathbf{A}$ [20]. Por lo tanto, el método de descenso más empinado converge solo de forma lineal y puede ser muy lento.

Debido a esto, se emplean métodos que busquen diferentes estrategias.

Tamaño de paso de Barzilai y Borwein

La lógica del enfoque de Barzilai y Borwein [21] es emplear información de la iteración anterior para decidir el tamaño del paso en la iteración actual. Se plantea:

$$\alpha_k = \underset{\alpha}{\operatorname{argmin}} ||(\Delta \mathbf{x} - \alpha \Delta \mathbf{g})||_2^2 \tag{19}$$

Donde $\Delta \mathbf{x} = \mathbf{x}_k - \mathbf{x}_{k-1}$ y $\Delta \mathbf{g} = \mathbf{g}_k - \mathbf{g}_{k-1}$. En este caso, se evalúa tanto la diferencia de la gradiente como la de los valores estimados. Con lo que se obtienen los siguientes dos tamaños de paso:

$$\alpha_k^{BB1} = \frac{\Delta \mathbf{x}^T \Delta \mathbf{g}}{||\Delta \mathbf{g}||_2^2} \tag{20}$$

$$\alpha_k^{BB2} = \frac{||\Delta \mathbf{x}||_2^2}{\Delta \mathbf{x}^T \Delta \mathbf{g}} \tag{21}$$

El método de Barzilai y Borwein para tamaños de paso α_k^{BB1} y α_k^{BB2} produce una convergencia superlineal [22] si la función $f(x)$ es una función cuadrática convexa de dos variables. En estos casos, las direcciones generadas Barzilai y Borwein pueden encontrarse en todo el subespacio de $\mathbf{A}$, a diferencia del tamaño de paso de Cauchy que solo toma dos direcciones.

En la Figura 4, se puede observar el comportamiento de los tamaños de paso de Barzilai y Borwein.

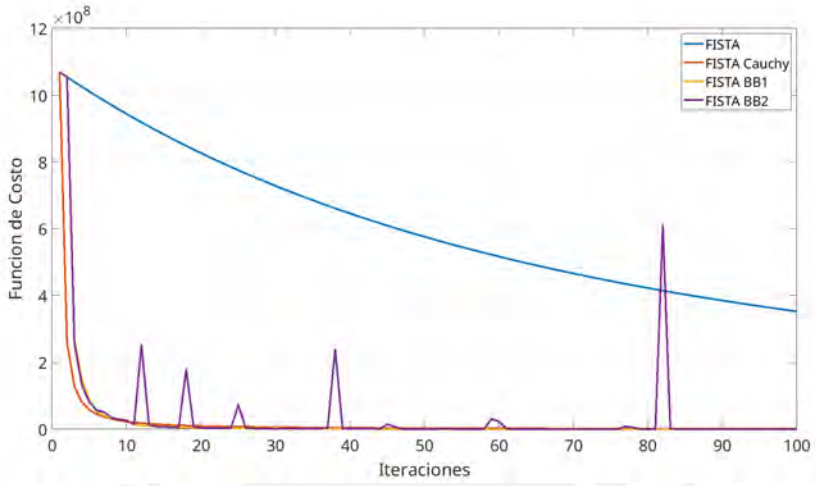


Figura 4 Comparación entre FISTA Regular (azul), tamaño de paso de Cauchy (rojo) y Barzilai y Borwein (amarillo y morado).

Para el caso del segundo tamaño de paso de Barzilai y Borwein α_k^{BB2}, se observan picos en la función de costo. La gradiente puede llegar a tomar valores muy bajos y, al encontrarse en el denominador, puede generar tamaños de paso muy elevados.

Se busca, además, un método que tenga más de una elección de tamaño de paso.

Tamaño de paso de Yuan

Partiendo de la idea que la ineficiencia del tamaño de paso de Cauchy sea debido a que se está empleando un criterio constante para cada iteración, Yuan propone ciclos de intercambio entre dos diferentes tamaños de paso:

$$\alpha_k^{YA} = \begin{cases} \alpha_k^C & k : par \\ \alpha_k^Y & k : impar \end{cases} \tag{22}$$

$$\alpha_k^Y = \frac{2}{\sqrt{(1/\alpha_{k-1}^C + 1/\alpha_k^C)^2 + 4||\mathbf{g}_k||_2^2/||\Delta\mathbf{x}||_2^2} + 1/\alpha_{k-1}^C + 1/\alpha_k^C} \tag{23}$$

Se tiene un tamaño de paso que cambia en la medida de ser una iteración par (paso de Cauchy) o impar (paso de Yuan). Con lo que se evita que las direcciones

de minimización sean repetitivas, como en el caso del tamaño de paso de Cauchy. Se puede apreciar su efectividad en la Figura 5.

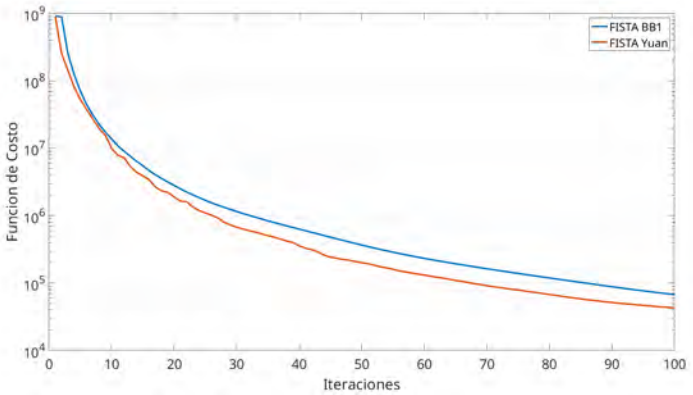


Figura 5 Comparación entre FISTA con tamaño de paso de Barlizali y Borwein 1 (azul) y Yuan (rojo).

Por otro lado, puede ocurrir en ocasiones que la función de costo llegue a divergir con un tamaño de paso automático, como se observa en la Figura 6.

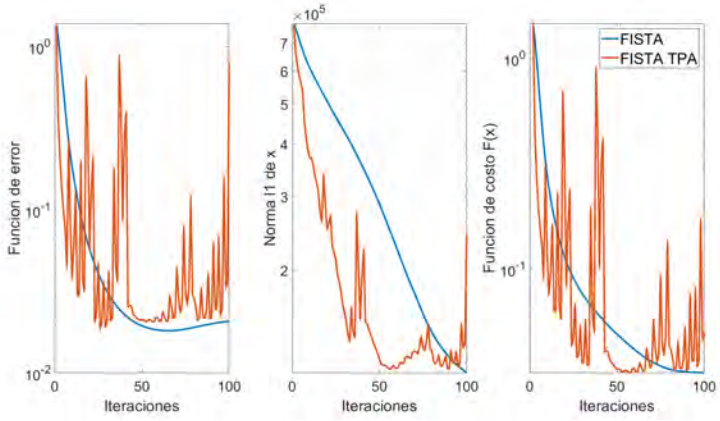


Figura 6 Comparación entre FISTA Regular (azul) y tamaño de paso automático que diverge (rojo).

En casos que la gradiente llegue a tomar valores muy reducidos, al ser la gradiente parte del denominador de los tamaños de paso (presente tanto en Cauchy, Barzilai y Borwein, y Yuan), puede llegar a generar tamaños de paso muy elevados que produzcan picos en la función de costo.

Se pueden observar los efectos del tamaño de paso en la función de costo en la Figura 7.

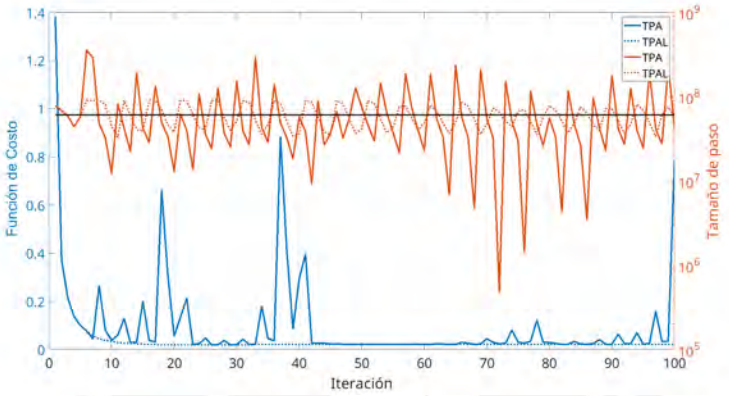


Figura 7 Comparación de funciones de costo y tamaños de paso entre FISTA con TPA (tamaño de paso automático) y TPAL (tamaño de paso automático con límites).

Además, se tiene una comparación con el efecto de la gradiente, la cual se puede apreciar en la Figura 8.

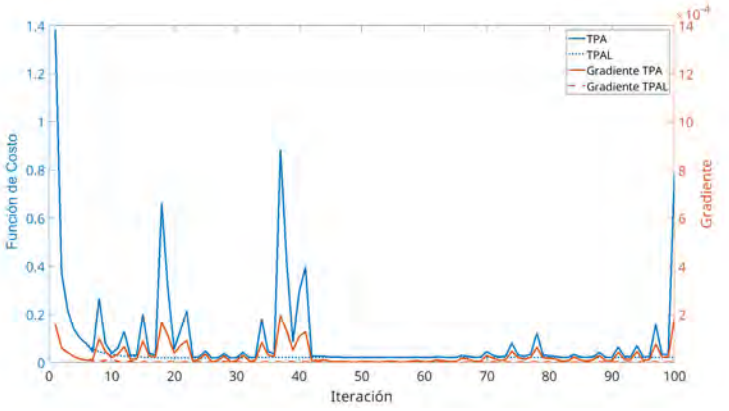


Figura 8 Comparación de funciones de costo y gradiente entre FISTA con TPA (tamaño de paso automático) y TPAL (tamaño de paso automático con límites).

Al delimitar los tamaños de paso se evita que un tamaño muy elevado incremente la distancia entre los datos originales y estimados, y por ende el error.

Con esto, es posible obtener una gradiente pequeña que tienda a decrecer y tamaños de paso automáticos que no crezcan desmesuradamente.

Adicionalmente, es posible multiplicar el tamaño de paso adaptativo por un factor con el objetivo de incrementar o disminuir su efecto.

Además del tamaño de paso, es posible modificar el parámetro de regularización λ, el cual está relacionado con el tamaño de la umbralización realizada en el *soft-thresholding* de cada iteración.

3.1.2 Warm Start

Partiendo del enfoque que los datos estimados son de naturaleza *sparse*, y que pueden ser representados por un pequeño número de *features*, se aplica el método *Warm Start* [23], con el que es posible obtener una reducción inicial de tamaño.

Para lograrlo, se incrementa el parámetro de regularización λ multiplicándolo por un factor γ_{ws}, lo que tiene como resultado una mayor supresión inicial de datos en el algoritmo iterativo, y con esto, un reducido número de valores evaluados. Se tiene:

$$\lambda_{ws} = \gamma_{ws} \cdot \lambda_0 \tag{24}$$

La Figura 9 busca ilustrar el comportamiento en las primeras iteraciones de *Warm Start*. Se tiene, además, el tamaño de la variable estimada con respecto al error.

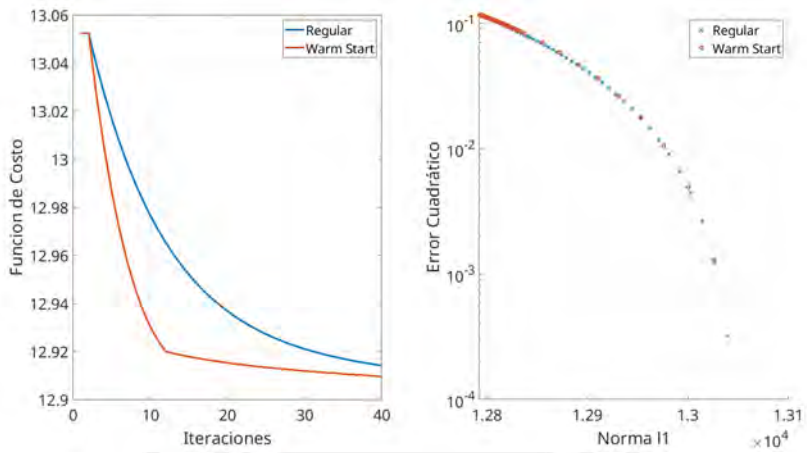


Figura 9 Comparación entre FISTA Regular (azul) y con *Warm Start* (rojo).

Como se aprecia, el algoritmo converge inicialmente más rápido en el caso de *Warm Start*. En el gráfico de la derecha, partiendo de la esquina inferior derecha, puede observarse iteración por iteración, cómo el caso de *Warm Start* (triángulos en rojo) consigue pasos iniciales más separados que para el caso regular (cruces en azul), con lo que disminuye la noma ℓ_1 y la función de costo más rápidamente.

El método de *Warm Start* converge en un punto de acumulación único (mínimo global) al ser aplicado en las primeras iteraciones.

Se emplean diferentes estrategias para incrementar el parámetro de regularización. Para iteraciones iniciales con *Warm Start,* se tiene $\gamma_{ws} > 1$ y, para el resto del algoritmo iterativo, $\gamma_f = 1$.

Valor constante inicial:

Multiplicar el término de regularización λ por un valor constante de γ_{ws} en las primeras iteraciones.

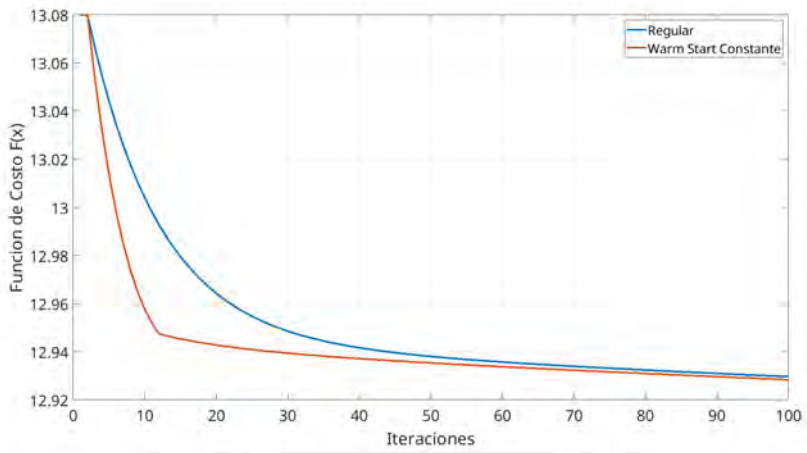


Figura 10 Comparación entre FISTA Regular (azul) y con *Warm Start* (rojo).

En la Figura 10 se aprecia claramente que para las primeras iteraciones el decremento en la función de costo ha sido mayor que para el caso regular. Se evaluarán también casos de decremento lineal y cuadrático.

Decremento lineal y cuadrático

Reducir γ lineal y cuadráticamente en las primeras iteraciones: $\gamma_{ws} \to \gamma_f = 1$.

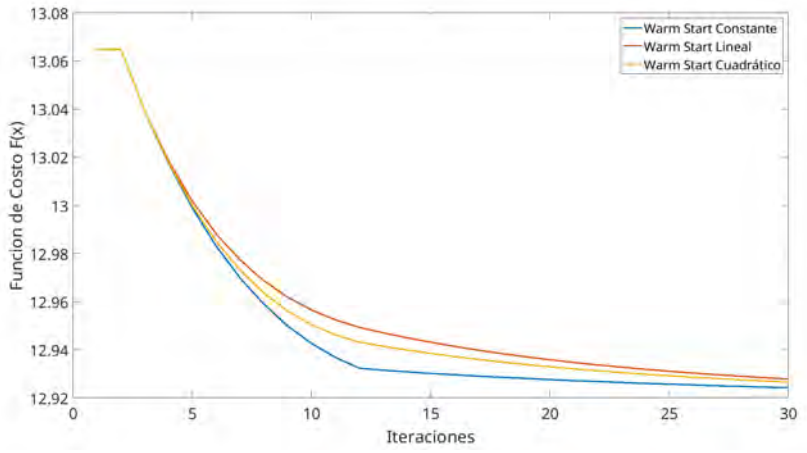


Figura 11 Comparación entre FISTA con *Warm Start* constante (azul), con decremento lineal (rojo) y decremento cuadrático (amarillo).

En la Figura 11 se observa que el mejor método de *Warm Start* es aquel que lleva a γ_{ws} constante en las primeras iteraciones, sin decremento.

Además, el enfoque de decremento cuadrático consigue mejor rendimiento que el método de decremento lineal debido a que lleva valores más elevados en las primeras iteraciones para el término de regularización. Por último, se evaluará mantener constante γ_{ws} para todas las iteraciones del algoritmo.

Warm Start en todas las iteraciones:

Mantener γ_{ws} en todas las iteraciones a ejecutar. Se puede observar los resultados en la Figura 12.

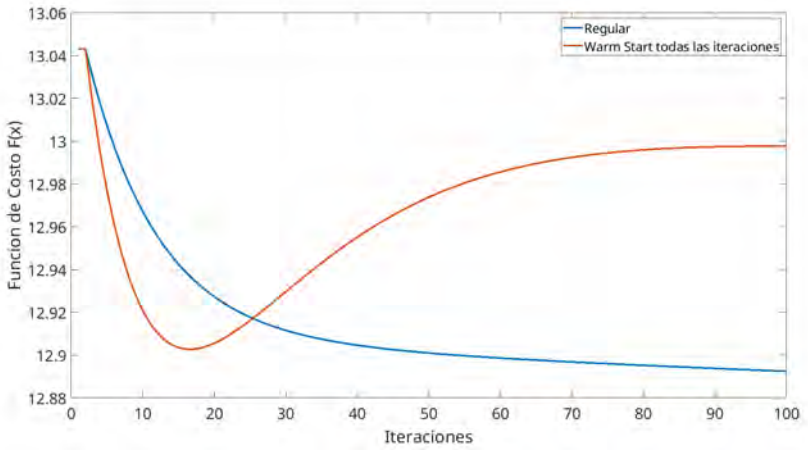


Figura 12 Comparación entre FISTA Regular (azul) y con *Warm Start* en todas las iteraciones (rojo).

Warm Start es empleado con el fin de obtener una representación inicial más *sparse*. Sin embargo, al ser ejecutado en todas las iteraciones, el umbral resulta muy elevado y son eliminadas *features* (o características importantes) con lo que, si bien disminuye el tamaño de los datos estimados, más peso tiene el incremento del error. Por lo que *Warm Start* debe ser aplicado solo a un porcentaje inicial de las iteraciones.

Además, se implementará el método de *Screening*, el cual será ejecutado previo al ciclo iterativo del algoritmo.

3.1.3 Screening

Debido a que la información es *sparse*, pocos valores son importantes y describen características relevantes de los datos originales que se desean recuperar. Es posible discernir entre la información importante y valores de los cuales se puede prescindir por medio de comparaciones. El objetivo de *Screening* es eliminar los valores que están garantizados ser ceros al converger el algoritmo de minimización iterativo.

Screening se basa en geometría de la gradiente y la función dual para delimitar la dirección del subdiferencial de la función $g(\mathbf{x})$ no suave. A partir de esto, se producen reglas de condición para la selección de variables que son seguras de eliminar. Además, la gradiente de la función convexa suave $f(\mathbf{x})$ pertenece a un subdiferencial de la función convexa no suave $g(\mathbf{x})$ calculados en el óptimo. Para valores diferentes de cero, la norma infinita (valor máximo) del gradiente es λ y es empleado para discriminar entre *features* y valores a que se pueden fijar en ceros.

Para tener un criterio de comparación, se emplea la función dual de maximización, que es la conjugada de la función primaria de minimización $F(\mathbf{x})$. Partimos de la función de minimización con regularización ℓ_1(Ec. 6), se obtiene el Lagrangiano de la función primaria, donde $\mathbf{z}$ son los multiplicadores de Lagrange [17]:

$$\mathscr{L}(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \frac{1}{2}||\mathbf{A}\mathbf{x} - \mathbf{b}||_2^2 + \lambda||\mathbf{y}||_1 + \mathbf{z}^T(\mathbf{x} - \mathbf{y}) \tag{25}$$

$$= \max_{\mathbf{z}} \mathbf{z}^T(\mathbf{B}\mathbf{A}^T\mathbf{b}) - \frac{1}{2}\mathbf{z}^T\mathbf{B}\mathbf{z} \tag{26}$$

Con lo que se obtiene la función dual, la cual es una función de maximización cóncava donde $\mathbf{B} = (\mathbf{A}^T\mathbf{A})^{-1}$. Además, se puede representar la función primaria y dual como una función de minimización con restricciones:

$$\min_{\mathbf{z}} \mathbf{z}^T\mathbf{B}\mathbf{z} - \mathbf{z}^T(\mathbf{B}\mathbf{A}^T\mathbf{b}), \quad \forall -\lambda < z_i < \lambda \tag{27}$$

El problema viene dado por una minimización cuadrática, la cual tiene solución hallando el subdiferencial del término $\lambda \cdot ||\mathbf{x}||_1$.

Por medio de la función dual de maximización y la función primaria de minimización, se pueden establecer límites máximos para el parámetro λ. Debido a las condiciones de dualidad del problema, se mantiene una fuerte relación entre la función dual y primaria, con lo que se alcanza los valores óptimos en ambas a la vez. Por medio de la función dual se obtiene un límite máximo para λ, donde $\mathbf{a}_j$ es la columna j-ésima de la matriz $\mathbf{A}$ [12]:

$$\lambda \geq \lambda_{max} := \max_{1 \leq j \leq n} |\mathbf{a}_j \mathbf{b}| = ||\mathbf{A}^T \mathbf{b}||_\infty \tag{28}$$

Con lo que se obtiene una métrica de comparación que permite eliminar componentes que están garantizados ser ceros al finalizar el algoritmo:

$$\lambda > \rho_k \lambda_{max} \tag{29}$$

$$\rho_k = \frac{||\mathbf{b}||_2 ||\mathbf{a}||_2 + |\mathbf{b}^T \mathbf{a}_j|}{||\mathbf{b}||_2 ||\mathbf{a}_j||_2 + \lambda_{max}} \tag{30}$$

La complejidad crece linealmente con el número de características m y valores de evaluación n : $\mathcal{O}(m \cdot n)$. Se obtienen mejores resultados en la medida que los datos evaluados sean más *sparse*. Además, con el fin de mejorar la *performance* computacional, se puede aproximar la norma ℓ_2 de las columnas de la matriz $\mathbf{A}$ por medio de la norma ℓ_1, logrando disminuir el tiempo de ejecución al realizar menos operaciones.

En la Figura 13 pueden apreciarse los efectos del método *Screening* previo a inicializar el algoritmo iterativo.

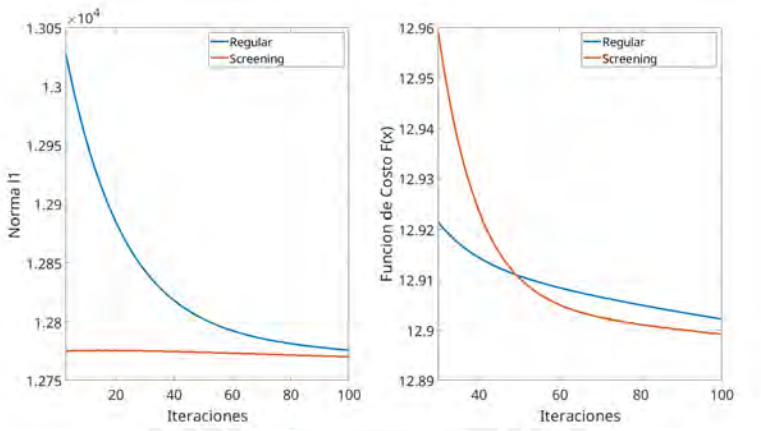


Figura 13 Comparación entre FISTA Regular (azul) y con *Screening* (rojo).

Se observa que el tamaño, representado por la norma ℓ_1 de los datos estimados, disminuye drásticamente. Además, se observa que si bien inicialmente la función de costo es mayor para el caso de *Screening* (incrementa el error por eliminar valores), logra converger más rápidamente que el caso regular. Se observa que finalmente *Screening* consigue menor valor para la función de costo.

La comparación puede modificarse considerando diferentes porcentajes de λ_{max}, con el objetivo de poder controlar la cantidad de valores a eliminar. Se presenta un caso extremo en la Figura 14.

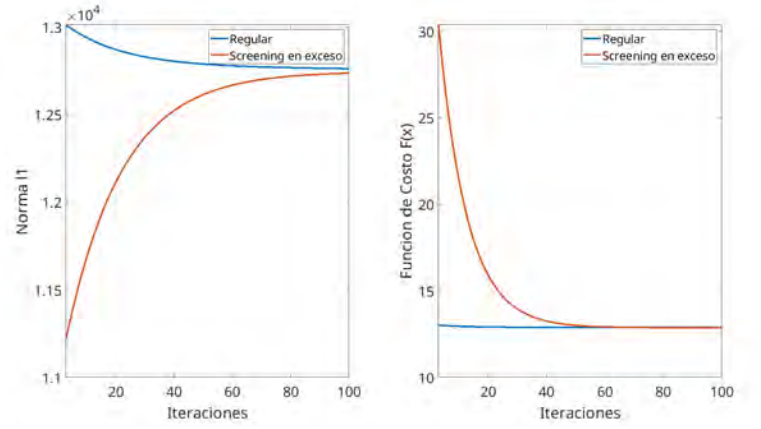


Figura 14 Comparación entre FISTA Regular (azul) y con *Screening* excesivo (rojo).

Si bien la norma ℓ_1 disminuye radicalmente para las primeras iteraciones, tiene un gran crecimiento a lo largo del resto de iteraciones. Se han eliminado rasgos relevantes de información que llevan a un error más elevado que para el caso sin *Screening*. Si bien el algoritmo llega a converger, para este caso, *Screening* con exceso no resulta conveniente. Es imperativo establecer un porcentaje apropiado para el valor de λ_{max}, con el fin de conservar un número adecuado de *features*.

Los métodos de *Warm Start* tienen aún mejores resultados cuando la data es *sparse,* por lo tanto, al implementar *Screening* previo a *Warm Start,* puede mejorar su eficacia.

La siguiente sección comprende la implementación de los métodos de optimización desarrollados en un algoritmo generalizado.

3.2 Código Generalizado

El programa desarrollado es un algoritmo generalizado que puede ser ejecutado para optimizar diversos tipos de datos.

La información a emplear en el presente trabajo consiste en variables sintéticas generadas aleatoriamente e imágenes reconstruidas. Además, el presente trabajo puede ser adaptado para diversas clases de valores en $\mathbb{R}^n$ de naturaleza *sparse*, los cuales pueden ser representados por la combinación lineal de una base ortonormal.

Lleva como centro de la estructura el algoritmo iterativo de primer orden FISTA, el cual será optimizado implementando diferentes tamaños de paso y métodos de *Screening* y *Warm Start*, con el objetivo de obtener una inicialización más eficiente de los datos.

A continuación, se detallará la estructura de las funciones que constituyen el algoritmo desarrollado.

3.2.1 Inicialización de opciones y constantes

Se emplean opciones y constantes para la inicialización de datos y variables del algoritmo, así como la selección de parámetros en los métodos de optimización.

Opciones:

Debido a que es un algoritmo generalizado, se emplean valores para representar la activación de diferentes tamaños de paso, así como de métodos de optimización de *Screening* y *Warm Start*. Además, se pueden elegir diferentes datos a evaluar y modificar sus parámetros.

Opción "Diccionario": se empleará el algoritmo para el caso de valores aleatorios e imágenes representadas mediante la Transformada Wavelet. Este parámetro permite la elección de uno de ellos, con posibilidad de ser adecuado a otros tipos de información *sparse* en $\mathbb{R}^n$.

Opción "Nombre de tamaño de paso": permite la selección del tamaño de paso α a emplear. Pueden ser seleccionados los tamaños de paso de Cauchy, Barzilai y Borwein 1 y 2, Yuan y un tamaño de paso constante.

Opción "Estrategia de Warm Start": elige el tipo de variación del término de regularización en las primeras iteraciones: continuo, lineal y cuadrático.

Opción "Tipo de Transformada Wavelet": seleccionada el tipo de Transformada Wavelet a evaluar. Se evaluarán *wavelets* de la familia Daubechies.

Constantes lógicas:

Se utilizan con el fin de activar los límites para el tamaño de paso y para que se efectúen los bloques de optimización de *Warm Start y Screening* (verdadero o falso).

Parámetros:

Entre los valores a ser especificados previo a inicializar el algoritmo, se encuentran:

- N : dimensionalidad de los valores a evaluar.
- L : número de iteraciones que realizará el algoritmo.
- α : tamaño de paso relacionado a la gradiente, el cual puede ser actualizado en cada iteración.
- λ : parámetro de regularización, define el umbral ligado a la reducción de tamaño.

- σ : delimita el ruido adherido a la información original, con el fin de obtener los datos observados.

- γ : define el incremento de λ en las iteraciones iniciales con *Warm Start*.

- iw : iteración hasta la cual se evaluará *Warm Start*.

- is : intensidad de *Screening*, porcentaje de λ_{max}.

- n : nivel de evaluación de la Transformada Wavelet.

3.2.2 Generación de los valores a evaluar

Recordando la forma general del problema de minimización (Ec. 6), se tiene que los valores a evaluar comprenden los datos originales $\mathbf{x}_{orig}$, una matriz $\mathbf{A}$, los datos observados $\mathbf{b}$ y la variable a estimar $\mathbf{x}$.

Los datos generados pueden ser aleatorios o imágenes mediante la Transformada Wavelet, su selección depende de las opciones elegidas en la sección anterior.

Caso Aleatorio:

- **Datos originales $\mathbf{x}_{orig}$**: Es la información original que se desea recuperar, es un vector aleatorio con distribución Gaussiana y *sparse*. $\mathbf{x}_{orig} \in \mathbb{R}^n$.

- **Matriz A:** es una matriz diagonal aleatoria, positiva definida y normalizada. $\mathbf{A} \in \mathbb{R}^{nxn}$.

- **Datos observados b:** Es el producto de la matriz $\mathbf{A}$ y el vector $\mathbf{x}_{orig}$, lo cual puede ser interpretado como convolución. Además, se le es sumado ruido aditivo gaussiano. Es importante notar que si $\mathbf{b} = \mathbf{A} \cdot \mathbf{x}_{orig}$, la solución al problema sería trivial, sin embargo, el ruido adicionado lo convierte en un problema mal condicionado, requiriendo de un algoritmo con regularización. $\mathbf{b} = \mathbf{A} \cdot \mathbf{x}_{orig} + \sigma$, $\mathbf{b} \in \mathbb{R}^n$.

- **Datos estimados x:** Es el vector de valores estimados, es inicializado con valores aleatorios. $\mathbf{x} \in \mathbb{R}^n$.

Caso Wavelets:

Para el caso de valores con la Transformada Wavelet, se emplean imágenes de dimensiones mxm y con un vector de coeficientes aproximados y detallados con p valores.

- Datos originales $\mathbf{x}_{orig}$: Los datos originales son representados por una imagen sin ruido.$\mathbf{x}_{orig} \in \mathbb{R}^{mxm}$.

- Datos observados b: El valor observado es la imagen original $\mathbf{x}_{orig}$, a la cual se le ha adherido ruido Gaussiano. $\mathbf{b} \in \mathbb{R}^{mxm}$.

- Coeficientes de datos estimados $c_{\mathbf{x}}$: Es el vector de coeficientes de valores estimados, es inicializado con valores aleatorios. . $\mathbf{x} \in \mathbb{R}^p$.

- Máscara de evaluación m: A diferencia del caso que se optimizaban valores aleatorios, al trabajar con la Transformada Wavelet, no todos los datos ingresan al proceso de regularización. La máscara es un vector de ceros en la posición de los coeficientes aproximados (los cuales no serán evaluados) y unos en la posición de los coeficientes detallados (los cuales sí serán regularizados). $\mathbf{m} \in \mathbb{R}^p$.

3.2.3 Reducción de dimensiones: Método Screening

Se ejecutará antes del bloque iterativo, con el fin de obtener una mejor solución inicial. Es posible modificar su efecto mediante el parámetro de intensidad de *Screening*.

El método emplea una comparación en base a los límites geométricos de la función primaria y dual, la cual es detallada en la Ec. 29. Los valores que cumplan con esta condición pueden ser eliminados. El bloque de *Screening* es de activación opcional y es aún más efectivo cuando los datos a evaluar son *sparse*.

3.2.4 Iteraciones

En esta sección se genera la evaluación del algoritmo iterativo de primer orden con un tamaño de paso que se actualiza en cada iteración. Además, las primeras

iteraciones tienen el objetivo de una inicialización más efectiva implementando el método de *Warm Start*.

Tamaños de paso actualizables

El tamaño de paso para la primera iteración es el tamaño de paso de Cauchy, el cual no requiere de iteraciones previas. $\alpha_0 = \alpha_k^C$.

Para el resto de iteraciones los tamaños de paso pueden emplear información de iteraciones previas. Estos pueden ser seleccionados modificando la constante del "nombre de tamaño de paso" en "opciones", con las diferentes alternativas: $\alpha_k = \alpha_k^{BB1}$, α_k^{BB2}, α_k^Y o α_k^{cte}.

Se generan además límites y se multiplica los tamaños de paso por una constante con fin de evitar picos.

Método Warm Start

El método de *Warm Start* puede ser activado al inicio de las primeras iteraciones. Como lo explicado previamente, consiste en incrementar el término de regularización. La magnitud de incremento, el tipo de decremento, así como la cantidad de iteraciones a ejecutarse, pueden ser seleccionados mediante las variables en opciones y parámetros.

FISTA

FISTA es la sección del algoritmo encargada de la reducción del tamaño y el error. Emplea la técnica de gradiente descendiente como argumento del operador proximal, obteniendo el gradiente proximal. Además, emplea valores de iteraciones anteriores obteniendo una convergencia acelerada.

Se emplea una actualización de los valores estimados conforme a la Ec. 16. Adicionalmente, para FISTA, es necesario el cálculo de la gradiente de la función $f(\mathbf{x})$ para cada iteración, por medio de la derivada de Fréchet [24], se tiene:

$$\nabla f(\mathbf{x}) = \mathbf{A}^T(\mathbf{A}\mathbf{x} - \mathbf{b}) \tag{31}$$

Por lo que es necesaria la matriz $\mathbf{A}$ "diccionario" para el cálculo de la gradiente. Se evalúa en cada uno de los dos escenarios:

Caso Aleatorio: Debido a que se tiene acceso a la matriz $\mathbf{A}$, la cual fue generada artificialmente al inicializar las variables, hallar la gradiente requiere simples operaciones de multiplicación de matriz por vector.

Caso Wavelets: En el caso de la Transformada Wavelet, no se tiene acceso a la matriz diccionario. Sin embargo, el proceso de convolución de $\mathbf{A} \cdot \mathbf{x}$, es interpretado como la combinación lineal entre los valores estimados y las frecuencias de las *wavelets*, representando el peso asociado a la frecuencia de las ondas para la reconstrucción de la imagen. Con DWT (Transformada Directa Wavelet) e IWT (Transformada Inversa Wavelet), se tiene:

$$\nabla f(\mathbf{x}) = DWT(IWT(\mathbf{x}) - \mathbf{b}) \tag{32}$$

Con lo cual, finaliza el bloque iterativo y se procede a graficar los resultados.

3.2.5 Valores estadísticos y gráficos

Con el fin de monitorear la evolución de la convergencia, se guardan tres valores para cada iteración: error cuadrático, tamaño de los datos estimados (norma ℓ_1) y la función de costo. Estos resultados son graficados para poder apreciar y evaluar la evolución de las tres funciones en cada iteración.

En adición a los algoritmos mencionados se creó un código general, el cual retorna los valores de estadísticas para diferentes formas de evaluación, es decir, para diferentes formas de inicialización de los datos y métodos, con el fin de compararlos. Este puede ser revisado a más detalle en la sección de Anexos.

Los resultados y comparaciones son desarrollados en el Capítulo 4, donde se modificarán las constantes del algoritmo desarrollado para poder comparar las técnicas de optimización y buscar el algoritmo con la mejor eficiencia.

4 Resultados

Habiendo descrito métodos de optimización para algoritmos de primer orden como FISTA, además de la lógica de las funciones del programa a emplear, se procede a medir los resultados obtenidos.

El algoritmo FISTA será optimizado tomando en consideración los métodos descritos anteriormente. Se empleará un tamaño de paso actualizable en cada iteración, una reducción del tamaño de datos previo a iniciar la estructura iterativa mediante *Screening* y un incremento del término de regularización en las primeras iteraciones por medio de *Warm Start*.

Para la evaluación de convergencia, se emplearán las siguientes tres medidas:

Función de error: $||\mathbf{Ax} - \mathbf{b}||_2^2$

Es una métrica de similitud que mide la distancia entre los datos estimados (convolucionados o mediante la Transformada Wavelet) y los datos observados. Mientras menor sea, mayor será la similitud entre la información estimada y la original.

Norma ℓ_1 de los datos estimados: $||\mathbf{x}||_1$

Debido a que la norma ℓ_1 es la suma de los valores absolutos, esta función representa el tamaño de los datos estimados en cada iteración.

Función de costo: $||\mathbf{Ax} - \mathbf{b}||_2^2 + \lambda||\mathbf{x}||_1$

Es la función convexa a minimizar, es la suma de función de error y la norma ℓ_1 de $\mathbf{x}$ multiplicado por el término de regularización λ. Su minimización representa un balance entre la disminución del tamaño y el error.

El programa es desarrollado en Matlab R2020a en el sistema operativo Manjaro Linux 21.0.7.

Además, se emplea un computador con las siguientes características:

- Procesador Intel Core i3 550 @3.2 GHz, 2 núcleos y 4 hilos.
- Memoria Cache L1: 64 KB, L2: 256 KB, L3: 4096 KB.
- Memoria RAM 8GB DDR3 @1333 MHz.

La configuración del experimento, será clasificada según la naturaleza de los datos a evaluar, los cuales serán de dos clases: información aleatoria generada artificialmente e imágenes, las cuales serán optimizadas empleando la Transformada Wavelet.

Los datos estimados iniciales son generados aleatoriamente en ambos casos y se busca la convergencia no-asintótica en un número finito de iteraciones.

4.1 Caso Aleatorio

Para el primer caso de evaluación, la información generada está compuesta por vectores aleatorios con *sparsity* de 25% y 50%. Se toman los siguientes valores para los parámetros empleados:

Parámetro de regularización $\lambda = 10^{-1}$, tamaño de paso $\alpha = 10^{-1}$ (empleado en todos los casos a excepción de FISTA con tamaño de paso actualizable) y parámetro de error aditivo $\sigma = 10^{-2}$.

El algoritmo logra convergencia si la diferencia entre dos iteraciones de la función de costo es menor a $5 \cdot 10^{-1}$.

4.1.1 ISTA y FISTA

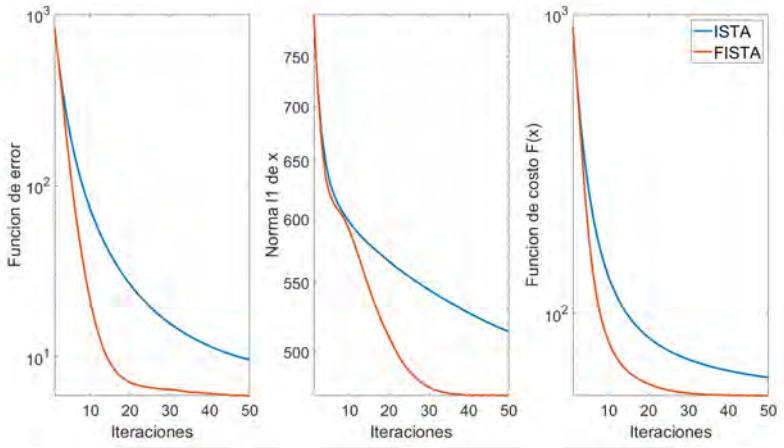
El primer caso a evaluar es la comparación entre los algoritmos ISTA y FISTA, ambos con tamaño de paso constante de $\alpha = 10^{-1}$. Como lo mencionado previamente, FISTA es la versión acelerada de ISTA, empleando información del valor estimado de dos iteraciones previas.

Se grafica la evolución de diversas métricas: (a) la función de error $||\mathbf{Ax} - \mathbf{b}||_2^2$, (b) la función de regularización $||\mathbf{x}||_1$ y (c) la función de costo: (a) + (b). La escala

logarítmica es empleada debido a que las reducciones pueden llegar a ser de varios órdenes de magnitud.

Se puede observar una comparación de las tres métricas estadísticas en la Figura 15. Es considerado un gráfico para la evaluación de los casos de *sparsity* de 25% y 50% debido a que mantienen una estructura similar.

Para mayor detalle, se pueden apreciar los resultados de los dos casos en las Tablas 1 y 2, en las que se muestran valores obtenidos en 50 iteraciones del algoritmo ejecutado 100 veces, hallando así la media (negro) y desviación estándar (azul) para determinadas iteraciones.



(a) **(b)** **(c)**

Figura 15 ISTA y FISTA.

Iter.	Error ISTA	Error FISTA	Norma ℓ_1 ISTA	Norma ℓ_1 FISTA	Costo ISTA	Costo FISTA
1	879.94±44.02	879.94±44.02	798.25±18.01	798.25±18.01	959.77±44.71	959.77±44.71
10	79.64±4.29	23.51±1.38	579.27±17.08	569.84±15.51	137.57±4.86	80.49±2.35
20	29.28±1.67	6.78±0.31	543.36±16.22	486.45±15.32	83.62±2.58	55.43±1.65
30	17.14±1.00	6.26±0.23	518.49±15.80	453.31±15.11	68.99±2.08	51.59±1.55
40	12.42±0.72	6.01±0.25	500.10±15.60	448.90±14.83	62.43±1.89	50.91±1.52
50	10.12±0.57	5.92±0.28	486.37±15.39	448.48±14.90	58.76±1.77	50.77±1.51

Tabla 1 ISTA y FISTA, *sparsity* de 25%. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error ISTA	Error FISTA	Norma ℓ_1 ISTA	Norma ℓ_1 FISTA	Costo ISTA	Costo FISTA
1	746.06±63.99	746.06±63.99	797.73±20.84	797.73±20.84	825.84±65.20	825.84±65.20
10	66.21±3.11	18.96±1.14	535.10±15.12	485.83±17.07	119.72±3.96	67.54±2.33
20	23.47±1.26	5.19±0.24	468.61±16.17	358.25±15.89	70.33±2.38	41.01±1.69
30	13.29±0.77	4.34±0.19	425.93±16.59	325.36±13.92	55.89±2.06	36.87±1.40
40	9.39±0.53	3.75±0.21	395.39±16.63	325.85±13.84	48.93±1.93	36.34±1.36
50	7.52±0.40	3.64±0.19	373.02±16.51	326.01±14.02	44.82±1.84	36.24±1.36

Tabla 2 ISTA y FISTA, *sparsity* de 50%. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

La función de costo de FISTA converge en promedio en la iteración 23, mientras que ISTA en la iteración 39. La disminución de tamaño y error de FISTA es más pronunciada que en el caso de ISTA, habiéndose reducido la función de costo en magnitudes similares en menor número iteraciones.

Además, existe una mayor reducción tanto de error como de tamaño para el caso de *sparsity* de 50% debido a la naturaleza de los algoritmos de optimización por regularización, la cual aprovecha esta característica para la eliminación de valores

En adelante, se genera una comparación entre FISTA con tamaño de paso constante y modificado por las estrategias de optimización.

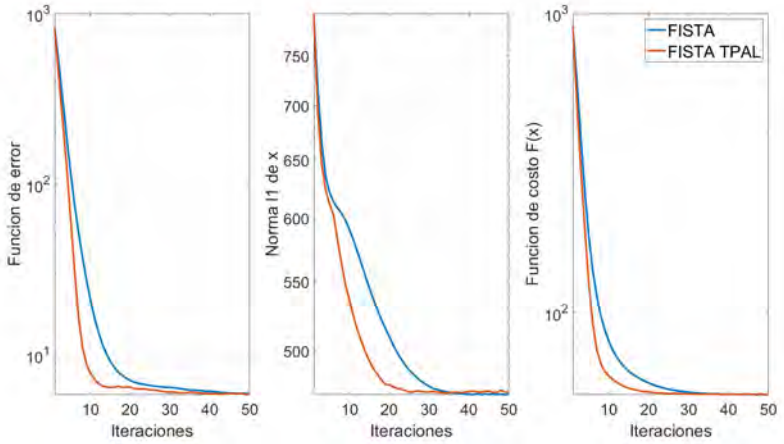
4.1.2 FISTA con tamaño de paso automático

La siguiente comparación será tomando un tamaño de paso automático. El primer tamaño de paso automático será el de Cauchy α_k^C debido a que no requiere de información de la gradiente o valores estimados de iteraciones previas. En el resto de iteraciones se tomará el tamaño de paso de Barzilai y Borwein α_k^{BB1}.

Además, como lo expuesto en el capítulo anterior, debido a que la norma de la gradiente puede ser muy pequeña y generar que los tamaños de paso sean muy elevados, se tomarán límites superiores e inferiores con el fin de impedir la generación de picos en la función de costo y evitar que el algoritmo diverja. Se toman los valores $\alpha_H^C = 2 \cdot 10^{-1}$ y $\alpha_L^C = 10^{-1}$, para Cauchy, y $\alpha_H^{BB1} = 3 \cdot 10^{-1}$ y

$\alpha_L^{BB1} = 1.3 \cdot 10^{-1}$, para Barzilai y Borwein. Además, se multiplica al tamaño de paso automático por un factor de 0.35.

En la Figura 16 se puede observar una comparación de rendimiento entre FISTA y FISTA TPAL (tamaño de paso automático con límites). Así mismo, en las Tablas 3 y 4 se tienen los casos de *sparsity* al 25% y 50%, se aprecian valores obtenidos para 50 iteraciones del algoritmo ejecutado 100 veces, obteniendo la media (negro) y desviación estándar (azul).



(a) (b) (c)

Figura 16 FISTA y FISTA TPAL (tamaño de paso automático con límites).

Iter.	Error FISTA	Error FISTA TPAL	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA TPAL	Costo FISTA	Costo FISTA TPAL
1	879.94±44.02	879.94±44.02	798.25±18.01	798.25±18.01	959.77±44.71	959.77±44.71
10	23.51±1.38	7.46±0.42	569.84±15.51	514.06±14.72	80.49±2.35	58.87±1.62
20	6.78±0.31	6.26±0.24	486.45±15.32	454.03±15.04	55.43±1.65	51.66±1.54
30	6.26±0.23	6.01±0.24	453.31±15.11	450.10±14.81	51.59±1.55	51.02±1.52
40	6.01±0.25	5.95±0.24	448.90±14.83	449.68±14.84	50.91±1.52	50.92±1.51
50	5.92±0.28	5.94±0.23	448.48±14.90	449.58±14.79	50.77±1.51	50.90 ± 1.50

Tabla 3 FISTA y FISTA TPAL (tamaño de paso automático con límites), *sparsity* de 25%. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA TPAL	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA TPAL	Costo FISTA	Costo FISTA TPAL
1	746.06±63.99	746.06±63.99	797.73±20.84	797.73±20.84	825.84±65.20	825.84±65.20
10	18.96±1.14	5.87±0.32	485.83±17.07	393.14±16.05	67.54±2.33	45.19±1.74
20	5.19±0.24	4.31±0.20	358.25±15.89	328.39±13.72	41.01±1.69	37.15±1.38
30	4.34±0.19	3.77±0.19	325.36±13.92	327.97±13.76	36.87±1.40	36.57±1.35
40	3.75±0.21	3.71±0.18	325.85±13.84	327.73±13.92	36.34±1.36	36.48±1.35
50	3.64±0.19	3.74±0.19	326.01±14.02	327.42±13.90	36.24±1.36	36.48 ± 1.34

Tabla 4 FISTA y FISTA TPAL (tamaño de paso automático con límites), *sparsity* de 50%. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

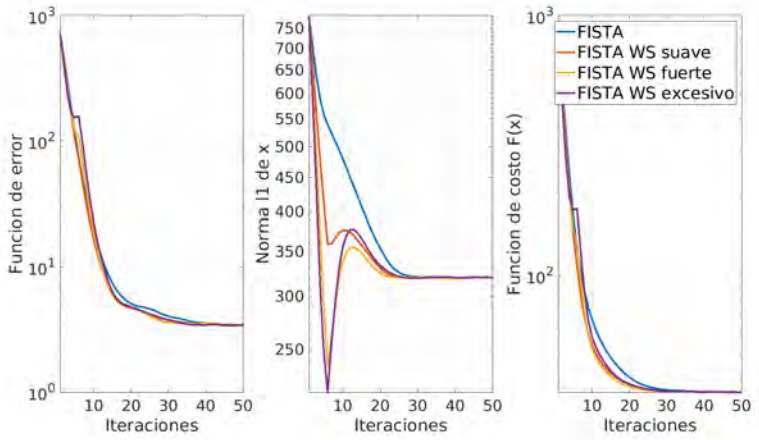
Se tiene un mayor rizado al llegar a la convergencia de la función de costo, esto debido a que los tamaños de paso están oscilando entre extremos y no son constantes. Se observa además un decrecimiento considerable de la norma ℓ_1 y el error, ya que los tamaños de paso actualizables permiten una reducción mayor en cada iteración, con lo que además la norma de la gradiente se minimiza más rápidamente y se alcanza el óptimo en una menor cantidad de iteraciones.

FISTA TPAL logra convergencia alrededor de la iteración 15 para ambos casos de *sparsity*. Similar al caso anterior, al 50% se experimenta una mayor reducción de tamaño tanto como de error.

4.1.3 FISTA con Warm Start

Por medio de *Warm Start*, se logra una reducción inicial mayor en el tamaño de datos incrementando el término de regularización. Debido a que la data es *sparse*, al incrementar el valor de λ, se produce una eliminación de valores que contribuyen al error y no son *features*, con lo que se obtiene una convergencia más rápida del algoritmo.

Para evaluar la mejor configuración de *Warm Start*, son tomados diferentes niveles de incremento para el término de regularización λ en el 10% del total de iteraciones. En la Figura 17 se aprecia la comparación entre FISTA y diferentes niveles de WS (*Warm Start*).



(a) **(b)** **(c)**

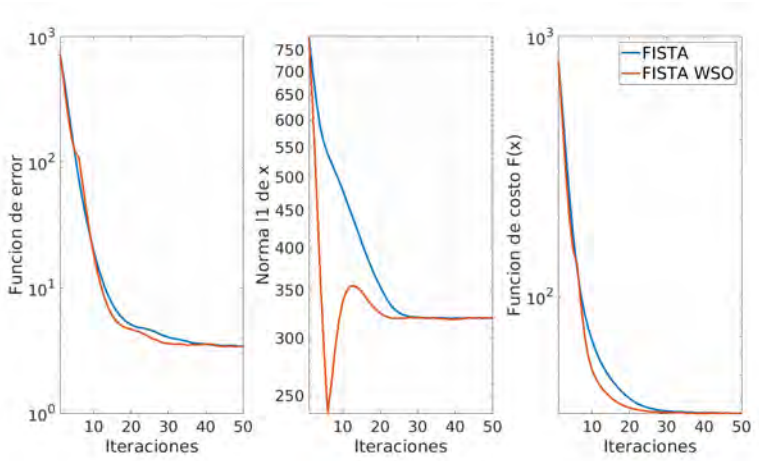
Figura 17 FISTA con diferentes niveles de WS *(Warm Start)*.

Se comparan tres niveles de *Warm Start*. Para *sparsity* al 25%: $\gamma_{suave} = 2$, $\gamma_{fuerte} = 5$ *y* $\gamma_{excesivo} = 10$, y para *sparsity* al 50%: $\gamma_{suave} = 5$, $\gamma_{fuerte} = 10$ y $\gamma_{excesivo} = 15$. Debido a que el caso de 50% es más *sparse*, se puede incrementar el término de regularización.

Para *Warm Start* con un incremento excesivo de λ, se genera una eliminación inadecuada de valores (eliminación de *features*), ya que en el proceso se está incrementando el error.

Además, la función de costo con un incremento del parámetro de regularización fuerte logra la convergencia en una menor cantidad de iteraciones que con un incremento suave y exagerado. Se elige el incremento fuerte del término de regularización λ para FISTA WSO (*Warm Start* Optimizado).

Se observan los resultados en la Figura 18. Además, la media (negro) y desviación estándar (azul) para los dos casos de *sparsity* de valores obtenidos en 50 iteraciones del algoritmo ejecutado 100 veces en las Tablas 5 y 6.



(a) **(b)** **(c)**

Figura 18 FISTA y FISTA WSO (*Warm Start* Optimizado).

Iter.	Error FISTA	Error FISTA WSO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA WSO	Costo FISTA	Costo FISTA WSO
1	879.94±44.02	879.±44.02	798.25±18.01	798.25±18.01	959.77±44.71	959.77±44.71
10	23.51±1.38	23.51±1.38	569.84±15.51	447.86±14.89	80.49±2.35	68.96±2.28
20	6.78±0.31	6.54±0.29	486.45±15.32	464.73±15.25	55.43±1.65	53.02±1.61
30	6.26±0.23	6.09±0.21	453.31±15.11	450.26±15.00	51.59±1.55	51.12±1.54
40	6.01±0.25	5.93±0.23	448.90±14.83	448.59±14.83	50.91±1.52	50.79±1.52
50	5.92±0.28	5.90±0.24	448.48±14.90	448.29±14.83	50.77±1.51	50.73±1.51

Tabla 5 FISTA y FISTA WSO (*Warm Start* Optimizado), *sparsity* de 25%. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA WSO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA WSO	Costo FISTA	Costo FISTA WSO
1	746.06±63.99	746.06±63.99	797.73±20.84	797.73±20.84	825.84±65.20	825.84±65.20
10	18.96±1.14	18.96±1.14	485.83±17.07	344.29±14.25	67.54±2.33	52.31±2.17
20	5.19±0.24	4.73±0.21	358.25±15.89	333.51±14.30	41.01±1.69	38.08±1.50
30	4.34±0.19	3.98±0.20	325.36±13.92	324.32±13.63	36.87±1.40	36.41±1.38
40	3.75±0.21	3.65±0.18	325.85±13.84	325.90±13.94	36.34±1.36	36.24±1.36
50	3.64±0.19	3.64±0.19	326.01±14.02	325.73±14.05	36.24±1.36	36.21±1.35

Tabla 6 FISTA y FISTA WSO (*Warm Start* Optimizado), *sparsity* de 50%. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

FISTA WSO logra converger aproximadamente en la iteración 19. Además, se observa un decrecimiento notable de la norma ℓ_1 en las primeras iteraciones.

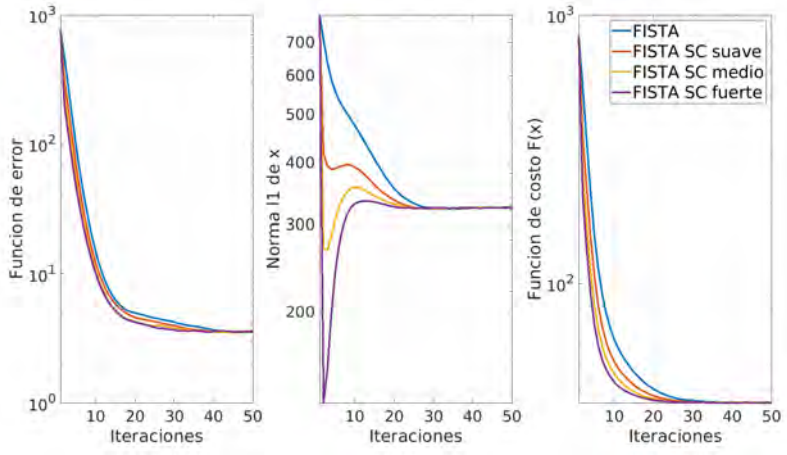
La función de error disminuye más rápido debido a que se eliminan valores que están incrementando la distancia entre los datos estimados y originales, los cuales no son parte de los *features* de la información original. Se alcanza mayor efectividad en el caso más *sparse*.

4.1.4 FISTA con Screening

Para el caso de optimización por *Screening*, se aplicará una reducción inicial al tamaño de los datos estimados previo a ingresar al algoritmo iterativo.

Cuando el número de *features* es elevado el número de valores eliminados es menor y *Screening* tiene menor efecto. Por lo tanto, los datos empleados deben ser de naturaleza *sparse*.

Se compara la reducción de datos por *Screening* para diferentes tamaños de λ_{max}, en su 89%, 91% y al 100%, para niveles de *Screening* suave, medio y fuerte. Se observan los resultados de diferentes niveles de SC (*Screening*) en la Figura 19.



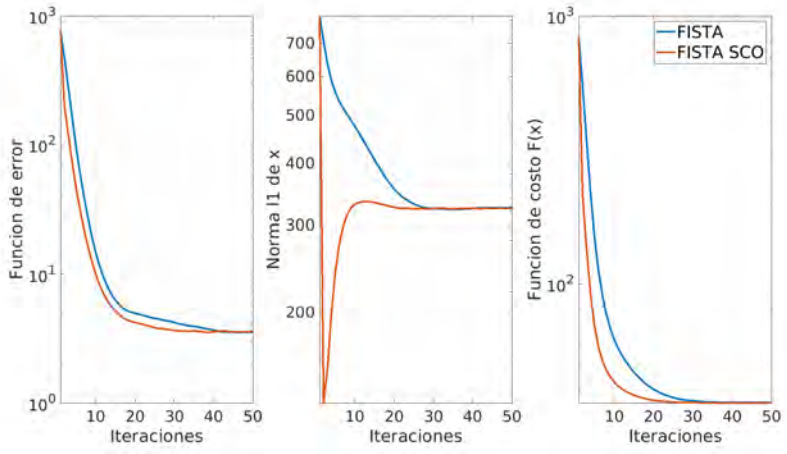
(a) **(b)** **(c)**

Figura 19 FISTA con diferentes niveles de SC (*Screening*).

Debido a que la información es *sparse,* se eliminan variables considerablemente, con lo que la reducción de la función de costo es elevada.

Además, incluso para valores elevados de λ_{max}, se tiene una convergencia más rápida en la función de costo, ya que los datos eliminados mediante *Screening* son valores que son garantizados ser ceros al terminar el algoritmo, por lo que la reducción de tamaño también disminuye considerablemente el error en las primeras iteraciones. Debido a ello se empleará un valor elevado de λ_{max} para FISTA SCO (*Screening* Optimizado).

Se pueden apreciar los resultados en la Figura 20. En las Tablas 7 y 8 se evalúan ambos casos de *sparsity*, en los que se tiene la media (negro) y desviación estándar (azul) para los valores obtenidos en 50 iteraciones del algoritmo ejecutado 100 veces.



(a) **(b)** **(c)**

Figura 20 FISTA y FISTA SCO (*Screening* Optimizado).

Iter.	Error FISTA	Error FISTA SCO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA SCO	Costo FISTA	Costo FISTA SCO
1	879.94±44.02	879.94±44.02	798.25±18.01	798.25±18.01	959.77±44.71	959.77±44.71
10	23.51±1.38	14.97±0.89	569.84±15.51	436.51±14.90	80.49±2.35	58.62±1.94
20	6.78±0.31	6.48±0.27	486.45±15.32	453.06±15.09	55.43±1.65	51.78±1.59
30	6.26±0.23	6.00±0.22	453.31±15.11	448.87±14.96	51.59±1.55	50.89±1.53
40	6.01±0.25	5.89±0.23	448.90±14.83	448.45±14.87	50.91±1.52	50.73±1.51
50	5.92±0.28	5.88±0.23	448.48±14.90	448.21±14.89	50.77±1.51	50.70±1.51

Tabla 7 FISTA y FISTA SCO (*Screening* Optimizado), *sparsity* de 25%. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA SCO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA SCO	Costo FISTA	Costo FISTA SCO
1	746.06±63.99	746.06±63.99	797.73±20.84	797.73±20.84	825.84±65.20	825.84±65.20
10	18.96±1.14	10.48±0.68	485.83±17.07	326.48±15.11	67.54±2.33	43.13±1.87
20	5.19±0.24	4.60±0.19	358.25±15.89	325.16±14.23	41.01±1.69	37.12±1.46
30	4.34±0.19	3.85±0.18	325.36±13.92	324.30±13.70	36.87±1.40	36.28±1.37
40	3.75±0.21	3.64±0.18	325.85±13.84	325.64±13.95	36.34±1.36	36.20±1.36
50	3.64±0.19	3.63±0.18	326.01±14.02	325.61±14.02	36.24±1.36	36.19±1.36

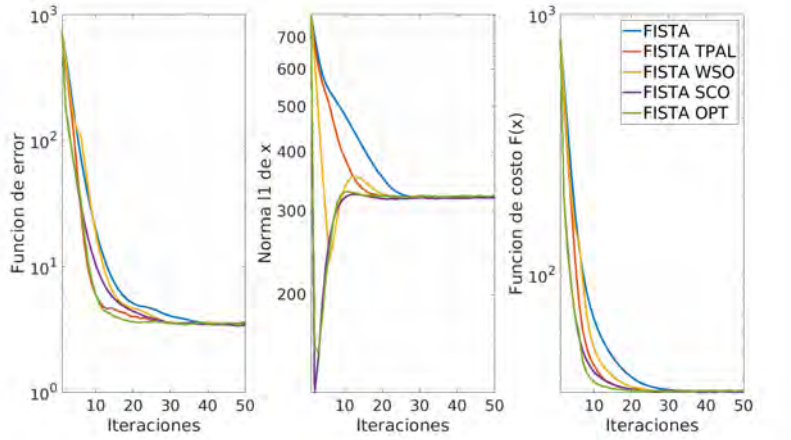
Tabla 8 FISTA y FISTA SCO (*Screening* Optimizado), *sparsity* de 50%. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

La convergencia de la función de costo para FISTA SCO ocurre en la iteración 15. Se aprecia que la norma ℓ_1 de los datos estimados presenta una gran reducción al iniciar el algoritmo, siendo mayor en el caso de *sparsity* al 50%. Posteriormente, sin embargo, se incrementa el tamaño de los valores estimados, lo cual se puede interpretar como una reducción inicial de los datos que estaban garantizados en ser ceros y luego un incremento de los datos que inician con un valor bajo distante a los datos originales. Esto se puede evidenciar ya que la función de error no incrementa como en el caso de *Warm Start* excesivo, donde la eliminación de valores estaba tomando datos que no debía eliminar y se incrementaba la función de error.

4.1.5 FISTA Optimizado

Con el fin de lograr un FISTA OPT (Optimizado), se aplican todos los métodos estudiados en un solo algoritmo. La función optimizada tiene un tamaño de paso actualizable en cada iteración, reducción de tamaño inicial mediante *Screening* con λ_{max} al 100% y además *Warm Start* con $\gamma = 2$ y $\gamma = 3$, para los casos de *sparsity* de 25% y 50%, respectivamente.

Se genera una comparación de todos los métodos aplicados hasta el momento y FISTA OPT en la Figura 21. Además, se calcula la media (negro) y varianza (azul) en 50 iteraciones del algoritmo ejecutado 100 veces para ambos casos de *sparsity* en las Tablas 9 y 10.



(a) **(b)** **(c)**

Figura 21 Comparación entre FISTA Regular, TPAL, WSO, SCO y OPT.

Iter.	Costo FISTA	Costo FISTA TPAL	Costo FISTA WSO	Costo FISTA SCO	Costo FISTA OPT
1	959.77±44.71	959.77±44.71	959.77±44.71	959.77±44.71	959.77±44.71
10	80.49±2.35	58.87±1.62	68.96±2.28	58.62±1.9418	53.94±1.63
20	55.43±1.65	51.66±1.54	53.02±1.61	51.78±1.5934	51.20±1.52
30	51.59±1.55	51.02±1.52	51.12±1.54	50.89±1.5316	50.92±1.51
40	50.91±1.52	50.92±1.51	50.79±1.52	50.73±1.5189	50.89±1.50
50	50.77±1.51	50.90±1.50	50.73±1.51	50.70±1.5140	50.90±1.52

Tabla 9 Comparación entre FISTA Regular, TPAL, WSO, SCO y OPT, *sparsity* de 25%. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Costo FISTA	Costo FISTA TPAL	Costo FISTA WSO	Costo FISTA SCO	Costo FISTA OPT
1	825.84±65.20	825.84±65.20	825.84±65.20	825.84±65.20	825.84±65.20
10	67.54±2.33	45.19±1.74	52.31±2.17	43.13±1.87	39.67±1.52
20	41.01±1.69	37.15±1.38	38.08±1.50	37.12±1.46	36.70±1.37
30	36.87±1.40	36.57±1.35	36.41±1.38	36.28±1.37	36.48±1.34
40	36.34±1.36	36.48±1.35	36.24±1.36	36.20±1.36	36.45±1.36
50	36.24±1.36	36.48 ± 1.34	36.21±1.35	36.19±1.36	36.46 ± 1.36

Tabla 10 Comparación entre FISTA Regular, TPAL, WSO, SCO y OPT, *sparsity* de 50%. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Ocurre un decremento importante de tamaño en las primeras iteraciones producto de *Screening* y *Warm Start*, además, la caída del tamaño se corrige rápidamente debido al tamaño paso actualizable, con lo que converge en una menor cantidad de iteraciones en comparación a los otros métodos

FISTA OPT logra valores de convergencia en la iteración 12, menor que en los demás casos de estudio. Además, se obtiene un mayor decremento para la función de costo de los datos con mayor *sparsity*, tal como en los métodos anteriores.

Cabe destacar que, al aplicar los métodos con los parámetros optimizados calculados previamente, ocurrió una sensibilidad mayor para el caso de *Warm Start*. Al mantener los valores óptimos de incremento iniciales para el término de regularización, el algoritmo llegó a divergir, por lo que se aplicó un incremento suave de γ para *Warm Start*. Esto se puede explicar ya que el término de regularización que es tomado en cuenta para la reducción en cada iteración no

considera el nuevo valor de regularización obtenido en *Warm Start*, lo que puede llevar a un incremento del error y finalmente a divergencia.

En la Figura 22 y la Tabla 11, se genera una comparación de la función de costo en el tiempo para los diferentes métodos evaluados ejecutados 100 veces, obteniendo la media (negro) y varianza (azul) para los tiempos en la iteración de convergencia.

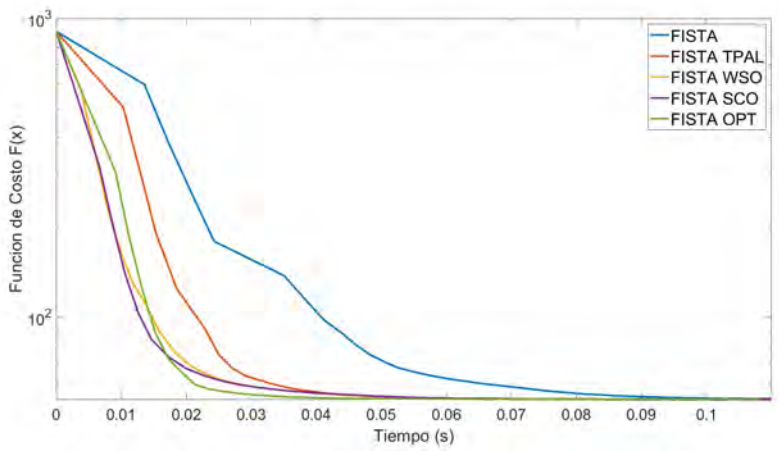


Figura 22 Comparación de tiempo para FISTA Regular, TPAL, WSO, SCO y OPT.

	FISTA	FISTA TPAL	FISTA WSO	FISTA SCO	FISTA OPT
Iter. de conv. (25% sparse)	22.780±0.773	15.270±0.708	18.990±0.594	15.120±0.498	11.870±0.464
Tiempo (25% sparse)	0.053±0.012	0.038±0.017	0.044±0.008	0.037±0.003	0.031±0.003
Iter. de conv. (50% sparse)	23.460±0.838	15.540±0.787	18.740±0.632	14.920±0.723	12.060±0.585
Tiempo (50% sparse)	0.054±0.009	0.036±0.004	0.044±0.006	0.035±0.003	0.030±0.003

Tabla 11 Tiempo de convergencia para FISTA Regular, TPAL, WSO, SCO y OPT. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Las iteraciones y tiempos de convergencia para el caso de *sparsity* al 25% y al 50% son semejantes, sin embargo, el mayor porcentaje obtiene mejores resultados en tanto a la reducción del error y tamaño de los datos estimados.

Cabe destacar que, debido a que los métodos aplicados consisten de cálculos que requieren muy poco costo computacional (incrementar el término de regularización en *Warm Start*, y simples operaciones y comparaciones en *Screening)*, el tiempo requerido para su ejecución es despreciable comparado con la demanda de recursos del algoritmo iterativo. Debido a esto, FISTA OPT converge en un menor número de iteraciones y, además, en un menor tiempo.

4.2 Caso Imágenes mediante la Transformada Wavelet

Para el segundo caso de evaluación, se emplearán imágenes optimizando la información contenida en la Transformada Wavelet. Se evalúan 5 diferentes imágenes de prueba: *Lena, Barbara, Goldhill, House y Baboon.*

La Transformada Wavelet Discreta en 2D emplea coeficientes aproximados y de detalle. Los coeficientes aproximados contienen información de bajas frecuencias y no serán modificados. Por otro lado, los coeficientes de detalle son conformados por información de altas frecuencias y serán optimizados ya que contienen la mayor parte del ruido.

Los coeficientes de detalle son altamente *sparse*, por lo que se obtendrá una convergencia no-asintótica en una menor cantidad de iteraciones.

Para el caso de imágenes, se toman los siguientes valores para los parámetros empleados: Parámetro de regularización $\lambda = 8 \cdot 10^{-2}$, tamaño de paso $\alpha = 5 \cdot 10^{-2}$ (empleado en todos los casos a excepción de FISTA con tamaño de paso actualizable) y parámetro de error aditivo $\sigma = 1.5 \cdot 10^{-1}$.

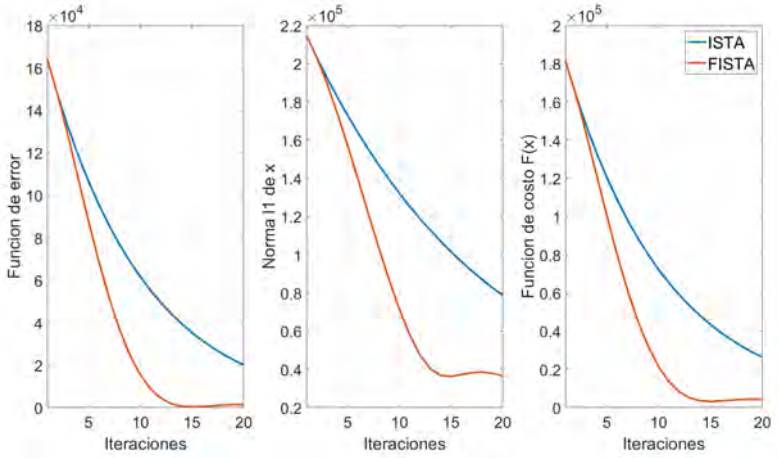
El algoritmo logra convergencia si la diferencia entre dos iteraciones de la función costo es menor a 10^2.

4.2.1 ISTA y FISTA

Siguiendo el orden antes establecido, la primera comparación a realizarse es entre ISTA y FISTA, ambos con tamaño de paso constante de $\alpha = 5 \cdot 10^{-2}$, siendo consideradas las primeras 20 iteraciones.

Se considera un gráfico para la evaluación de las 5 imágenes debido a que mantienen una estructura similar. Para mayor detalle se pueden comprobar los resultados en 5 diferentes tablas para cada caso.

Se observan los resultados en la Figura 23. Además, en las Tablas 12, 13, 14, 15 y 16 se aprecian valores obtenidos para diferentes imágenes en 20 iteraciones del algoritmo ejecutado 100 veces, hallando así la media (negro) y desviación estándar (azul) de los datos estadísticos de interés para determinadas iteraciones.



(a) **(b)** **(c)**

Figura 23 ISTA y FISTA.

Iter.	Error ISTA	Error FISTA	Norma ℓ_1 ISTA	Norma ℓ_1 FISTA	Costo ISTA	Costo FISTA
1	1.69e5±338.13	1.69e5±338.13	2.14e5±371.28	2.14e5±371.28	1.86e5±363.06	1.86e5±363.06
5	1.10e5±220.99	9.06e4±182.71	1.72e5±313.86	1.56e5±295.55	1.23e5±242.11	1.03e5±202.60
10	6.37e4±129.24	1.61e4±33.84	1.32e5±269.17	7.14e4±223.37	7.42e4±147.32	2.18e4±46.85
15	3.65e4±75.06	327.44±0.71	1.01e5±235.80	3.61e4±116.70	4.46e4±90.72	3.21e3±9.08
20	2.07e4±43.19	1.42e3±3.54	7.85e4±207.54	3.64e4±144.69	2.70e4±56.79	4.33e3±12.46

Tabla 12 Lena: ISTA y FISTA. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error ISTA	Error FISTA	Norma ℓ_1 ISTA	Norma ℓ_1 FISTA	Costo ISTA	Costo FISTA
1	1.76e5±301.31	1.76e5±301.31	2.14e5±275.83	2.14e5±275.83	1.93e5±312.51	1.96e5±312.51
5	1.14e5±199.07	9.46e4±165.52	1.72e5±230.06	1.57e5±212.64	1.28e5±207.77	1.07e5±173.36
10	6.65e4±118.45	1.69e4±33.03	1.32e5±185.99	7.25e4±109.57	7.71e4±125.02	2.27e4±36.03
15	3.82e4±70.28	350.15±0.74	1.01e5±152.67	3.92e4±84.59	4.64e4±75.48	3.49e3±6.68
20	2.18e4±41.41	1.47e3±3.62	7.94e4±123.18	3.96e4±105.65	2.81e4±45.09	4.64e3±7.03

Tabla 13 Barbara: ISTA y FISTA. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error ISTA	Error FISTA	Norma ℓ_1 ISTA	Norma ℓ_1 FISTA	Costo ISTA	Costo FISTA
1	1.64e5±406.29	1.64e5±406.29	2.14e5±257.30	2.14e5±257.30	1.81e5±421.59	1.81e5±421.59
5	1.06e5±266.52	8.76e4±220.81	1.72e5±217.71	1.57e5±204.86	1.20e5±278.63	1.00e5±231.93
10	6.15e4±156.97	1.54e4±42.31	1.32e5±181.13	7.14e4±131.35	7.20e4±166.22	2.11e4±48.15
15	3.52e4±91.98	322.10±0.71	1.01e5±155.96	3.63e4±109.90	4.33e4±99.05	3.23e3±8.50
20	2.00e4±53.98	1.38e3±3.09	7.86e4±138.52	3.66e4±127.09	2.63e4±59.91	4.31e3±10.52

Tabla 14 Goldhill: ISTA y FISTA. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error ISTA	Error FISTA	Norma ℓ_1 ISTA	Norma ℓ_1 FISTA	Costo ISTA	Costo FISTA
1	1.53e5±385.89	1.53e5±385.89	2.14e5±308.00	2.14e5±308.00	1.70e5±403.77	1.70e5±403.77
5	9.95e4±252.89	8.19e4±209.30	1.72e5±252.69	1.57e5±231.24	1.13e5±266.36	9.44e4±220.98
10	5.74e4±148.30	1.43e4 ± 39.06	1.32e5±197.65	7.28e4±133.91	6.80e4±156.92	2.01e4±40.10
15	3.28e4±86.34	327.50±0.71	1.02e5±160.38	4.01e4±110.63	4.09e4±91.10	3.53e3±8.66
20	1.85e4±49.67	1.31e3±2.98	7.97e4±141.30	4.05e4±107.14	2.49e4±51.66	4.56e3±8.17

Tabla 15 House: ISTA y FISTA. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error ISTA	Error FISTA	Norma ℓ_1 ISTA	Norma ℓ_1 FISTA	Costo ISTA	Costo FISTA
1	1.74e5±425.14	1.71e5±425.14	2.14e5±360.24	2.14e5±360.24	1.88e5±444.71	1.88e5±444.71
5	1.11e5±279.05	9.14e4±231.32	1.72e5±293.83	1.57e5±270.68	1.24e5±294.79	1.04e5±245.83
10	6.42e4±164.58	1.62e4±44.15	1.32e5±235.59	7.24e4±128.78	7.48e4±177.41	2.20e4±51.27
15	3.69e4 ± 96.45	344.75±0.70	1.01e5±186.73	3.94e4±82.93	4.50e4±106.81	3.49e3±7.04
20	2.10e4±56.04	1.44e3±3.03	7.93e4±144.67	3.97e4±98.96	2.73e4±64.04	4.62e3±9.09

Tabla 16 Baboon: ISTA y FISTA. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

FISTA, al ser la versión acelerada de ISTA, tiene menor valor para la función de costo en todas las iteraciones.

Se observa también, que ocurren pequeños incrementos en la función de costo en FISTA al acercarse al valor de convergencia. A diferencia del decrecimiento continuo observado en ISTA, FISTA es un algoritmo no monotono [13], por lo que los valores de la función no están garantizados en no incrementarse en todas las iteraciones.

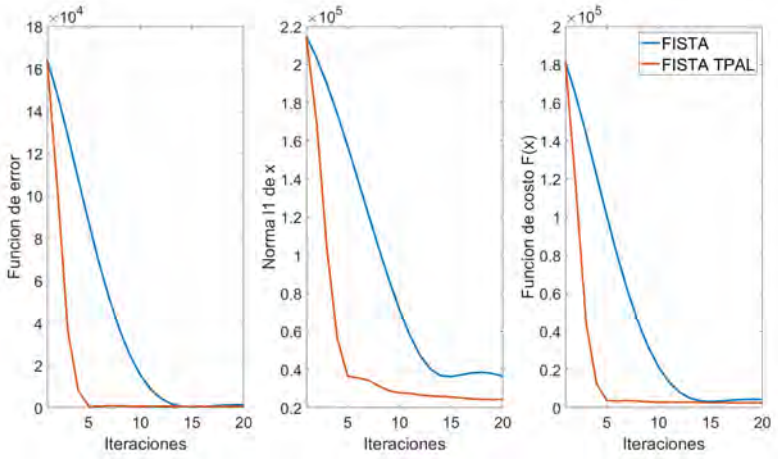
FISTA logra convergencia en la iteración 19, mientras que ISTA no logra convergencia en el número establecido de 20 iteraciones.

En adelante, se genera una comparación entre FISTA con tamaño de paso constante y modificado por las estrategias de optimización.

4.2.2 FISTA con tamaño de paso automático

Como en el caso aleatorio, el primer tamaño de paso automático será el de Cauchy con límites $\alpha_H^C = 2 \cdot 10^{-1}$ y $\alpha_L^C = 10^{-1}$, con un factor multiplicativo de 10^{-1}. El resto de iteraciones se tomará el tamaño de paso de Barzilai y Borwein con $\alpha_H^{BB1} = 3 \cdot 10^{-1}$ y $\alpha_L^{BB1} = 2 \cdot 10^{-1}$ y factor multiplicativo de $3 \cdot 10^{-1}$.

En la Figura 24 se puede observar una comparación de rendimiento entre FISTA y FISTA TPAL (tamaño de paso automático con límites). Así mismo, en las Tablas 17, 18, 19, 20 y 21 se aprecian valores obtenidos para 20 iteraciones del algoritmo ejecutado 100 veces, obteniendo la media (negro) y desviación estándar (azul).



(a) **(b)** **(c)**

Figura 24 FISTA y FISTA TPAL (tamaño de paso automático con límites).

Iter.	Error FISTA	Error FISTA TPAL	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA TPAL	Costo FISTA	Costo FISTA TPAL
1	1.69e5±338.13	1.69e5±338.13	2.14e5±371.28	2.14e5±371.28	1.86e5±363.06	1.88e5±363.06
5	9.06e4±182.71	875.19±1.28	1.56e5±295.55	3.63e4±118.15	1.03e5±202.60	3.78e3±9.59
10	1.61e4±33.84	707.26±0.95	7.14e4±223.37	2.76e4±121.38	2.18e4±46.85	2.91e3±9.49
15	327.44±0.71	629.01±0.98	3.61e4±116.70	2.54e4±107.66	3.21e3±9.08	2.66e3±8.45
20	1.42e3±3.54	634.65±1.26	3.64e4±144.69	2.42e4±114.31	4.33e3±12.46	2.57e3±8.84

Tabla 17 Lena: FISTA y FISTA TPAL (tamaño de paso automático con límites). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA TPAL	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA TPAL	Costo FISTA	Costo FISTA TPAL
1	1.76e5±301.31	1.76e5±301.31	2.14e5±275.83	2.14e5±275.83	1.96e5±312.51	1.96e5±312.51
5	9.46e4±165.52	965.26±2.51	1.57e5±212.64	3.92e4±77.09	1.07e5±173.36	4.10e3±6.36
10	1.69e4±33.034	722.24±1.42	7.25e4±109.57	3.08e4±91.95	2.27e4±36.03	3.19e3±6.38
15	350.15±0.74	646.76±1.00	3.92e4±84.59	2.87e4±85.09	3.49e3±6.68	2.94e3±7.00
20	1.47e3±3.62	641.58±1.02	3.96e4±105.65	2.75e4±91.08	4.64e3±7.03	2.84e3±6.69

Tabla 18 Barbara: FISTA y FISTA TPAL (tamaño de paso automático con límites). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA TPAL	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA TPAL	Costo FISTA	Costo FISTA TPAL
1	1.64e5±406.29	1.64e5±406.29	2.14e5±257.30	2.14e5±257.30	1.81e5±421.59	1.81e5±421.59
5	8.76e4±220.81	835.73±2.86	1.57e5±204.86	3.66e4±105.71	1.00e5±231.93	3.78e3±9.25
10	1.54e4±42.31	703.64±1.15	7.14e4±131.35	2.78e4±98.41	2.11e4±48.15	2.92e3±7.97
15	322.10±0.71	630.21±1.29	3.63e4±109.90	2.56e4±79.43	3.23e3±8.50	2.68e3±6.02
20	1.38e3±3.09	636.17±1.56	3.66e4±127.09	2.44e4±75.87	4.31e3±10.52	2.59e3±6.62

Tabla 19 Goldhill: FISTA y FISTA TPAL (tamaño de paso automático con límites). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA TPAL	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA TPAL	Costo FISTA	Costo FISTA TPAL
1	1.53e5±385.89	1.53e5±385.89	2.14e5±308.00	2.14e5±308.00	1.70e5±403.77	1.70e5±403.77
5	8.19e4±209.30	802.67±1.62	1.57e5±231.24	3.99e4±113.86	9.44e4±220.98	4.00e3±8.03
10	1.43e4±39.06	701.27±1.40	7.28e4±133.91	3.18e4±108.84	2.01e4±40.10	3.25e3±9.10
15	327.50±0.71	644.47±0.97	4.01e4±110.63	2.96e4±110.15	3.53e3±8.66	3.01e3±8.74
20	1.31e3±2.98	649.04±0.72	4.05e4±107.14	2.85e4±102.30	4.56e3±8.17	2.93e3±7.74

Tabla 20 House: FISTA y FISTA TPAL (tamaño de paso automático con límites). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA TPAL	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA TPAL	Costo FISTA	Costo FISTA TPAL
1	1.71e5±425.14	1.71e5±425.14	2.14e5±360.24	2.14e5±360.24	1.88e5±444.71	1.88e5±444.71
5	9.14e4±231.32	925.50±3.25	1.57e5±270.68	3.92e4±77.94	1.04e5±245.83	4.06e3±8.54
10	1.62e4±44.15	723.81±0.93	7.24e4±128.78	3.10e4±87.73	2.20e4±51.27	3.20e3±7.47
15	344.75±0.70	648.50±0.68	3.94e4±82.93	2.89e4±64.44	3.49e3±7.04	2.96e3±4.84
20	1.44e3±3.03	648.45±1.14	3.97e4±98.96	2.78e4±66.99	4.62e3±9.09	2.87e3±5.83

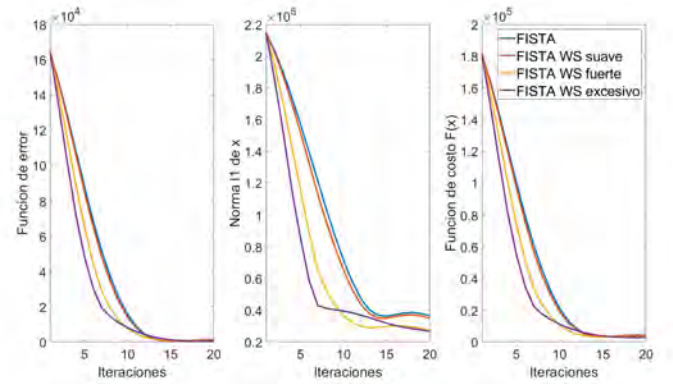
Tabla 21 Baboon: FISTA y FISTA TPAL (tamaño de paso automático con límites). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Gracias al tamaño de paso, FISTA TPAL logra convergencia en la iteración 10, cerca a la mitad que en el caso de FISTA sin optimizar. Además, los crecimientos no monotonos de FISTA se reducen considerablemente al tener tamaño de paso automático limitado que no llegue a extremos.

4.2.3 FISTA con Warm Start

Para el caso de optimización de *Warm Start*, se evaluarán tres niveles del parámetro de regularización. Con lo que se obtienen tres grados de incremento de λ para las primeras iteraciones: $\gamma_{suave} = 2$, $\gamma_{fuerte} = 10$ *y* $\gamma_{excesivo} = 20$.

En la Figura 25 se aprecia la comparación entre FISTA y diferentes niveles de WS (*Warm Start*).

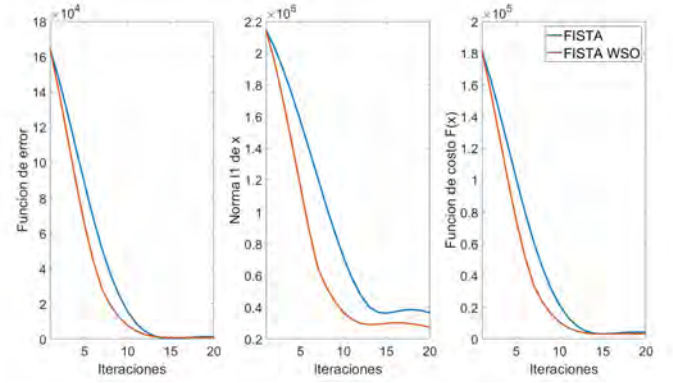


(a) **(b)** **(c)**

Figura 25 FISTA con diferentes niveles de WS (*Warm Start*).

Warm Start excesivo con un parámetro γ demasiado elevado lleva a que la norma ℓ_1 disminuya demasiado y logre mayores valores para la función de costo. Se observa que *Warm Start* con un incremento fuerte logra menores valores para las tres métricas de evaluación. Tomando estas consideraciones, se toma *Warm Start* fuerte como WSO (*Warm Start* Optimizado).

Se observan los resultados en la Figura 26, además de la media (negro) y desviación estándar (azul) para los valores obtenidos en 20 iteraciones del algoritmo ejecutado 100 veces en las Tablas 22, 23, 24, 25 y 26.



(a) **(b)** **(c)**

Figura 26 FISTA y FISTA WSO (*Warm Start* Optimizado).

Iter.	Error FISTA	Error FISTA WSO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA WSO	Costo FISTA	Costo FISTA WSO
1	1.69e5±338.13	1.69e5±338.13	2.14e5±371.28	2.14e5±371.28	1.86e5±363.06	1.86e5±363.06
5	9.06e4±182.71	6.78e4±147.68	1.56e5±295.55	1.15e5±284.59	1.03e5±202.60	7.70e4±165.29
10	1.61e4±33.84	1.61e4±33.84	7.14e4±223.37	3.66e4±121.82	2.18e4±46.85	1.11e4±27.22
15	327.44±0.71	800.81±2.26	3.61e4±116.70	2.97e4±97.02	3.21e3±9.08	3.17e3±7.81
20	1.42e3±3.54	937.75±2.59	3.64e4±144.69	2.72e4±114.64	4.33e3±12.46	3.11e3±9.05

Tabla 22 Lena: FISTA y FISTA WSO (*Warm Start* Optimizado). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA WSO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA WSO	Costo FISTA	Costo FISTA WSO
1	1.76e5±301.31	1.76e5±301.31	2.14e5±275.83	2.14e5±275.83	1.96e5±312.51	1.96e5±312.51
5	9.46e4±165.52	7.18e4±160.28	1.57e5±212.64	1.15e5±203.93	1.07e5±173.36	8.10e4±165.51
10	1.69e4±33.034	1.69e4±33.03	7.25e4±109.57	3.79e4±71.63	2.27e4±36.03	1.22e4±32.60
15	350.15±0.74	856.09±2.26	3.92e4±84.59	3.25e4±78.62	3.49e3±6.68	3.45e3±5.14
20	1.47e3±3.62	988.98±2.69	3.96e4±105.65	3.07e4±67.97	4.64e3±7.03	3.44e3±5.50

Tabla 23 Barbara: FISTA y FISTA WSO (*Warm Start* Optimizado). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA WSO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA WSO	Costo FISTA	Costo FISTA WSO
1	1.64e5±406.29	1.64e5±406.29	2.14e5±257.30	2.14e5±257.30	1.81e5±421.59	1.81e5±421.59
5	8.76e4±220.81	6.48e4±189.72	1.57e5±204.86	1.57e5±204.86	1.00e5±231.93	7.40e4±199.66
10	1.54e4±42.31	1.54e4±42.31	7.14e4±131.35	7.14e4±131.35	2.11e4±48.15	1.05e4±31.07
15	322.10±0.71	799.02±1.16	3.63e4±109.90	3.63e4±109.90	3.23e3±8.50	3.19e3±7.04
20	1.38e3±3.09	903.23±2.51	3.66e4±127.09	3.66e4±127.09	4.31e3±10.52	3.10e3±5.83

Tabla 24 Goldhill: FISTA y FISTA WSO (*Warm Start* Optimizado). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA WSO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA WSO	Costo FISTA	Costo FISTA WSO
1	1.53e5±385.89	1.53e5±385.89	2.14e5±308.00	2.14e5±308.00	1.70e5±403.77	1.70e5±403.77
5	8.19e4±209.30	5.91e4±175.30	1.57e5±231.24	1.16e5±211.49	9.44e4±220.98	6.84e4±185.07
10	1.43e4±39.06	1.43e4±39.06	7.28e4±133.91	3.82e4±93.68	2.01e4±40.10	9.71e3±21.37
15	327.50±0.71	841.27±1.80	4.01e4±110.63	3.33e4±66.82	3.53e3±8.66	3.51e3±6.08
20	1.31e3±2.98	833.82±2.22	4.05e4±107.14	3.17e4±74.65	4.56e3±8.17	3.37e3±5.29

Tabla 25 House: FISTA y FISTA WSO (*Warm Start* Optimizado). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

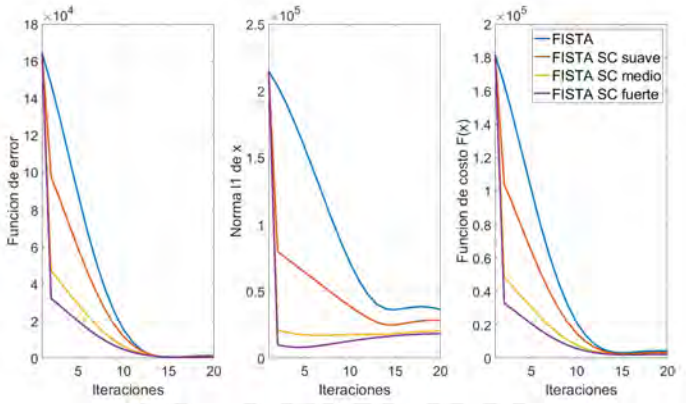
Iter.	Error FISTA	Error FISTA WSO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA WSO	Costo FISTA	Costo FISTA WSO
1	1.71e5±425.14	1.71e5±425.14	2.14e5±360.24	2.14e5±360.24	1.88e5±444.71	1.88e5±444.71
5	9.14e4±231.32	6.86e4±200.95	1.57e5±270.68	1.15e5±257.68	1.04e5±245.83	7.79e4±212.87
10	1.62e4±44.15	1.62e4±44.15	7.24e4±128.78	3.78e4±72.20	2.20e4±51.27	1.15e4±32.38
15	344.75±0.70	855.87±1.11	3.94e4±82.93	3.26e4±72.70	3.49e3±7.04	3.47e3±5.62
20	1.44e3±3.03	957.49±2.16	3.97e4±98.96	3.09e4±83.62	4.62e3±9.09	3.43e3±7.48

Tabla 26 Baboon: FISTA y FISTA WSO (*Warm Start* Optimizado). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Se observa un mayor decrecimiento para las tres métricas en el caso de las iteraciones iniciales con *Warm Start*. Además, incluso al finalizar el algoritmo, se aprecia una mayor reducción de tamaño para FISTA WSO. Se logra convergencia en la iteración 15.

4.2.4 FISTA con Screening

La optimización por *Screening* también será empleada en tres niveles, modificando los valores de λ_{max} al 30%, 50% y 100%. Se observan los resultados de diferentes niveles de SC (*Screening*) en la Figura 27.

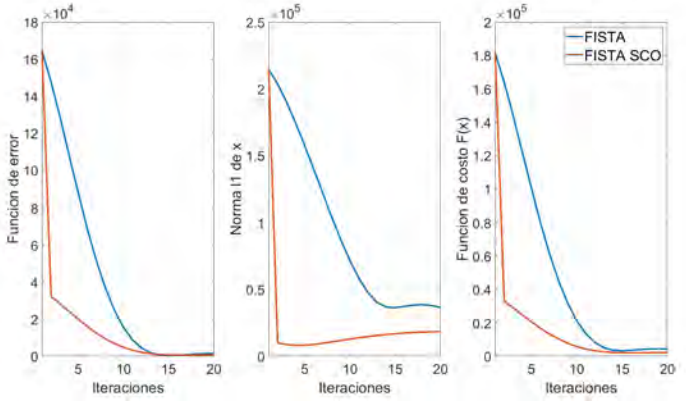


(a) **(b)** **(c)**

Figura 27 FISTA con diferentes niveles de SC (*Screening*).

Similarmente al caso aleatorio, se elige el nivel de *Screening* más elevado ya que, al asegurar eliminar valores que son garantizados ser ceros al finalizar el algoritmo, no se incrementa la función de error. *Screening* fuerte es considerado para SCO (*Screening* Optimizado).

Se pueden apreciar los resultados en la Figura 28 y en las Tablas 27, 28, 29, 30 y 31, la media (negro) y desviación estándar (azul) para los valores obtenidos en 20 iteraciones del algoritmo ejecutado 100 veces.



(a) **(b)** **(c)**

Figura 28 FISTA y FISTA SCO (*Screening* Optimizado).

Iter.	Error FISTA	Error FISTA SCO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA SCO	Costo FISTA	Costo FISTA SCO
1	1.69e5±338.13	1.69e5±338.13	2.14e5±371.28	2.14e5±371.28	1.86e5±363.06	1.86e5±363.06
5	9.06e4±182.71	2.28e4±162.25	1.56e5±295.55	8.14e3±16.08	1.03e5±202.60	2.35e4±161.69
10	1.61e4±33.84	5.35e3±33.41	7.14e4±223.37	1.26e4±24.87	2.18e4±46.85	6.36e3±33.46
15	327.44±0.71	685.83±0.92	3.61e4±116.70	1.66e4±33.83	3.21e3±9.08	2.01e3±3.39
20	1.42e3±3.54	786.51±2.06	3.64e4±144.69	1.81e4±38.08	4.33e3±12.46	2.23e3±4.16

Tabla 27 Lena: FISTA y FISTA SCO (*Screening* Optimizado). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA SCO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA SCO	Costo FISTA	Costo FISTA SCO
1	1.76e5±301.31	1.76e5±301.31	2.14e5±275.83	2.14e5±275.83	1.96e5±312.51	1.96e5±312.51
5	9.46e4±165.52	2.68e4±144.14	1.57e5±212.64	8.92e3±15.78	1.07e5±173.36	2.75e4±144.25
10	1.69e4±33.034	6.22e3±30.14	7.25e4±109.57	1.49e4±26.95	2.27e4±36.03	7.41e3±31.55
15	350.15±0.74	715.98±0.70	3.92e4±84.59	1.99e4±38.51	3.49e3±6.68	2.31e3±3.65
20	1.47e3±3.62	824.11±1.85	3.96e4±105.65	2.19e4±45.34	4.64e3±7.03	2.57e3±5.11

Tabla 28 Barbara: FISTA y FISTA SCO (*Screening* Optimizado). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA SCO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA SCO	Costo FISTA	Costo FISTA SCO
1	1.64e5±406.29	1.64e5±406.29	2.14e5±257.30	2.14e5±257.30	1.81e5±421.59	1.81e5±421.59
5	8.76e4±220.81	1.98e4±87.13	1.57e5±204.86	8.19e3±10.72	1.00e5±231.93	2.04e4±87.33
10	1.54e4±42.31	4.72e3±17.98	7.14e4±131.35	1.28e4±18.26	2.11e4±48.15	5.75e3±18.51
15	322.10±0.71	682.86±1.00	3.63e4±109.90	1.68e4±27.25	3.23e3±8.50	2.03e3±2.99
20	1.38e3±3.09	750.30±1.34	3.66e4±127.09	1.84e4±28.34	4.31e3±10.52	2.22e3±3.10

Tabla 29 Goldhill: FISTA y FISTA SCO (*Screening* Optimizado). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA SCO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA SCO	Costo FISTA	Costo FISTA SCO
1	1.53e5±385.89	1.53e5±385.89	2.14e5±308.00	2.14e5±308.00	1.70e5±403.77	1.70e5±403.77
5	8.19e4±209.30	1.41e4±119.20	1.57e5±231.24	9.17e3±20.66	9.44e4±220.98	1.48e4±118.57
10	1.43e4±39.06	3.63e3±24.43	7.28e4±133.91	1.56e4±28.69	2.01e4±40.10	4.88e3±23.66
15	327.50±0.71	696.35±0.71	4.01e4±110.63	2.09e4±38.79	3.53e3±8.66	2.37e3±3.47
20	1.31e3±2.98	666.92±1.52	4.05e4±107.14	2.30e4±44.62	4.56e3±8.17	2.51e3±3.59

Tabla 30 House: FISTA y FISTA SCO (*Screening* Optimizado). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Error FISTA	Error FISTA SCO	Norma ℓ_1 FISTA	Norma ℓ_1 FISTA SCO	Costo FISTA	Costo FISTA SCO
1	1.71e5±425.14	1.71e5±425.14	2.14e5±360.24	2.14e5±360.24	1.88e5±444.71	1.88e5±444.71
5	9.14e4±231.32	2.37e4±136.37	1.57e5±270.68	9.00e3±13.68	1.04e5±245.83	2.44e4±136.64
10	1.62e4±44.15	5.59e3±28.42	7.24e4±128.78	1.52e4±24.21	2.20e4±51.27	6.81e3±29.15
15	344.75±0.70	715.44±0.74	3.94e4±82.93	2.03e4±36.20	3.49e3±7.04	2.34e3±3.32
20	1.44e3±3.03	789.80±1.97	3.97e4±98.96	2.23e4±40.55	4.62e3±9.09	2.57e3±4.25

Tabla 31 Baboon: FISTA y FISTA SCO (*Screening* Optimizado). Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

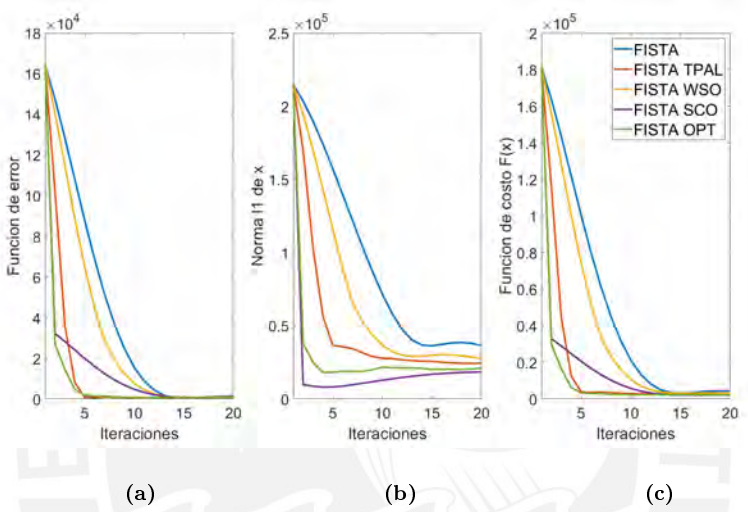
Como se aprecia, se logra una gran reducción de tamaño inicial e incluso al finalizar en el algoritmo iterativo. FISTA SCO converge en la iteración 15.

4.2.5 FISTA Optimizado

Finalmente, buscando lograr un FISTA OPT (Optimizado), se aplican todos los métodos estudiados en un solo algoritmo:

Tamaño de paso actualizable en cada iteración (primera iteración por el método de Cauchy y el resto por Barzilai y Borwein), reducción de tamaño inicial mediante *Screening* con λ_{max} al 100% y, además, *Warm Start* con un incremento del término de regularización de $\gamma = 1.5$.

Se genera una comparación de todos los métodos aplicados hasta el momento y FISTA OPT en la Figura 29. Además, se calcula la media (negro) y varianza (azul) en 20 iteraciones del algoritmo ejecutado 100 veces en las Tablas 32, 33, 34, 35 y 36.



(a) **(b)** **(c)**

Figura 29 Comparación entre FISTA Regular, TPAL, WSO, SCO y OPT.

Iter.	Costo FISTA	Costo FISTA TPAL	Costo FISTA WSO	Costo FISTA SCO	Costo FISTA OPT
1	1.86e5±363.06	1.88e5±363.06	1.86e5±363.06	1.86e5±363.06	1.86e5±363.06
5	1.03e5±202.60	3.78e3±9.59	7.70e4±165.29	2.35e4±161.69	3.33e3±5.48
10	2.18e4±46.85	2.91e3±9.49	1.11e4±27.22	6.36e3±33.46	2.35e3±3.18
15	3.21e3±9.08	2.66e3±8.45	3.17e3±7.81	2.01e3±3.39	2.23e3±3.91
20	4.33e3±12.46	2.57e3±8.84	3.11e3±9.05	2.23e3±4.16	2.36e3±3.84

Tabla 32 Lena: Comparación entre FISTA Regular, TPAL, WSO, SCO y OPT. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Costo FISTA	Costo FISTA TPAL	Costo FISTA WSO	Costo FISTA SCO	Costo FISTA OPT
1	1.96e5±312.51	1.96e5±312.51	1.96e5±312.51	1.96e5±312.51	1.96e5±312.51
5	1.07e5±173.36	4.10e3±6.36	8.10e4±165.51	2.75e4±144.25	3.70e3±7.36
10	2.27e4±36.03	3.19e3±6.38	1.22e4±32.60	7.41e3±31.55	2.61e3±3.50
15	3.49e3±6.68	2.94e3±7.00	3.45e3±5.14	2.31e3±3.65	2.49e3±4.42
20	4.64e3±7.03	2.84e3±6.69	3.44e3±5.50	2.57e3±5.11	2.64e3±4.54

Tabla 33 Barbara: Comparación entre FISTA Regular, TPAL, WSO, SCO y OPT. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Costo FISTA	Costo FISTA TPAL	Costo FISTA WSO	Costo FISTA SCO	Costo FISTA OPT
1	1.81e5±421.59	1.81e5±421.59	1.81e5±421.59	1.81e5±421.59	1.81e5±421.59
5	1.00e5±231.93	3.78e3±9.25	7.40e4±199.66	2.04e4±87.33	3.27e3±3.17
10	2.11e4±48.15	2.92e3±7.97	1.05e4±31.07	5.75e3±18.51	2.36e3±3.01
15	3.23e3±8.50	2.68e3±6.02	3.19e3±7.04	2.03e3±2.99	2.26e3±3.84
20	4.31e3±10.52	2.59e3±6.62	3.10e3±5.83	2.22e3±3.10	2.37e3±4.00

Tabla 34 Goldhill: Comparación entre FISTA Regular, TPAL, WSO, SCO y OPT. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Costo FISTA	Costo FISTA TPAL	Costo FISTA WSO	Costo FISTA SCO	Costo FISTA OPT
1	1.70e5±403.77	1.70e5±403.77	1.70e5±403.77	1.70e5±403.77	1.70e5±403.77
5	9.44e4±220.98	4.00e3±8.03	6.84e4±185.07	1.48e4±118.57	3.44e3±4.33
10	2.01e4±40.10	3.25e3±9.10	9.71e3±21.37	4.88e3±23.66	2.70e3±3.50
15	3.53e3±8.66	3.01e3±8.74	3.51e3±6.08	2.37e3±3.47	2.60e3±4.80
20	4.56e3±8.17	2.93e3±7.74	3.37e3±5.29	2.51e3±3.59	2.69e3±5.77

Tabla 35 House: Comparación entre FISTA Regular, TPAL, WSO, SCO y OPT. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Iter.	Costo FISTA	Costo FISTA TPAL	Costo FISTA WSO	Costo FISTA SCO	Costo FISTA OPT
1	1.88e5±444.71	1.88e5±444.71	1.88e5±444.71	1.88e5±444.71	1.88e5±444.71
5	1.04e5±245.83	4.06e3±8.54	7.79e4±212.87	2.44e4±136.64	3.64e3±5.08
10	2.20e4±51.27	3.20e3±7.47	1.15e4±32.38	6.81e3±29.15	2.65e3±3.47
15	3.49e3±7.04	2.96e3±4.84	3.47e3±5.62	2.34e3±3.32	2.53e3±4.27
20	4.62e3±9.09	2.87e3±5.83	3.43e3±7.48	2.57e3±4.25	2.67e3±2.99

Tabla 36 Baboon: Comparación entre FISTA Regular, TPAL, WSO, SCO y OPT. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Se observa que FISTA OPT alcanza convergencia en 6 iteraciones, logrando mejor rendimiento comparado con los demás métodos de optimización.

Además, en la Figura 30 y la Tabla 37, se genera una comparación de la función de costo en el tiempo para los diferentes métodos evaluados ejecutados 100 veces, obteniendo la media (negro) y varianza (azul) para los tiempos en la iteración de convergencia.

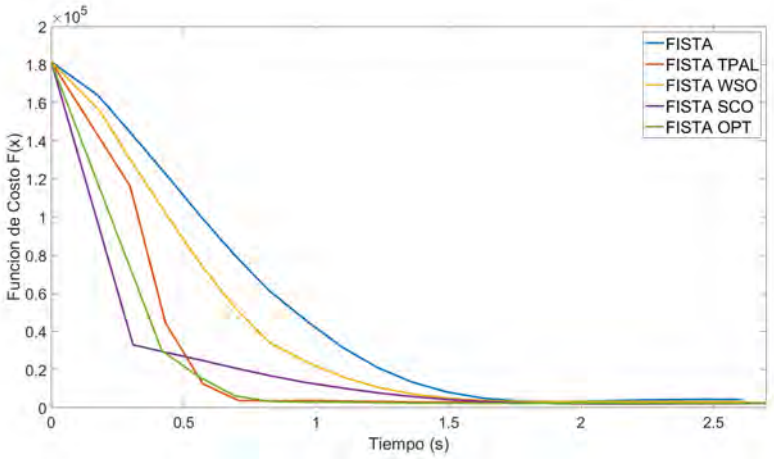


Figura 30 Comparación de tiempo para FISTA Regular, TPAL, WSO, SCO y OPT.

	FISTA	FISTA TPAL	FISTA WSO	FISTA SCO	FISTA OPT
Iter. de conv.	19	10	15	15	6
Tiempo (Lena)	2.621±0.027	1.553±0.015	2.065±0.007	2.197±0.018	1.125±0.019
Tiempo (Barbara)	2.612±0.024	1.550±0.014	2.061±0.012	2.189±0.015	1.140±0.074
Tiempo (Goldhill)	2.638±0.071	1.557±0.015	2.070±0.011	2.201±0.014	1.117±0.014
Tiempo (House)	2.639±0.053	1.558±0.015	1.910±0.008	2.199±0.009	1.124±0.024
Tiempo (Baboon)	2.637±0.058	1.556±0.021	2.077±0.012	2.199±0.007	1.148±0.079

Tabla 37 Tiempo de convergencia para FISTA Regular, TPAL, WSO, SCO y OPT. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

FISTA TPAL tiene valores similares de la función de costo en el tiempo a FISTA OPT, sin embargo, el caso optimizado consigue además una mayor reducción en tamaño.

Además, similarmente al caso aleatorio, los métodos aplicados consisten de operaciones que requieren muy poco costo computacional, se obtiene menor tiempo para FISTA WSO y FISTA SCO comparado con FISTA sin optimizar en el mismo número de iteraciones. Adicionalmente el costo computacional disminuye al producirse una reducción inicial de tamaño en los datos estimados.

En las Figuras 31, 32, 33, 34 y 35 se observa las imágenes recuperadas con FISTA OPT.

Figura 31 Lena: Comparación entre la imagen original, observada y estimada con FISTA Optimizado.

Figura 32 Barbara: Comparación entre la imagen original, observada y estimada con FISTA Optimizado.

Figura 33 Goldhill: Comparación entre la imagen original, observada y estimada con FISTA Optimizado.

Figura 34 House: Comparación entre la imagen original, observada y estimada con FISTA Optimizado.

Figura 35 Baboon: Comparación entre la imagen original, observada y estimada con FISTA Optimizado.

Se compara además la calidad de las imágenes recuperadas, se obtienen los valores de media (negro) y varianza (azul) de PSNR y SSIM para los 5 casos evaluados en 20 iteraciones 100 veces en las Tablas 38 y 39.

	FISTA	FISTA TPAL	FISTA WSO	FISTA SCO	FISTA OPT
PSNR (Lena)	17.012±0.012	20.051±0.014	17.954±0.011	18.969±0.015	20.126±0.015
PSNR (Barbara)	16.764±0.010	19.601±0.010	17.626±0.012	18.521±0.014	19.662±0.010
PSNR (Goldhill)	17.033±0.017	19.971±0.017	17.976±0.020	18.988±0.022	20.042±0.017
PSNR (House)	16.938±0.011	19.479±0.011	17.832±0.010	18.763±0.011	19.529±0.011
PSNR (Baboon)	16.748±0.014	19.437±0.016	17.597±0.018	18.454±0.019	19.482±0.015

Tabla 38 Comparación de PSNR de las diferentes imágenes empleadas. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

	FISTA	FISTA TPAL	FISTA WSO	FISTA SCO	FISTA OPT
SSIM (Lena)	0.1762±0.0008	0.2504±0.0007	0.2058±0.0008	0.2406±0.0007	0.2525±0.0007
SSIM (Barbara)	0.3423±0.0004	0.4215±0.0006	0.3770±0.0004	0.4128±0.0006	0.4234±0.0006
SSIM (Goldhill)	0.1992±0.0011	0.2824±0.0012	0.2337±0.0012	0.2718±0.0013	0.2843±0.0012
SSIM (House)	0.3826±0.0008	0.4641±0.0012	0.4194±0.0009	0.4549±0.0011	0.4653±0.0012
SSIM (Baboon)	0.3680±0.0006	0.4560±0.0011	0.4086±0.0008	0.4462±0.0010	0.4567±0.0010

Tabla 39 Comparación de SSIM de las diferentes imágenes empleadas. Se lista la evolución de la función de costo (Ec. 6). Dicho problema se ejecutó 100 veces y (i) en color negro se lista el costo promedio, y (ii) en color azul la correspondiente desviación estándar.

Al ser los datos estimados iniciales aleatorios, se observa una reconstrucción considerablemente similar a la imagen original, esto se comprueba además con la reducción de la función de error. Adicionalmente, se busca recuperar la imagen con la menor cantidad de valores disminuyendo el tamaño de los datos.

Se comprueba la eficacia del algoritmo desarrollado al obtener mayores valores de PSNR y SSIM para el caso de FISTA Optimizado.

5 Conclusiones

Se ha desarrollado la lógica detrás de soluciones de problemas lineales inversos, en particular en la minimización de funciones convexas y en el método de primer orden: FISTA. En base a las características del algoritmo, se han evaluado técnicas de optimización que logren una convergencia acelerada. Con lo que se obtienen las siguientes conclusiones:

- El empleo de tamaño de paso dinámico, logra que el algoritmo converja en un menor número de iteraciones al adecuar el paso de la gradiente a la minimización, obteniendo reducciones más rápidas de error y tamaño. Se ha comprobado su eficacia tanto para datos aleatorios como imágenes.
- Los métodos de optimización de *Warm Start* y *Screening* logran una mejor solución inicial, con la que se obtiene convergencia en un menor número de iteraciones al poder discernir entre *features* y valores irrelevantes. Además, debido a la reducción de tamaño, se logra una menor utilización de recursos computacionales y por ende menor tiempo de procesamiento.
- En optimización de datos aleatorios, FISTA OPT (Optimizado) obtiene una mayor velocidad de convergencia para las tres métricas de comparación: tanto la función de error, regularización y costo.
- En imágenes, se consiguen valores similares de la función de costo en el tiempo para FISTA TPAL (tamaño de paso automático con límites) y FISTA OPT (Optimizado). Sin embargo, el caso optimizado converge en un menor número de iteraciones y logra una mayor disminución final de tamaño.

Se comprueba que los métodos de optimización planteados mejoran el desempeño de FISTA, mejorando su tasa de convergencia.

6 Recomendaciones

Al evaluar los métodos de optimización descritos, acontecieron ciertos imprevistos:

- Emplear un tamaño de paso automático puede llevar a que el algoritmo diverja. En casos que la gradiente llegue a tomar valores muy reducidos, puede generar tamaños de paso muy elevados y que se produzcan picos en las funciones de error, regularización y costo. Esto es solucionado adoptando límites superiores e inferiores para el tamaño de paso.

- Para *Warm Start*, incrementar excesivamente el término de regularización, o emplearlo en todas las iteraciones, puede llevar a que el algoritmo diverja. Debido a que la regularización en el algoritmo iterativo no considera el incremento de *Warm Start*, puede llegar a eliminar *features* relevantes, incrementando irreversiblemente la función de error, y por ende la función de costo.

- En *Screening*, definir valores elevados para λ_{max} logra una mejor tasa de convergencia. Sin embargo, también es posible disminuir su valor si conservar mayores niveles de detalle es más relevante que reducir el tamaño.

7 Trabajos Futuros

Debido a que es un algoritmo generalizado, es posible su adaptación en aplicaciones que empleen métodos de optimización basados en la minimización de funciones convexas, tales como *Total Variation* y *Compressed Sensing*.

Asimismo, resultaría significativo evaluar el impacto de FISTA Optimizado como etapa de preprocesamiento en algoritmos de inteligencia artificial.

8 Referencias

[1] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[2] R. Devakunchari, "Analysis on big data over the years," *International Journal of Scientific and Research Publications*, vol. 4, no. 1, 2014.

[3] H. B. Curry., "The method of steepest descent for non-linear minimization problems," *Quarterly of Applied Mathematics*, vol. 2, no. 3, pp. 258–261, 1944.

[4] A. N. Tikhonov and V. Y. Arsenin, *Solutions of ill-posed problems*. Washington, D.C.: John Wiley & Sons, New York: V. H. Winston & Sons, 1977. Translated from the Russian, Preface by translation editor Fritz John, Scripta Series in Mathematics.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, 01 2011.

[6] P. Combettes and V. Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Modeling Simulation - MULTISCALE MODEL SIMUL*, vol. 4, 01 2005.

[7] R. Shimmura and J. Suzuki, "Converting ADMM to a proximal gradient for efficient sparse estimation," *ArXiv*, vol. abs/2104.10911, 2021.

[8] T. Goldstein, C. Studer, and R. Baraniuk, "A field guide to forward-backward splitting with a FASTA implementation," *ArXiv*, vol. abs/1411.3406, 2016.

[9] Y. X. Yuan, "Step-sizes for the gradient method," *AMS/IP Studies in Advanced Mathematics*, vol. 42(2), 2008.

[10] E. T. Hale, W. Yin, and Y. Zhang, "Fixed-Point continuation for l1-Minimization: methodology and convergence," *SIAM J. Optimization*, vol. 19, no. 3, pp. 1107–1130, 2008.

[11] A. Raj, J. Olbrich, B. Gartner, B. Scholkopf, and M. Jaggi, "Screening rules for convex problems," 2016.

[12] L. E. Ghaoui, V. Viallon, and T. Rabbani, "Safe feature elimination in sparse supervised learning," *CoRR*, vol. abs/1009.4219, 2010.

[13] A. Beck and M. Teboulle, "Fast gradient based algorithms for constrained total variation image denoising and deblurring roblems," *Trans. Img. Proc.*, vol. 18, pp. 2419–â02434, Nov. 2009.

[14] E. Candes, Y. Eldar, D. Needell, and P. Randall, "Compressed sensing with coherent and redundant dictionaries," *Applied and Computational Harmonic Analysis*, vol. 31, pp. 59–73, 07 2011.

[15] A. Beck, *First-order methods in optimization*. MOS-SIAM Series on Optimization, Society for Industrial and Applied Mathematics, 1 ed., 2017.

[16] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," 2010.

[17] J. Nocedal and S. Wright, *Numerical optimization*. Springer series in operations research and financial engineering, New York, NY: Springer, 2. ed. ed., 2006.

[18] I. Daubechies, *Ten lectures on wavelets*. CBMS-NSF Regional Conference Series in Applied Mathematics, Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 1992.

[19] A. Cauchy, "Methode generale pour la resolution des systemes d'equations simultanees," *C.R. Acad. Sci. Paris*, vol. 25, pp. 536–538, 1847.

[20] H. Akaike, *On a Successive Transformation of Probability Distribution and Its Application to the Analysis of the Optimum Gradient Method*, pp. 37–52. New York, NY: Springer New York, 1998.

[21] J. Barzilai and J. M. Borwein, "Two-Point Step Size Gradient Methods," *IMA Journal of Numerical Analysis*, vol. 8, pp. 141–148, 01 1988.

[22] M. Raydan, "On the Barzilai and Borwein choice of steplength for the gradient method," *IMA Journal of Numerical Analysis*, vol. 13, pp. 321–326, 07 1993.

[23] I. Daubechies, M. Fornasier, and I. Loris, "Accelerated projected gradient method for linear inverse problems with sparsity constraints," *Journal of Fourier Analysis and Applications*, vol. 14, pp. 764–792, Sep 2008.

[24] C. R. Vogel, *Computational methods for inverse problems*, vol. 23 of *Frontiers in Applied Mathematics*. SIAM, Jan. 2002.

ANEXO

Funciones empleadas en el desarrollo de la tesis:

Función general de FISTA

[ ]:
```
%Función general de FISTA
function [stats, x, var] = general_fista(opt, var, b, x, L)

    %Cálculo de la primera estadística previo a computar el algoritmo
    Ax = Ax_fun(x, opt, var);
    z = Ax- b;
    stats.fx(1) = 0.5*sum(z(:).*z(:)); %Función de Error
    stats.l1(1) = norm(x.val.*var.c_mask,1); %Función de la norma l1 (tamaño)
    stats.F(1) = stats.fx(1) + opt.lambda.*stats.l1(1); %Función de Costo

    %Inicio de conteo de tiempo
    var.time(1) = 0;
    tic
    y = x;
    t = 1;

    %Creación del vector de lambdas para Warm Start
    if opt.warms
        var.lambda_vec_init = var.lambda_vec;
        var.lambda_vec_ws = opt.ws_inc*var.lambda_vec;
    end
    maskS = ones(length(x.val),1);

    %Reducción de tamaño por Screening
    if opt.screening
        var.mask_SC = screening_fun(b, opt, var, y);
        x.val = x.val.*var.mask_SC;
        y.val = y.val.*var.mask_SC;
    end

    %Primer paso de Cauchy
    i = 0;
    ATz = ATx_fun(z, opt, var);
    g = ATz;
```

```
var.g(1,:) = g.val;
var.alpha(1) = step_fun(y, g, opt, var, i);
alphaCi = var.alpha(1);

%Inicio del algoritmo iterativo
for i = 1:L

    %Warm Start
    if opt.warms
        var.lambda_vec =  warms_fun(opt, var,i);
    end

    %Cálculo de tamaño de paso de Cauchy de iter. previa para Yuan B
    if opt.stepT == 'YnB'
        ATz = ATx_fun(z, opt, var);
        g = ATz;
        alphaCi = 0;
        alphaCi = step_fun(z, y, g, opt, var, i, alphaCi);
    end

    %Guardar valores anteriores
    var.x_ant = x.val;
    var.g_ant = g.val;
    var.y_ant = y.val;

    %Shrinkage
    arg_p = y.val - var.alpha(i).*g.val;
    v = abs(arg_p);
    mask = v > var.alpha(i).*var.lambda_vec;
    mask = mask.*maskS;
    x.val = sign(arg_p).*(mask.*v - var.alpha(i).*var.lambda_vec);

    %Actualización de valores
    t_next = (1+(1+4*t^2)^(1/2))/2;
    y.val = x.val + ((t-1)/t_next).*(x.val-var.x_ant);
    t = t_next;

    %Gradiente
    Ax = Ax_fun(y, opt, var);
    z = Ax-b;
    ATz  = ATx_fun(z, opt, var);
    g = ATz;
    var.g(i+1,:) = g.val;

    %Tamano de paso BB1
    var.alpha(i+1) = step_fun(y, g, opt, var, i , alphaCi);
```

```
        %Cálculo de estadisticas
        stats.fx(i+1) = 0.5*sum(z(:).*z(:)); %Función de Error
        stats.l1(i+1) = norm(y.val.*var.c_mask,1); %Función de la norma l1
        stats.F(i+1) = stats.fx(i+1) + opt.lambda.*stats.l1(i+1); %F. de Costo
        maskS = ones(length(y.val),1);
        var.time(i+1) = toc;
    end
end
```

Función general de ISTA

[ ]:
```
%Función general de ISTA
function [stats, x, var] = general_ista(opt, var, b, x, L)

    %Cálculo de la primera estadística previo a computar el algoritmo
    Ax = Ax_fun(x, opt, var);
    z = Ax- b;
    stats.fx(1) = 0.5*sum(z(:).*z(:)); %Función de Error
    stats.l1(1) = norm(x.val.*var.c_mask,1); %Función de la norma l1 (tamaño)
    stats.F(1) = stats.fx(1) + opt.lambda.*stats.l1(1); %Función de Costo

    %Inicio de conteo de tiempo
    var.time(1) = 0;
    tic

    %Creación del vector de lambdas para Warm Start
    if opt.warms
        var.lambda_vec_init = var.lambda_vec;
        var.lambda_vec_ws = opt.ws_inc*var.lambda_vec;
    end
    maskS = ones(length(x.val),1);

    %Reducción de tamaño por Screening
    if opt.screening
        var.mask_SC = screening_fun(b, opt, var, x);
        x.val = x.val.*var.mask_SC;
    end

    %Primer paso de Cauchy
    i = 0;
    ATz = ATx_fun(z, opt, var);
    g = ATz;
    var.alpha = step_fun(x, g, opt, var, i);

    %Inicio del algoritmo iterativo
    for i = 1:L
```

```
        %Warm Start
        if opt.warms
            var.lambda_vec =  warms_fun(opt, var,i);
        end

        %Guardar valores anteriores
        var.x_ant = x.val;
        var.g_ant = g.val;

        %Shrinkage
        arg_p = x.val - var.alpha.*g.val;
        v = abs(arg_p);
        mask = v > var.alpha.*var.lambda_vec;
        mask = mask.*maskS;

        %Actualización de valores
        x.val = sign(arg_p).*(mask.*v - var.alpha.*var.lambda_vec);

        %Gradiente
        Ax = Ax_fun(x, opt, var);
        z = Ax-b;
        ATz  = ATx_fun(z, opt, var);
        g = ATz;

        %Tamano de paso BB1
        var.alpha = step_fun(x, g, opt, var, i);

        %Estadisticas
        stats.fx(i+1) = 0.5*sum(z(:).*z(:)); %Función de Error
        stats.l1(i+1) = norm(x.val.*var.c_mask,1);  %Función de la norma l1
        stats.F(i+1) = stats.fx(i+1) + opt.lambda.*stats.l1(i+1); %F. de Costo
        maskS = ones(length(x.val),1);
        var.time(i+1) = toc;

    end
end
```

Función Ax

```
[ ]: %Función A*x, empleada para la gradiente y las funciones de Error y Costo
function Ax = Ax_fun(z, opt, var)
    switch opt.dict

        %En el caso de datos sintéticos aleatorios, debido a que la matriz A
        %es conocida, se tiene multiplicación de la matriz A por el vector x
        case 'rand'
            Ax = var.A*z.val;

        %En el caso de imágenes, se toma la Transformada Discreta Wavelets
        %2D Inversa
        case 'wave'
            Ax = waverec2(z.val,z.size,opt.wname);
    end

end
```

Función $A^T x$

```
[ ]: %Función A'*x, empleada para el cálculo de la gradiente (A'*(A*x-b))
function ATx = ATx_fun(z, opt, var)
    switch opt.dict

        %En el caso de datos sintéticos aleatorios, debido a que la matriz A
        %es conocida, se tiene multiplicación de la matriz A' por el vector
        %z = A*x-b
        case 'rand'
            ATx.val = var.A'*z;

        %En el caso de imágenes, se toma la Transformada Discreta Wavelets 2D
        case 'wave'
            [ATx.val,ATx.size] = wavedec2(z,opt.n,opt.wname);
            ATx.val = ATx.val';
    end

end
```

Generación de datos iniciales

```
[ ]: %Función para la generación de datos iniciales y variables necesrias
function [x_orig, b, x, var] = data_gen(opt)
    switch opt.dict

        %Caso de datos sintéticos aleatorios
        case 'rand'
            %Matriz A aleatoria normalizada
```

```
            var.A = sqrt(1/opt.N)*randn(opt.N, opt.N);
            %Datos originales (vector de datos aleatorios)
            x_orig = randn(opt.N,1);
            %Generar datos originales que sean sparse
            ind = randperm(opt.N);
            R = 0.25*(opt.N);
            x_orig(ind(1:R)) = 0;
            %Datos observados b
            b = var.A*x_orig + opt.sigma*randn(opt.N,1);
            %Datos iniciales aleatorios
            x.val = randn(opt.N,1);
            %Vector de lambdas (necesario para shrinkage y Warm Start)
            var.c_mask = ones(opt.N,1);
            var.lambda_vec = opt.lambda*var.c_mask;

        %Caso de imágenes
        case 'wave'
            %Datos originales: lectura de imagen
            x_orig = imread(opt.sample_img);
            S = length(x_orig);
            x_orig = double(x_orig)./255.0;
            %Imagen observada: imágene original y suma de ruido Gaussiano
            b = x_orig;
            b = b + opt.sigma*randn(S,S);
            %Datos iniciales de las mismas dimensiones que la transformada
            %directa 2D de los datos observados
            [x.val,x.size] = wavedec2(b,opt.n,opt.wname);
            opt.N = length(x.val);
            x.val = randn(opt.N, 1);
            %Vector de lambdas necesario para shrinkage y Warm Start:
            %cantidad de coeficientes de aproximación
            cA_size = x.size(1,1)*x.size(1,2);
            %cantidad de coeficientes de detalle
            cD_size = sum(x.size(2:opt.n+1,1).*x.size(2:opt.n+1,2))*3;
            var.c_mask = [zeros(cA_size,1); ones(cD_size,1)];
            %lambda solo influye en los coeficientes de detalle
            var.lambda_vec = opt.lambda*var.c_mask;
    end
end
```

Función de tamaño de paso automático

[ ]:
```
function alpha = step_fun(x, g, opt, var, i, alphaCi)

    %Si se encuentra en la iteracion 0, se calcula el tamaño de paso de Cauchy
    if ((isequal(opt.stepT,'BB1') ) || (isequal(opt.stepT,'YnB') )) && i == 0
        Ag = Ax_fun(g, opt, var); %Cálculo de A*g
        alpha0 = norm(g.val)^2/norm(Ag)^2; %Tamaño de paso de Cauchy
        alpha0 = opt.step0_mult*alpha0; %Multiplicación por un coeficiente
        %Limites para el tamaño de paso  (opcional)
        if opt.steplim
            if alpha0 > opt.stepCH
                alpha0 = opt.stepCH ;
            end
            if alpha0 < opt.stepCL
                alpha0 = opt.stepCL;
            end
        end
    end

        switch opt.stepT

        %Tamaño de paso constante
        case 'cte'
        alpha = opt.alpha;

        %Tamaño de paso de Barzilai-Borwein
        case 'BB1'
        if i == 0
            alpha = alpha0; %Si es iteración 0, usar Cauchy
        else
            %Diferencia entre gradiente y valores de iteración actual y
            %pasada
            s = x.val - var.x_ant;
            y = g.val - var.g_ant;
            alpha = s'*y/norm(y)^2; %Tamaño de paso BB1
            alpha = opt.step_mult*alpha; %Multiplicación por un coeficiente
            %Limites para el tamaño de paso  (opcional)
            if opt.steplim
                if alpha > opt.stepH
                    alpha = opt.stepH ;
                end
                if alpha < opt.stepL
                    alpha = opt.stepL;
                end
            end
        end
```

```
        %Tamaño de paso de Yuan B
        case 'YnB'
        if i == 0
            alpha = alpha0;  %Si es iteración 0, usar Cauchy
        else
            %Intercalamiento  entre tamaño de paso de Cauchy y Yuan
            f = mod(i,3);
            if i == 3 ||  f
                %Tamaño de paso de Cauchy
                alpha = alpha0;
            else
                %Tamaño de paso de Yuan B
                s = x.val - var.x_ant;
                alpha = 2/((((1/alphaCi) -1/(alpha0))^2+(4*norm(g.val)^2)/
↪(norm(s)^2))^(1/2) + 1/alphaCi+1/alpha0);
            end
            alpha = opt.step_mult*alpha; %Multiplicación por un coeficiente
            %Limites para el tamaño de paso  (opcional)
            if opt.steplim
                if alpha >  opt.stepH
                    alpha =  opt.stepH ;
                end
                if alpha <  opt.stepL
                    alpha =  opt.stepL;
                end
            end
        end
    end
end
```

Función de Warm Start

[ ]:
```
%Función de Warm Start
function lambda_vec =  warms_fun(opt, var,i)

    if i > (opt.warms_iter)
        lambda_vec = var.lambda_vec_init; %vector inicial de lambdas
    else
         switch opt.wtype

            %Decrecimiento lineal hallado en base a dos puntos
            case 'line'
                lambda_vec = var.lambda_vec_ws*(((1-opt.ws_inc)*i+(opt.
↪warms_iter*opt.ws_inc+opt.ws_inc-1))/(opt.warms_iter*opt.ws_inc));

            %Decrecimiento cuadrático hallado en base al vertice
```

```
            %(derivada = 0) y otro punto
            case 'cuad'
                lambda_vec = var.lambda_vec_ws*(((1-opt.ws_inc)*i*i+(opt.
↪ws_inc-1)*2*i+(opt.warms_iter*opt.warms_iter*opt.ws_inc+1-opt.ws_inc))/(opt.
↪warms_iter*opt.warms_iter*opt.ws_inc));

            %Caso con lambda continuo, sin decrecimiento
            case 'cont'
                lambda_vec = var.lambda_vec_ws;
        end
    end
end
```

Función de Screening

[ ]:
```
%Función de Screening
function mask_SC =  screening_fun(b, opt, var, x)
    switch opt.dict

        %Caso de datos sintéticos aleatorios
        case 'rand'
            lambda_max = max(b'*var.A);%Calculo de lambda_max
            lambda = opt.screeningfeatures*lambda_max; %Porcentaje de
            %lambda_max a ser evaluado (diferentes intensidades de screening)
            normb = norm(b); %norma l_2 de b
            normAi = 0.0394*vecnorm(var.A,1)'; %aproximacion de la norma l2
            %de las columnas de A  en base a la norma l1 (menor tiempo de
            %computo)
            absbA = abs(b'*var.A)';
            pk = (normb.*normAi+absbA)./(normb.*normAi+lambda_max); %Cálculo
            %del vector pk
            pklambdaMax = pk*lambda_max; %Vector pk*lambda_max
            %Comparacion de lambda y pk*lambda_max
            mask_SC = lambda < pklambdaMax ; %máscara lógica de 0's y 1's
            mask_SC = double(mask_SC); %máscara de doubles

        %Caso de imágenes
        case 'wave'
            c = x.val; %valores de la transformada wavelet de la imágen
            s = x.size; %tamaño de los niveles de la transformada
            c_size = length(c); %tamaño de todos los coeficientes
            cA_size = s(1,1)*s(1,2); %tamaño de los coeficientes aproximados
            cD_size = c_size - cA_size; %tamaño de los coeficientes de detalle
            c_mask = [zeros(cA_size,1); ones(cD_size,1)]; %máscara de 0's
            %(coeficientes aproximados) y 1's (coeficientes de detalle)
            c = c_mask.*c; %se evaluan solo los coeficiente de detalle
            lambda_max = max(c); %maximo valor de los coeficientes de detalle
```

```
            lambda = opt.screeningfeatures*lambda_max;  %Porcentaje de
            %lambda_max a ser evaluado (diferentes intensidades de screening)
            xk2 = (norm(c)/norm(b))/length(c); %aproximación del cálculo de
            %la norma de las columnas de A
            y2 = norm(b) ;
            pk = (y2*xk2+ abs(c))/(y2*xk2+ lambda_max); %Cálculo del vector pk
            pklambdaMax = pk*lambda_max; %Vector pk*lambda_max
            %Comparacion de lambda y pk*lambda_max
            mask_SC = lambda < pklambdaMax ;  %máscara lógica de 0's y 1's
            mask_SC = double(mask_SC); %máscara de doubles
            mask_SC = mask_SC .*c_mask; %máscara que solo toma los
            %coeficientes de detalle)
            mask_SC(1:cA_size) = ones(cA_size, 1); %máscara decoefientes
            %aproximados sin cambiar (1's)
    end
end
```

Caso de datos aleatorios generados sintéticamente

[ ]:
```
clc
clear all
close all

%Opciones de selección
opt.dict = 'rand';      %Diccionario 'rand' 'wave'
opt.stepT = 'cte';      %Nombre de tamaño de paso 'cte' 'BB1' 'YnB'
opt.wtype = 'cont';     %Estrategia de Warm Start 'cont' 'line' 'cuad'
opt.wname = 'db4';      %Tipo de Transformada Wavelet 'db1' 'db4' 'bior4.4'

%Contasntes lógicas
opt.steplim = 0;            %Activación de límites para el tamaño de paso
opt.warms = 0;          %Activación de Warm Start
opt.screening = 0;      %Activación de Screening

%Parámetros
opt.N = 1000;           %tamaño de los valores a evaluar
L = 50;                 %número de iteraciones
opt.alpha = 0.1;        %tamaño de paso
opt.lambda = 0.1;       %parámetro de regularización
opt.sigma = 1.0*10^-2;  %ruido aditivo
opt.ws_inc = 2;         %incremento para Warm Start
opt.warms_iter = 0.1*L; %iteración hasta la cual se evalua Warm Start
opt.screeningfeatures = 1; %intensidad de Screening
opt.n = 3;              %nivel de evaluación de la Transformada Wavelets

%Generacion de datos y variables iniciales
[x_orig, b, x_init, var_init] = data_gen(opt);
```

```
%Casos

%ISTA
x = x_init;
var = var_init;
[stats_ista, x_ista, var_ista] = general_ista(opt, var, b, x, L);

%FISTA
x = x_init;
var = var_init;
[stats_fista, x_fista, var_fista] = general_fista(opt, var, b, x, L);

%FISTA TPAL (tamaño de paso automático con límites)
opt.stepT = 'BB1';
opt.steplim = 1;
opt.stepCH = 0.2;       %valor máximo de tamaño de paso de Cauchy
opt.stepCL = 0.1;       %valor mínimo de tamaño de paso de Caichy
opt.stepH = 0.4 ;       %valor máximo de tamaño de paso automático
opt.stepL = 0.13;       %valor mínimo de tamaño de paso automático
opt.step0_mult = 0.35;  %coeficiente que multiplica al t. de paso de Cauchy
opt.step_mult = 0.35;   %coeficiente que multiplica al t. de paso automático
x = x_init;
var = var_init;
[stats_fista_auto, x_fista_auto, var_fista_auto] = general_fista(opt, var, b,␣
 ↪x, L);

%FISTA WS suave
opt.warms = 1;
opt.stepT = 'cte';
opt.ws_inc = 2;
x = x_init;
var = var_init;
[stats_fista_wsL, x_fista_wsL, var_fista_wsL] = general_fista(opt, var, b, x,␣
 ↪L);

%FISTA WS fuerte
opt.ws_inc = 5;
x = x_init;
var = var_init;
[stats_fista_wsM, x_fista_wsM, var_fista_wsM] = general_fista(opt, var, b, x,␣
 ↪L);

%FISTA WS excesivo
opt.ws_inc = 10;
x = x_init;
var = var_init;
```

```
[stats_fista_wsH, x_fista_wsH, var_fista_wsH] = general_fista(opt, var, b, x,␣
↪L);

%FISTA SC suave
opt.warms = 0;
opt.screening = 1;
opt.screeningfeatures = 0.89;
x = x_init;
var = var_init;
[stats_fista_scL, x_fista_sL, var_fista_scL] = general_fista(opt, var, b, x, L);

%FISTA SC medio
opt.screeningfeatures = 0.91;
x = x_init;
var = var_init;
[stats_fista_scM, x_fista_scM, var_fista_scM] = general_fista(opt, var, b, x,␣
↪L);

%FISTA SC fuerte
opt.screeningfeatures = 1;
x = x_init;
var = var_init;
[stats_fista_scH, x_fista_scH, var_fista_scH] = general_fista(opt, var, b, x,␣
↪L);

%FISTA OPT
opt.warms = 1;
opt.ws_inc = 2;
opt.screening = 1;
opt.screeningfeatures = 1;
opt.stepT = 'BB1';
opt.steplim = 1;
opt.step0_mult = 0.35;
opt.step_mult = 0.35;
opt.stepCH = 0.2;
opt.stepCL = 0.1;
opt.stepH = 0.25 ;
opt.stepL = 0.12;
x = x_init;
var = var_init;
[stats_fista_all, x_fista_all, var_fista_all] = general_fista(opt, var, b, x,␣
↪L);

%Graficos

%Comparación entre FISTA Regular, TPAL, WSO, SCO y OPT
figure(1)
```

```
subplot(1,3,1), semilogy(stats_fista.fx,'LineWidth',3)
hold on
semilogy(stats_fista_auto.fx,'LineWidth',3)
semilogy(stats_fista_wsM.fx,'LineWidth',3)
semilogy(stats_fista_scH.fx,'LineWidth',3)
semilogy(stats_fista_all.fx,'LineWidth',3)
set(gca,'FontSize',24)
grid;
xlabel('Iteraciones'), ylabel('Funcion de error')
xlim([1 50])
subplot(1,3,2), semilogy(stats_fista.l1,'LineWidth',3)
hold on
semilogy(stats_fista_auto.l1,'LineWidth',3)
semilogy(stats_fista_wsM.l1,'LineWidth',3)
semilogy(stats_fista_scH.l1,'LineWidth',3)
semilogy(stats_fista_all.l1,'LineWidth',3)
set(gca,'FontSize',24)
grid;
xlabel('Iteraciones'), ylabel('Norma l1 de x')
xlim([1 50])
subplot(1,3,3), semilogy(stats_fista.F,'LineWidth',3)
hold on
semilogy(stats_fista_auto.F,'LineWidth',3)
semilogy(stats_fista_wsM.F,'LineWidth',3)
semilogy(stats_fista_scH.F,'LineWidth',3)
semilogy(stats_fista_all.F,'LineWidth',3)
set(gca,'FontSize',24)
legend('FISTA', 'FISTA TPAL','FISTA WSO',  'FISTA SCO', 'FISTA OPT', ␣
↪'FontSize',  26)
grid;
xlabel('Iteraciones'), ylabel('Funcion de costo F(x)')
xlim([1 50])

%Comparación de tiempo para FISTA Regular, TPAL, WSO, SCO y OPT
figure(2)
semilogy(var_fista.time(1:L), stats_fista.F(1:L),'LineWidth',3)
hold on
semilogy(var_fista_auto.time(1:L), stats_fista_auto.F(1:L),'LineWidth',3)
semilogy(var_fista_wsM.time(1:L), stats_fista_wsM.F(1:L),'LineWidth',3)
semilogy(var_fista_scH.time(1:L), stats_fista_scH.F(1:L),'LineWidth',3)
semilogy(var_fista_all.time(1:L), stats_fista_all.F(1:L),'LineWidth',3)
set(gca,'FontSize',24)
legend('FISTA', 'FISTA TPAL','FISTA WSO',  'FISTA SCO', 'FISTA OPT', ␣
↪'FontSize',  26)
grid;
xlabel('Tiempo (s)'), ylabel('Funcion de Costo F(x)')
xlim([0 0.11])
```

Caso de imágenes con la Transformada Wavelets

[ ]:
```
clc
clear all
close all

%Opciones de selección
opt.dict = 'wave';      %Diccionario 'rand' 'wave'
opt.stepT = 'cte';      %Nombre de tamaño de paso 'cte' 'BB1' 'YnB'
opt.wtype = 'cont';     %Estrategia de Warm Start 'cont' 'line' 'cuad'
opt.wname = 'db4';      %Tipo de Transformada Wavelet 'db1' 'db4' 'bior4.4'
opt.sample_img = 'baboon.gif'; % 'lena.bmp' 'barbara.bmp' 'goldhill.gif' 'house.
 ↪gif' 'baboon.gif'

%Constantes lógicas
opt.steplim = 0;        %Activación de límites para el tamaño de paso
opt.warms = 0;          %Activación de Warm Start
opt.screening = 0;      %Activación de Screening

%Parámetros
L = 20;                 %número de iteraciones
opt.alpha = 0.05;       %tamaño de paso
opt.lambda = 80*10^-3;  %parámetro de regularización
opt.sigma = 1.5*10^-1;  %ruido aditivo
opt.ws_inc = 2;         %incremento para Warm Start
opt.warms_iter = 5;     %iteración hasta la cual se evalua Warm Start
opt.screeningfeatures = 1; %intensidad de Screening
opt.n = 3;              %nivel de evaluación de la Transformada Wavelets

%Generacion de datos y variables iniciales
[x_orig, b, x_init, var_init] = data_gen(opt);

%Casos

%ISTA
x = x_init;
var = var_init;
[stats_ista, x_ista, var_ista] = general_ista(opt, var, b, x, L);

%FISTA
x = x_init;
var = var_init;
[stats_fista, x_fista, var_fista] = general_fista(opt, var, b, x, L);

%FISTA TPAL (tamaño de paso automático con límites)
opt.stepT = 'BB1';
opt.steplim = 1;
```

```
opt.stepCH = 0.2;        %valor máximo de tamaño de paso de Cauchy
opt.stepCL = 0.1;        %valor mínimo de tamaño de paso de Caichy
opt.stepH = 0.3 ;        %valor máximo de tamaño de paso automático
opt.stepL = 0.2;         %valor mínimo de tamaño de paso automático
opt.step0_mult = 0.1;    %coeficiente que multiplica al t. de paso de Cauchy
opt.step_mult = 0.3;     %coeficiente que multiplica al t. de paso automático
x = x_init;
var = var_init;
[stats_fista_auto, x_fista_auto, var_fista_auto] = general_fista(opt, var, b,␣
 ↪x, L);

%FISTA WS suave
opt.warms = 1;
opt.stepT = 'cte';
opt.ws_inc = 2;
x = x_init;
var = var_init;
[stats_fista_wsL, x_fista_wsL, var_fista_wsL] = general_fista(opt, var, b, x,␣
 ↪L);

%FISTA WS fuerte
opt.ws_inc = 10;
x = x_init;
var = var_init;
[stats_fista_wsM, x_fista_wsM, var_fista_wsM] = general_fista(opt, var, b, x,␣
 ↪L);

%FISTA WS excesivo
opt.ws_inc = 20;
x = x_init;
var = var_init;
[stats_fista_wsH, x_fista_wsH, var_fista_wsH] = general_fista(opt, var, b, x,␣
 ↪L);

%FISTA SC suave
opt.warms = 0;
opt.screening = 1;
opt.screeningfeatures = 0.3;
x = x_init;
var = var_init;
[stats_fista_scL, x_fista_sL, var_fista_scL] = general_fista(opt, var, b, x, L);

%%FISTA SC medio
opt.screeningfeatures = 0.5;
x = x_init;
var = var_init;
```

```
[stats_fista_scM, x_fista_scM, var_fista_scM] = general_fista(opt, var, b, x,␣
↪L);

%FISTA SC fuerte
opt.screeningfeatures = 1;
x = x_init;
var = var_init;
[stats_fista_scH, x_fista_scH, var_fista_scH] = general_fista(opt, var, b, x,␣
↪L);

%FISTA OPT
opt.warms = 1;
opt.ws_inc = 1.5;
opt.screening = 1;
opt.screeningfeatures = 1;
opt.stepT = 'BB1';
opt.steplim = 1;
opt.step0_mult = 0.1;
opt.step_mult = 0.3;
opt.stepCH =   0.2;
opt.stepCL =   0.1;
opt.stepH =  0.3;
opt.stepL =   0.2;
x = x_init;
var = var_init;
[stats_fista_all, x_fista_all, var_fista_all] = general_fista(opt, var, b, x,␣
↪L);

%Graficos

%Comparación entre FISTA Regular, TPAL, WSO, SCO y OPT
figure(1)
subplot(1,3,1), plot(stats_fista.fx,'LineWidth',3)
hold on
plot(stats_fista_auto.fx,'LineWidth',3)
plot(stats_fista_wsM.fx,'LineWidth',3)
plot(stats_fista_scH.fx,'LineWidth',3)
plot(stats_fista_all.fx,'LineWidth',3)
set(gca,'FontSize',24)
grid;
xlabel('Iteraciones'), ylabel('Funcion de error')
xlim([1 20])
subplot(1,3,2), plot(stats_fista.l1,'LineWidth',3)
hold on
plot(stats_fista_auto.l1,'LineWidth',3)
plot(stats_fista_wsM.l1,'LineWidth',3)
plot(stats_fista_scH.l1,'LineWidth',3)
```

```
plot(stats_fista_all.l1,'LineWidth',3)
set(gca,'FontSize',24)
grid;
xlabel('Iteraciones'), ylabel('Norma l1 de x')
xlim([1 20])
subplot(1,3,3), plot(stats_fista.F,'LineWidth',3)
hold on
plot(stats_fista_auto.F,'LineWidth',3)
plot(stats_fista_wsM.F,'LineWidth',3)
plot(stats_fista_scH.F,'LineWidth',3)
plot(stats_fista_all.F,'LineWidth',3)
set(gca,'FontSize',24)
legend('FISTA', 'FISTA TPAL','FISTA WSO',  'FISTA SCO', 'FISTA OPT', ␣
↪'FontSize',  26)
grid;
xlabel('Iteraciones'), ylabel('Funcion de costo F(x)')
xlim([1 20])

%Comparación de tiempo para FISTA Regular, TPAL, WSO, SCO y OPT
figure(2)
plot(var_fista.time(1:L), stats_fista.F(1:L),'LineWidth',3)
hold on
plot(var_fista_auto.time(1:L), stats_fista_auto.F(1:L),'LineWidth',3)
plot(var_fista_wsM.time(1:L), stats_fista_wsM.F(1:L),'LineWidth',3)
plot(var_fista_scH.time(1:L), stats_fista_scH.F(1:L),'LineWidth',3)
plot(var_fista_all.time(1:L), stats_fista_all.F(1:L),'LineWidth',3)
set(gca,'FontSize',24)
legend('FISTA', 'FISTA TPAL','FISTA WSO',  'FISTA SCO', 'FISTA OPT', ␣
↪'FontSize',  26)
grid;
xlabel('Tiempo (s)'), ylabel('Funcion de Costo F(x)')
xlim([0 2.7])
```