Revista Digital: Matemática, Educación e Internet
ISSN: 1659-0643
revistadigitalmatematica@itcr.ac.cr
Instituto Tecnológico de Costa Rica
Costa Rica

Vázquez Mourazos, Manuel
Deducción de un nuevo algoritmo para la resolución de problemas de optimización con restricciones
Revista Digital: Matemática, Educación e Internet, vol. 22, núm. 1, 2021, -Marzo, pp. 1-19
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica

DOI: https://doi.org/10.18845/rdmei.v22i1.5730

Revista digital
Matemática, Educación e Internet
https://tecdigital.tec.ac.cr/revistamatematica/
Vol 22, No 1. Agosto – Marzo, 2021 ISSN 1659 -0643

Deducción de un nuevo algoritmo para la resolución de problemas de optimización con restricciones

Deduction of a new algorithm for solving constrained optimization problems

Manuel Vázquez Mourazos
manuelvazquezmourazos@gmail.com
Escuela Waldorf Meniñeiros
España (Melide, A Coruña)

Recibido: 13 setiembre 2020 Aceptado: 10 mayo 2021

Resumen: Con el objetivo de aportar una nueva visión en la resolución de algunos problemas de optimización, este artículo versará acerca de la deducción de un nuevo método numérico en optimización. Para ello, se planteará una modelización genérica de los problemas que se pretenden resolver. De este modo, se introducirán los preliminares necesarios para poder proceder con la deducción que posteriormente se hará. En este sentido, el método planteado se enmarcará dentro de los métodos de direcciones factibles, por lo que se introducirán conceptos topológicos sencillos (relacionados con el conjunto admisible) y definiciones relacionadas con las direcciones factibles y de descenso que se podrán considerar. Posteriormente se procederá con la deducción del algoritmo que presenta este artículo, procurando que el lector pueda observar el procedimiento de construcción del mismo. Además, se introducirán los aspectos computacionales del método y se propondrán una serie de códigos que permitirán implementar el algoritmo en un lenguaje de programación, en el caso de este artículo se utilizará MatLab. Finalmente se mostrarán los resultados obtenidos al ejecutar el método junto con otros métodos ya conocidos para poder comparar los resultados. De esta forma, se finalizará aportando las conclusiones más relevantes y las futuras líneas de investigación para mejorar el método presentado.

Palabras Clave: optimización con restricciones, programación matemática, dirección de descenso, dirección factible, discretización, método de Gram-Schmidt, método de direcciones factibles

Abstract: With the aim of providing a new vision in solving some optimization problems, this article will deal with the deduction of a new numerical method in optimization. To do this, a generic modeling of the problems to be solved will be proposed. In this way, the necessary preliminaries will be introduced to be able to proceed with the deduction that will be made later. In this sense, the proposed method will be framed within the feasible directions methods, for which simple topological concepts (related to the admissible set) and definitions related to the feasible and descent directions that may be considered will be introduced. Subsequently, we will proceed with the deduction of the algorithm presented in this article, ensuring that the reader can observe the procedure for its construction. In addition, the computational aspects of the method will be introduced and a series of codes will be proposed that will allow the algorithm to be implemented in a programming language. In the case of this article, MatLab will be used. Finally, the results obtained when executing the method will be

shown together with other methods already known to be able to compare the results obtained. In this way, it will be concluded by providing the most relevant conclusions and future lines of research to improve the presented method.

Keywords: constrained optimization, mathematical programming, descent direction, feasible direction, discretization, GramSchmidt method, feasible direction methods

1. Introducción

La optimización es una disciplina matemática estudiada desde hace tiempo. Importantes matemáticos como Pierre de Fermat (1601-1665) en su memoria Methodus ad disquirendam maximan et minimam (escrita sobre 1629 pero publicada en 1679 después de su muerte por su hijo Samuel), ya estableció reglas para el cálculo de máximos y mínimos. También Joseph Louis Lagrange (1736-1813) en su publicación Mécanique Analytique (1788-89), fue capaz de hallar fórmulas basadas en el cálculo que permitieron caracterizar los valores óptimos que optimizaban ciertos problemas de optimización. Otros, dieron solución a problemas clásicos que ya se enmarcaban dentro del contexto de la optimización, como Leonhard Euler (1707-1783), con su famoso problema de los siete puentes de Königsberg. Además, cabe resaltar las aportaciones de Isaac Newton (1642-1727) y Johann Carl Friedrich Gauss (1777-1855) los cuales propusieron métodos iterativos que convergiesen a un óptimo.

Posteriormente, a mediados del siglo XX se resolvieron problemas más complejos dentro de esta disciplina. En esta época, se produjo un impulso del estudio de estos problemas; así, se puede consultar [3, pags. 326 - 332] para comprobar la aplicabilidad de los mismos. Además, desde el punto de vista matemático, la resolución de estos problemas resulta especialmente interesante al requerir de distintos conocimientos matemáticos como la topología, el análisis o la matemática computacional. Este hecho, se puede apreciar en [5], ya que en esta obra se puede observar la optimización desde un punto de vista teórico, estudiando casos especiales que ofrecen situaciones idílicas en las que es fácil calcular un óptimo.

Hogaño, debido a la amplia (y creciente) uso de la optimización en ciencia, ingeniería, economía e industria, tal y como se comenta en el prefacio de [12], se hace esencial el estudio de nuevos algoritmos que resuelvan este tipo de problemas. De hecho, hoy en día todas las empresas desean utilizar la optimización en sus procesos de fabricación, desde ahorrar los costes hasta maximizar los beneficios obtenidos. El estudio e implicación de esta disciplina en otras ramas del conocimiento, se puede ver en estudios como los que se incluyen en [7], [6], [9] o [15].

En este artículo se presentará la deducción de un nuevo algoritmo de resolución en los problemas de optimización en varias variables con restricciones. Para ello se realizará una reseña de los resultados más importantes que se utilizarán de forma reiterada. En concreto, se introducirá la definición genérica del problema de estudio, junto con definiciones topológicas que permitan caracterizar el conjunto admisible y las direcciones de descenso, las cuales cobrarán gran importancia en el algoritmo que se planteará. A continuación, se introducirá la deducción del método que presenta este artículo. En esta sección se propondrá la construcción de una sucesión que converja al valor óptimo del problema.

Para ello, se propondrá una idea intuitiva, cuya eficacia quedará entredicho al plantear un contraejemplo, lo que permitirá visualizar una carencia en este sentido. Luego, a partir de la visión del contraejemplo se introducirá una definición que permita solventar la dificultad observada. A partir de este punto, se incluirán una serie conceptos como las coordenadas esféricas en un espacio n-dimensional o el método de Gram-Schmidt para poder exponer un algoritmo aplicable. Finalmente, se estudiarán los aspectos computacionales del algoritmo para poder implementarlo en un lenguaje de programación numérica, concretamente se utilizará MatLab. De este modo, se hará una serie de ejecuciones para comprobar el funcionamiento del método planteado y se comparará con las soluciones exactas a los problemas resueltos y con los resultados obtenidos por otros métodos ya conocidos.

Con este artículo se pretende aportar una nueva visión en la resolución de algunos problemas de optimización. A partir del método propuesto, este queda a objeto de estudio del lector con vistas a una posible mejora en sus aspectos computacionales u otros que se consideren.

Notación 1

Antes de comenzar con la redacción, se hacen las siguientes aclaraciones correspondientes a la notación que se empleará en lo sucesivo:

1. A lo largo del artículo se emplearán escalares, vectores y matrices de forma frecuente. Para diferenciarlos, se escribirá a los vectores en letra negrita minúscula (por ejemplo $\mathbf{v}$), a las matrices por letra negrita mayúscula (por ejemplo $\mathbf{M}$) y en letra normal minúscula a los escalares (por ejemplo k).
2. El empleo de los vectores será muy reiterado. Dado un vector $\mathbf{v}$, si se desea referirse a su componente i-ésima, esta se denotará en letra normal minúscula con el subíndice, es decir, v_i.
3. En una sucesión de vectores, se denotará al vector k-ésimo de la sucesión con la siguiente escritura: $\mathbf{v}^{(k)}$.
4. A los conjuntos se les denotará en letra normal mayúscula, por ejemplo S. El término $\text{Int}(S)$ se refiere al interior del conjunto y ∂S para denotar a la frontera del conjunto.
5. Dada una función $f : \mathbb{R}^n \to \mathbb{R}$, se denotará por $\nabla f(\mathbf{x})$ al valor del gradiente de la función f evaluado en el vector (o también se denominará por punto) $\mathbf{x}$.
6. Si $f(x)$ y $g(x)$ son funciones definidas en un entorno de un punto x_0, entonces $f = o(g)$ cuando $x \to x_0$, si y sólo si, para todo $\varepsilon > 0$ se tiene que $|f(x)| < \varepsilon|g(x)|$ para todo x en un entorno de x_0.
7. Se utilizará el concepto de la esfera unitaria n-dimensional, que se denotará por $\mathbb{S}^n$ y se definirá del siguiente modo:

$$\mathbb{S}^n = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\| = 1\}$$

2. Preliminares

Antes de comenzar con la deducción del método que se expondrá en este artículo, se hará una recopilación de preliminares que se utilizarán con cierta frecuencia. Para la realización de estos preliminares se consultaron las referencias: [2], [12], [16], [17], [13], las cuales se pueden aportar una mayor ampliación de los contenidos aquí mostrados.

En este artículo se considerará el problema (1),

$$\begin{aligned} &\min_{\mathbf{v}\in\mathbb{R}^n} \quad J(\mathbf{v}) \\ &\text{Sujeto a:} \\ &\qquad \varphi_i(\mathbf{v}) \leq 0, \quad i = 1, \ldots, m\,. \end{aligned} \tag{1}$$

Este problema pretende minimizar la función $J(\mathbf{v})$, denominada *funcional objetivo*, en un subconjunto de su dominio, el cual se denominará *conjunto admisible* o *factible*. Según las restricciones, $\varphi_i(\mathbf{v})$, el conjunto admisible quedaría definido del siguiente modo:

$$S := \{\mathbf{v} \in \mathbb{R}^n : \varphi_i(\mathbf{v}) \leq 0, \quad i = 1, \ldots, m\}\,.$$

De esta manera, el problema inicial (1) se podría reescribir considerando la notación mostrada en (2),

$$\min_{\mathbf{v}\in S} J(\mathbf{v})\,. \tag{2}$$

Definición 1

Dado un funcional $J:\mathbb{R}^n\to\mathbb{R}$ y un conjunto admisible S, se dice que $\mathbf{u}$ es solución al problema (1) si se verifica que existe un $\varepsilon>0$ tal que,

$$J(\mathbf{u})\le J(\mathbf{v}),\qquad \forall \mathbf{v}\in S\cap B(\mathbf{u},\varepsilon)\ ,$$

donde $B(\mathbf{u},\varepsilon)$ denota a la bola de centro $\mathbf{u}$ y radio ε.

Para resolver el problema (2) se cuentan con múltiples opciones de algoritmos. Algunos de ellos se basan en la creación de una función auxiliar cuyo mínimo en todo su dominio coincida con el mínimo del problema de estudio, otros se caracterizan por la transformación del problema (2) en un problema cuya solución se pueda calcular de un modo más sencillo (como los métodos SQP). Sin embargo, en este artículo se presentará un método que tiene en consideración el concepto de dirección factible, se podría consultar [17, Cap. XI] para estudiar métodos inscritos en esta disciplina. Por este motivo, a continuación se introducirán una serie de definiciones y resultados que serán de enorme importancia para la deducción del algoritmo final.

Definición 2

Dado un punto $\mathbf{v}\in S$ y $\mathbf{d}\in\mathbb{R}^n$ no nulo, si existe un $\delta>0$ de tal forma que $\mathbf{v}+t\mathbf{d}\in S$, para todo $t\in[0,\delta]$, entonces se dice que $\mathbf{d}$ es una *dirección factible* de S en $\mathbf{v}$. Además, el conjunto de dichas direcciones factibles se denotará como:

$$DF(\mathbf{v},S):=\left\{\mathbf{d}\in\mathbb{R}^n:\exists\delta>0 \ \text{ tal que }\ \mathbf{v}+t\mathbf{d}\in S;\forall t\in[0,\delta]\right\}\ .$$

Observación 1

Para profundizar en la visión geométrica del concepto de dirección factible, se puede consultar [8, pags. 387-393].

Definición 3

Dado un punto $\mathbf{v}\in S$, se define el conjunto de restricciones activas de $\mathbf{v}$ como sigue:

$$I(\mathbf{v}):=\{i\in\{1,\dots,m\}:\varphi_i(\mathbf{v})=0\}\ .$$

La Definición 2 hace referencia a aquellas direcciones, a partir de las cuales, se puede desplazar una cierta distancia, desde un punto admisible, permaneciendo en el conjunto factible. En la Figura 1 se muestran algunos ejemplos. En ella se vislumbra un conjunto admisible (el conjunto rayado) y desde algunos puntos se dispersan una serie de direcciones que permiten desplazarse a través de ellas permaneciendo en el conjunto.

En este artículo, el método que se planteará utilizará las direcciones factibles como punto de partida. Esta idea se inscribe en aquellos métodos que se conocen con el nombre de *métodos de direcciones factibles*. La idea consiste en partir de un punto cualquiera del conjunto admisible y, a partir de él,

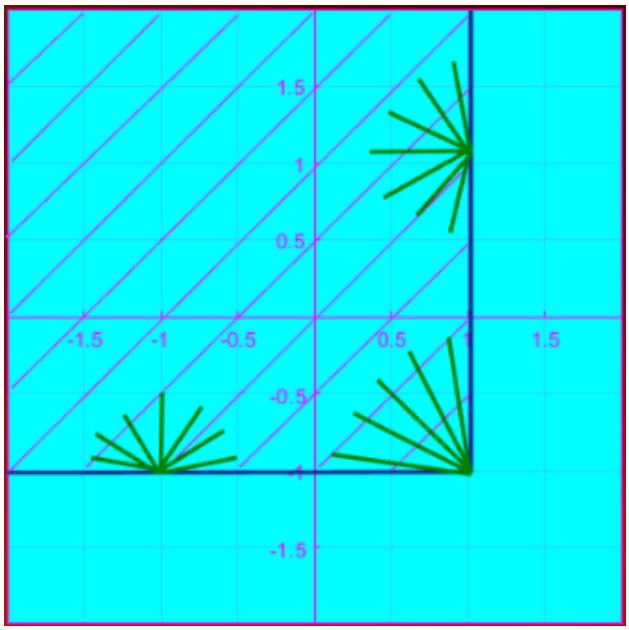


Figura 1: Ejemplos de direcciones factibles.

moverse a través de direcciones factibles que permitan descender el valor del funcional. Para ello, es necesario establecer la Definición 4.

Definición 4

Se considera una función $J : \mathbb{R}^n \to \mathbb{R}$ diferenciable en todo su dominio. Si existe un vector $\mathbf{d} \in \mathbb{R}^n$ tal que,

$$J(\mathbf{v} + t\mathbf{d}) < J(\mathbf{v}), \text{ para algún } t \in \mathbb{R},$$

entonces $\mathbf{d}$ es una *dirección de descenso* de J en el punto $\mathbf{v}$.

La Definición 4 caracteriza a las direcciones correspondientes a un punto dado que permiten descender el valor del funcional. Este hecho se puede comprobar utilizando la fórmula de Taylor. Pues bien, partiendo de un punto arbitrario $\mathbf{v} \in S$ se puede desarrollar la expresión (3),

$$J(\mathbf{v} + t\mathbf{d}) = J(\mathbf{v}) + t\, \nabla J(\mathbf{v})^T \mathbf{d} + o(t)\,. \tag{3}$$

Así, es fácil comprobar que,

$$\exists \delta > 0 \ \text{ tal que } \ J(\mathbf{v} + t\mathbf{d}) < J(\mathbf{v})\,, \ \ \forall t \in (0, \delta)\,, \tag{4}$$

si y sólo si, $\mathbf{d}$ es una dirección de descenso de J correspondiente al punto $\mathbf{v}$.

Por consiguiente, dado un punto factible cualquiera, la idea que se enmarca dentro de los métodos de direcciones factibles, es la de moverse dentro del conjunto admisible a través de direcciones de descenso que sean factibles. De este modo, una vez que se obtenga un punto para el que no existen direcciones de descenso que sean admisibles, se habrá conseguido un mínimo al problema de partida.

En la Figura 2 se muestra un ejemplo del hecho comentado. En la gráfica se aprecian las curvas de nivel del funcional objetivo $J(\mathbf{v}) = 1.8^{-1.5\sqrt{v_1^2+v_2^2}} \cdot \sin(v_1)\cos(v_2)$ y una elipse que delimita el conjunto admisible. El mínimo del problema se sitúa en el origen, el punto marcado en la gráfica, y de él salen distintas direcciones que delimitan el cono de las direcciones de descenso en el punto. Sin embargo, ninguna de esas direcciones permite permanecer en el conjunto admisible, por lo que no existen direcciones factibles de descenso.

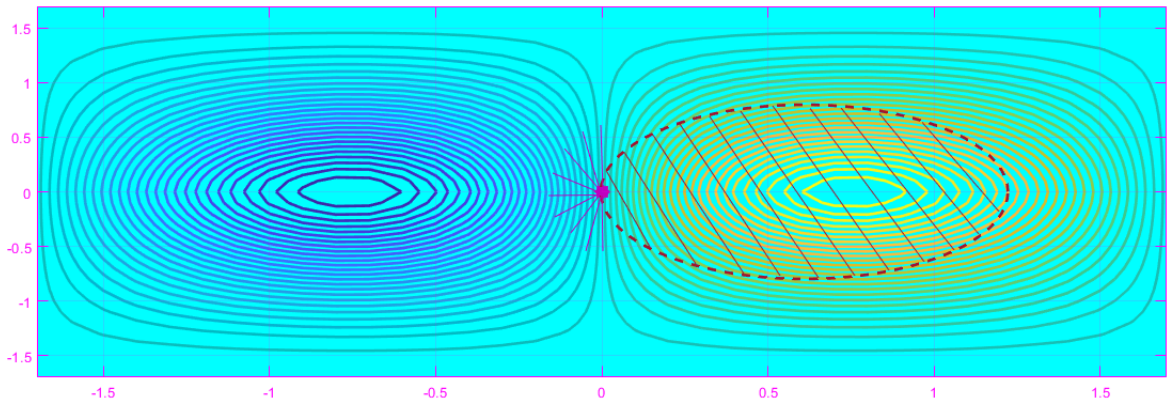


Figura 2: Ejemplo de imposibilidad de descenso a través de direcciones de descenso dentro de un conjunto factible considerando el funcional objetivo $J(\mathbf{v}) = 1.8^{-1.5\sqrt{v_1^2+v_2^2}} \cdot \sin(v_1)\cos(v_2)$.

3. Deducción del método

En esta sección se deducirá el método a estudiar en este artículo. Para ello se comenzará aproximando el problema original (1) por el problema (5),

$$\begin{aligned} &\min_{\mathbf{v}\in\mathbb{R}^n} \quad J(\mathbf{v}) \\ &\text{Sujeto a:} \\ &\qquad \varphi_i(\mathbf{v}) < \varepsilon, \quad i = 1,\dots,m\,, \end{aligned} \tag{5}$$

para un valor suficientemente pequeño de $\varepsilon > 0$.

La ventaja de esta notación radica en que el conjunto admisible S es abierto. Por tanto, dado un punto $\mathbf{v} \in S$ existe un conjunto abierto $W \subset S$ tal que $\mathbf{v} \in W \subset S$. Entonces, tomando $\delta = \min_{\mathbf{y}\in\partial W} \|\mathbf{y}-\mathbf{v}\|$ se tiene que cualquier dirección $\mathbf{d} \in \mathbb{R}^n$, con $\|\mathbf{d}\| = 1$, es admisible, pues $\mathbf{v} + t\mathbf{d} \in S$ para todo $t \in [0, \delta]$.

Hecho este análisis, el objetivo es conseguir, a partir de un punto factible inicial $\mathbf{u}^{(0)} \in S$ cualquiera, una sucesión $\left\{\mathbf{u}^{(k)}\right\}_{k\in\mathbb{N}} \to \mathbf{u}$, donde $\mathbf{u}$ es un mínimo local del problema (5). En este sentido, se optará por construir la sucesión de tal forma que se reduzca el valor del funcional en cada iteración del método, es decir, la sucesión $\left\{\mathbf{u}^{(k)}\right\}_{k\in\mathbb{N}}$ se construirá de tal manera que:

$$J\left(\mathbf{u}^{(k+1)}\right) < J\left(\mathbf{u}^{(k)}\right) < \ldots < J\left(\mathbf{u}^{(1)}\right) < J\left(\mathbf{u}^{(0)}\right)\,.$$

Con este objetivo, se propone descender por el conjunto admisible a través de direcciones de descenso. Generalmente no todas las direcciones de descenso son admisibles, pero debido a que $DF(\mathbf{v}, S) = \mathbb{R}^n$ para todo $\mathbf{v} \in S$, entonces cualquier dirección de descenso es factible.

Intuitivamente se podría considerar, en un primer momento, como dirección de descenso en la iteración k-ésima el valor de $-\nabla J\left(\mathbf{u}^{(k)}\right)$, pues esta es la dirección de máximo descenso en un punto. Sin embargo, este razonamiento intuitivo (que parecería funcionar) no resulta de utilidad debido a las posibles formas que puede tomar el conjunto factible. A continuación se presenta el siguiente Contraejemplo 1 propio, propuesto en este artículo, para visualizar un caso.

Contraejemplo 1

Se considera el problema de minimización

$$\begin{aligned} &\min_{\mathbf{v}\in\mathbb{R}^2} \ J(\mathbf{v}) = v_1^2 + v_2^2 \\ &\text{Sujeto a:} \\ &\qquad \varphi_1(\mathbf{v}) = v_1^2 - v_2 + 1\,. \end{aligned} \tag{6}$$

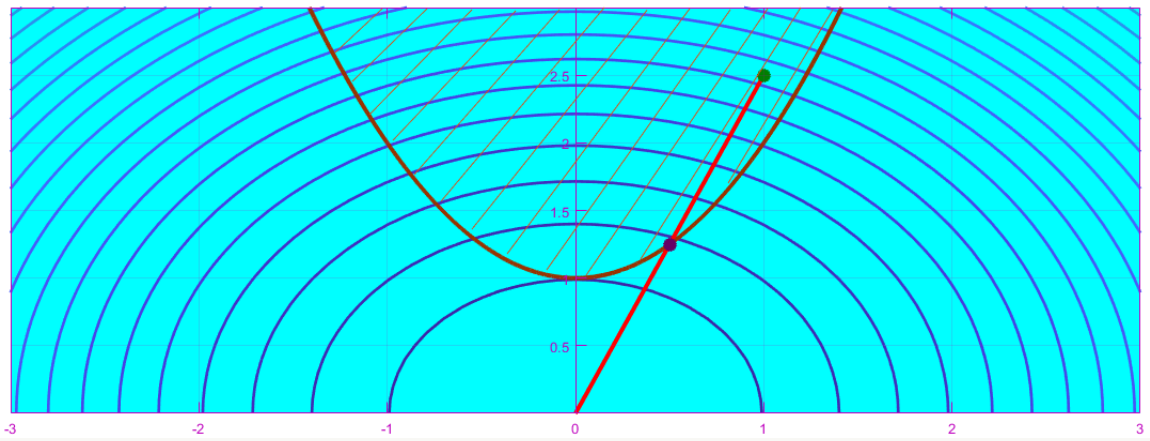


Figura 3: Representación del contraejemplo 1

En la Figura 3 se representa el problema 6. Tomando como punto inicial $\mathbf{u}^{(0)} = (1, 5/2)^T$ (el punto verde) y como dirección de descenso $\mathbf{d}^{(0)} = -\nabla J\left(\mathbf{u}^{(0)}\right)$, lo que se obtendría es un punto contenido en la recta roja de la imagen (dentro del conjunto admisible). Procediendo sucesivamente, sin salir del conjunto factible, se obtendría como punto de acumulación el $(1/2, 5/4)^T$ (el punto morado de la representación) que dista una distancia de 0.55902 unidades del mínimo del problema (que es el $(0, 1)^T$). Por consiguiente, no se está convergiendo a la solución que se busca.

Entonces, por el motivo visto en el Contraejemplo 1, es necesario ampliar el concepto de dirección de descenso a un marco mayor. Con este objetivo, se plantea la Definición 5.

Definición 5

Se denominará *dirección de descenso en el conjunto* S *respecto de un punto* $\boldsymbol{v}$ a una dirección de descenso $\mathbf{d} \in \mathbb{R}^n$ con $\|\mathbf{d}\| = 1$, tal que, siendo $\alpha \in \mathbb{R}$ el máximo valor verificando que $\mathbf{v} + \alpha\mathbf{d} \in S$ y $J(\mathbf{v} + \alpha\mathbf{d}) < J(\mathbf{v})$, esta cumple que:

$$J(\mathbf{v} + \alpha\mathbf{d}) < J(\mathbf{v} + \alpha'\mathbf{d}'), \qquad \forall \mathbf{d}' \text{ dirección de descenso.}$$

Observación 2

La Definición 5 toma como dirección de descenso aquella a través de la que más se desciende el valor del funcional. Como el conjunto admisible no es el total, puede que dicha dirección no sea la de máximo descenso (en el sentido habitual). Dicho de otro modo, puede que otra dirección de descenso recorra una mayor distancia permaneciendo en el conjunto admisible, y esto haga que el valor del funcional descienda más que usando simplemente el valor negativo del gradiente del funcional en el punto.

Teniendo en cuenta la Definición 5, sin pérdida de generalidad, se van a considerar a partir de ahora las direcciones de descenso normalizadas. Como el objetivo es crear una sucesión $\left\{\mathbf{u}^{(k)}\right\}_{k\in\mathbb{N}} \to \mathbf{u}$ donde $\mathbf{u}$ es solución al problema (5), en cada iteración se va a procurar aproximar la dirección $\mathbf{d}^{(k)} = \dfrac{\mathbf{u}^{(k)} - \mathbf{u}}{\|\mathbf{u}^{(k)} - \mathbf{u}\|}$ y el valor $\alpha_k = \|\mathbf{u}^{(k)} - \mathbf{u}\|$, para tomar $\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \alpha_k \mathbf{d}^{(k)}$.

El mayor inconveniente de este procedimiento erradica en la aproximación de las direcciones $\mathbf{d}^{(k)}$. Para ello se usará el concepto de discretización en el sentido de ángulos. Pues bien, $\mathbf{d}^{(k)} = \dfrac{\mathbf{u}^{(k)} - \mathbf{u}}{\|\mathbf{u}^{(k)} - \mathbf{u}\|}$ es una dirección de descenso, luego ${\mathbf{d}^{(k)}}^{\mathsf{T}} \nabla J(\mathbf{u}^{(k)}) < 0$. Ahora bien, como

$$ {\mathbf{d}^{(k)}}^{\mathsf{T}} \nabla J(\mathbf{u}^{(k)}) = \|\mathbf{d}^{(k)}\| \left\|\nabla J\left(\mathbf{u}^{(k)}\right)\right\| \cos\left(\measuredangle\left(\mathbf{d}^{(k)}, -\nabla J\left(\mathbf{u}^{(k)}\right)\right)\right) < 0, $$

se deduce que $\measuredangle\left(\mathbf{d}^{(k)}, -\nabla J\left(\mathbf{u}^{(k)}\right)\right) \in (-\pi/2, \pi/2)$.

Considerando entonces $-\nabla J\left(\mathbf{u}^{(k)}\right)$, se pretende discretizar los valores de los ángulos que forman con $-\nabla J\left(\mathbf{u}^{(k)}\right)$ un ángulo comprendido en el intervalo $(-\pi/2, \pi/2)$. Es decir, se calculan direcciones separadas por un mismo ángulo entre ellas cumpliendo que su producto escalar con la dirección de máximo descenso sea negativo.

Por otra parte, como las direcciones calculadas deben tener norma 1, se pueden interpretar como puntos de la esfera $\mathbb{S}^n$. Así, si se tiene en cuenta lo que se pretende hacer con la discretización, lo que se busca es calcular una discretización de puntos del hemisferio de la esfera $\mathbb{S}^n$ que tiene como polo el punto $\Gamma^{(k)} = -\nabla J\left(\mathbf{u}^{(k)}\right) / \left\|-\nabla J\left(\mathbf{u}^{(k)}\right)\right\|$. Con este objetivo, se recurre a las coordenadas polares, las cuales resultan cómodas desde el punto de vista práctico.

Ejemplo 1

Si se tiene un problema de dimensión 2, se consideraría la esfera $\mathbb{S}^2$ (la circunferencia usual). Entonces, se parte de $\theta_k \in [0, 2\pi]$ el valor que define el punto $\Gamma^{(k)}$ en coordenadas polares, es decir, $(\cos(\theta_k), \sin(\theta_k))^{\mathsf{T}} = \Gamma^{(k)}$. Luego, se divide el intervalo $(0, \pi/2)$ en una serie de puntos igualmente espaciados $0 < t_1 < \ldots < t_m < \pi/2$, por lo que tomando

$$ \begin{cases} \xi_i = \theta_k + t_i, & i = 1, \ldots, m \\ \xi_{m+i} = \theta_k - t_i, & i = 1, \ldots, m \end{cases} $$

se obtienen $2m + 1$ puntos que discretizan el hemisferio de la esfera $\mathbb{S}^2$, cuyo polo es el punto $\Gamma^{(k)}$.

En la Figura 4 se muestra un ejemplo de discretización de la esfera $\mathbb{S}^2$ entorno al punto $(\sqrt{2}/2, \sqrt{2}/2)^{\mathsf{T}}$ (punto rojo). En torno a él, los puntos morados son los que generan la discretización buscada (con 4 puntos en este caso, 8 en total) y los puntos en amarillo serán los que tendrían producto escalar nulo al operarlos con el punto de estudio en este caso.

Para un espacio de dimensión $n > 2$, es necesario recurrir a la expresión de las coordenadas esféricas. En el caso general, para una esfera de dimensión n, se tiene que las coordenadas de un punto arbitrario

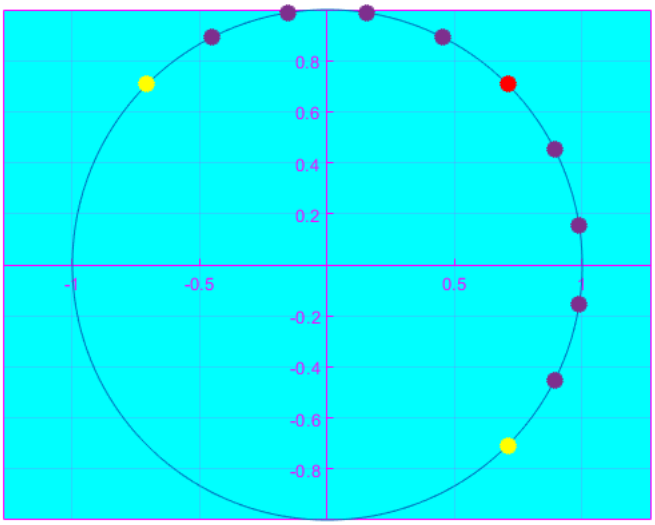


Figura 4: Discretización de la esfera $\mathbb{S}^2$

v vienen dadas por,

$$\begin{cases} v_1 = r\cos(\tau_1) \\ v_2 = r\sin(\tau_1)\cos(\tau_2) \\ v_3 = r\sin(\tau_1)\sin(\tau_2)\cos(\tau_3) \\ \vdots \\ v_{n-1} = r\sin(\tau_1)\sin(\tau_2)\ldots\sin(\tau_{n-2})\cos(\tau_{n-1}) \\ v_n = r\sin(\tau_1)\sin(\tau_2)\ldots\sin(\tau_{n-2})\sin(\tau_{n-1}) \end{cases}$$

donde r es el radio de la esfera (en este caso 1, por lo que se puede omitir), y los valores de $\tau_1, \ldots, \tau_{n-1}$ vienen dados por:

$$\begin{cases} \tau_1 = \operatorname{arc\,cot}\left(\dfrac{v_1}{\sqrt{v_n^2 + v_{n-1}^2 + \ldots + v_2^2}}\right) = \arccos\left(\dfrac{v_1}{\sqrt{v_n^2 + v_{n-1}^2 + \ldots + v_1^2}}\right) \\ \tau_2 = \operatorname{arc\,cot}\left(\dfrac{v_2}{\sqrt{v_n^2 + v_{n-1}^2 + \ldots + v_3^2}}\right) = \arccos\left(\dfrac{v_2}{\sqrt{v_n^2 + v_{n-1}^2 + \ldots + v_2^2}}\right) \\ \vdots \\ \tau_{n-2} = \operatorname{arc\,cot}\left(\dfrac{v_{n-2}}{\sqrt{v_n^2 + v_{n-1}^2}}\right) = \arccos\left(\dfrac{v_{n-2}}{\sqrt{v_n^2 + v_{n-1}^2 + v_{n-2}^2}}\right) \\ \tau_{n-1} = \begin{cases} \arccos\left(\dfrac{v_{n-1}}{\sqrt{v_n^2 + v_{n-1}^2}}\right) & x_n \geq 0 \\ 2\pi - \arccos\left(\dfrac{v_{n-1}}{\sqrt{v_n^2 + v_{n-1}^2}}\right) & v_n < 0 \end{cases} \end{cases}$$

donde $\tau_1, \ldots, \tau_{n-2} \in [0, \pi]$ y $\tau_{n-1} \in [0, 2\pi)$.

Con estas fórmulas es posible crear una discretización de un hemisferio de la esfera $\mathbb{S}^n$, con $n > 2$. Para ello se procede como en el caso de la esfera $\mathbb{S}^3$, salvo que en este caso existen más variables. Para los ángulos $\tau_1, \ldots, \tau_{n-2}$, se divide el intervalo $(0, \pi/2)$ en m partes iguales $(0 < t_1 < \ldots, t_m < \pi/2)$

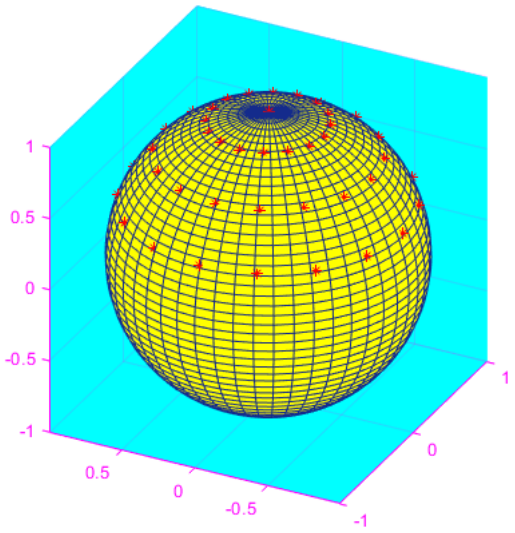


Figura 5: Discretización de la esfera $\mathbb{S}^3$

y se toman los ángulos $\tau_j + t_i$ con $j = 1, \ldots, n-2$ e $i = 1, \ldots, m$. Por otra parte, para τ_{n-1} se divide el intervalo $[0, 2\pi)$ de tal modo que $0 < q_1 < \ldots < q_{4m} < 2\pi$ y se toman los ángulos $\tau_{n-1} + q_l$ con $l = 1, \ldots, 4m$. Entonces, haciendo todas las posibles combinaciones entre los ángulos obtenidos, se obtiene toda la discretización buscada de un hemisferio de la esfera $\mathbb{S}^n$. En la Figura 5 se muestra un ejemplo visual de una discretización de $\mathbb{S}^3$ con $m = 4$ para el hemisferio que tiene por polo el punto $\mathbf{e}^{(3)} = (0, 0, 1)^T$.

Sin embargo, el problema que resulta de esta construcción de la esfera, es que esta se entiende como una superficie de revolución. En este caso el eje bajo el que se gira es entorno al vector $\mathbf{e}^{(1)} = (1, 0, \ldots, 0)^T$, pero el interés es que sea sobre el vector que determina $\Gamma^{(k)}$. Para ello, bajo la discretización obtenida del hemisferio cuyo polo es el vector $\mathbf{e}^{(1)}$, lo que se hará es una isometría $f : \mathbb{R}^n \to \mathbb{R}^n$ de tal manera que $f(\mathbf{e}^{(1)}) = \Gamma^{(k)}$. Para construir dicha isometría se comenzará tomando el vector $\Gamma^{(k)}$ y se completará a una base del espacio vectorial $\mathbb{R}^n$. Luego, para conseguir que esta nueva base sea ortonormal, se puede usar algún procedimiento conocido como el método de ortogonalización de Gram-Schmidt que se muestra en el Algoritmo 1, y cuya demostración y deducción se puede consultar en [4, pag. 273] o [10, pág. 365-366].

Algoritmo 1 Método de ortogonalización de Gram-Schmidt

Si $B = \{\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(n)}\}$ es una base del espacio euclidiano V, entonces se puede construir una base ortogonal $B' = \{\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(n)}\}$ tomando

$$\begin{cases} \mathbf{v}^{(1)} := \mathbf{u}^{(1)} \\ \mathbf{v}^{(2)} := \mathbf{u}^{(2)} + \lambda_{21}\mathbf{v}^{(1)} \\ \vdots \\ \mathbf{v}^{(i)} := \mathbf{u}^{(i)} + \sum_{j=1}^{i-1} \lambda_{ij}\mathbf{v}^{(j)}\,, \end{cases}$$

donde $\lambda_{ij} = -\dfrac{{\mathbf{u}^{(i)}}^T \mathbf{v}^{(j)}}{{\mathbf{v}^{(j)}}^T \mathbf{v}^{(j)}}$.

Entonces se completa $\Gamma^{(k)}$ a una base y, con el método de Gram-Schimtd, a una base ortonormal. De este modo, sea $\mathbf{A}$ la matriz que tiene por columnas los vectores de esta base ortonormal, si el

determinante $|\mathbf{A}| = -1$, se intercambian dos columnas (sin incluir la primera). Así se obtiene que $|\mathbf{A}| = 1$, por lo que esta matriz determina una aplicación lineal que mantiene la orientación del espacio y, en consecuencia, define una isometría $f : \mathbb{R}^n \to \mathbb{R}^n$ tal que $f(\mathbf{e}^{(1)}) = \Gamma^{(k)}$. De esta manera, si se emplea la aplicación definida por $\mathbf{A}$ a cada punto de la discretización obtenida por el procedimiento descrito anteriormente, se obtienen los puntos de la discretización del hemisferio que tiene por polo el punto que se buscaba.

Una vez vistos todos los aspectos relativos a la discretización, sólo queda comparar todas las direcciones calculadas por el método de discretización para escoger la mejor. Es decir, se comprobará (numéricamente) cuál, de entre todas las direcciones obtenidas, cumple la Definición 5. Una vez hallada dicha dirección, $\mathbf{d}^{(k)}$, se toma $\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \alpha_k \mathbf{d}^{(k)}$, de tal manera que $J\left(\mathbf{u}^{k+1}\right) < J\left(\mathbf{u}^{k}\right)$, por lo que se crea una sucesión $\left\{\mathbf{u}^{(k)}\right\}_{k\in\mathbb{N}}$ de tal forma que

$$J\left(\mathbf{u}^{(k+1)}\right) < J\left(\mathbf{u}^{(k)}\right) < \ldots < J\left(\mathbf{u}^{(2)}\right) < J\left(\mathbf{u}^{(1)}\right) .$$

Cuando el valor de α_k es muy pequeño, indica que, de entre todas la direcciones, en la que mayor descenso se produce y se avanza, el avance es mínimo y, por tanto, se alcanza una convergencia. Por consiguiente, un criterio de parada que se podría tomar, sería que α_k fuese menor que una cierta tolerancia.

Con esto, se concluiría la deducción del método que se plantea. Haciendo una recopilación de lo visto, se puede presentar el siguiente algoritmo:

Algoritmo 2

Se procede con el siguiente procedimiento iterativo:
PASO 0: Se parte de un $\mathbf{u}^{(0)} \in S$, un valor de tolerancia $tol > 0$ pequeño y $k = 0$.
PASO 1: Se calcula $\Gamma^{(k)}$ y, a partir de él, se construye una discretización.
PASO 2: A partir de la discretización obtenida, se comparan todas las direcciones para encontrar aquella que cumple la Definición 5, de lo que se obtiene como solución la dirección $\mathbf{d}^{(k)}$ y el paso α_k.
PASO 3: Se toma $\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \alpha_k \mathbf{d}^{(k)}$. Si $\alpha_k < tol$ se finaliza y, en caso contrario, se vuelve al PASO 1.

4. Aspectos computacionales

Para poder aplicar en la práctica el Algoritmo 2, este se debe poder programar en algún lenguaje de programación para su utilización. Por ello, en esta sección se presentará una manera de programar el método planteado en el lenguaje de MatLab.

Código 1: Programa principal

```
% Declaracion de datos
u0, tol, alpha0, gradJ, J % Se introducen los datos, el iterante inicial, la
    tolerancia, un valor inicial de alpha, el gradiente del funcional y el
    funcional objetivo
it=1; % Se comienza con el contador de iteraciones.
while (alpha0 > tol)
    d1=-gradJ(u0)./norm(-gradJ(u0),2); % Direccion de maximo descenso
    k=20; % Se aporta un valor para la discretizacion
    [di]=discretizacion(d1,k); % Se llama a la funcion discretizacion
    [fi,co]=size(di); alpha0=10;
    while ((conjunto(u0,alpha0,d1) == 0)  ||  (J(u0+alpha0*d1) > J(u0)))
```

```
        alpha0=alpha0/2; % Se reduce el valor incial de alpha hasta obtener
            factibilidad y descenso
    end
% Se compara el resto de direcciones de la discretizacion:
    for i=2:fi
        alpha=10;
        while ((conjunto(u0,alpha,di(i,:)') == 0)  ||  ( J(u0+alpha*di(i,:)) > f
            (u0)))
            alpha=alpha/2; % Se reduce el valor incial de alpha hasta obtener
                factibilidad y descenso
        end
        if (J(u0+alpha*di(i,:)') < J(u0+alpha0*d1)) % Se encuentra una mejor
            direccion de descenso
            alpha0=alpha; % Se actualiza el valor de alpha
            d1=di(i,:)'; % Se actualiza la direccion de descenso
        end

    end
    u1=u0+alpha0*d1; u0=u1; % Se actualizan los iterantes
    it=it+1; % Se actualiza el numero de iteraciones
end
Sol=u0;
```

En Código 1 se muestra un ejemplo de programa principal. En este programa se muestra una forma de programar el Algoritmo 2. Sin embargo, en él se utilizan diferentes funciones auxiliares. En concreto se utiliza la función "discretizacion" para crear la discretización de las direcciones y la función "conjunto" para determinar si el punto obtenido al desplazarse en una dirección un determinado valor, a partir de un punto dado, pertenece al conjunto admisible.

Código 2: Función que calcula la discretización

```
function [di]=discretizacion(d,k)
    n=length(d);
    if (n == 2)
        di=zeros(2*k-1,2); di(1,:)=d;
        if (d(2) >= 0)
            varphi=acos(d(1)/norm(d,2));
        else
            varphi=2*pi-acos(d(1)/norm(d,2));
        end
        i=1:k-1; angulospos=varphi+(pi/(2*k))*(i); angulosneg=varphi-(pi/(2*k))
            *(i);
        angulos=[angulospos,angulosneg];
        for i=1:2*(k-1)
            di(i+1,:)=[cos(angulos(i)),sin(angulos(i))];
        end
    else
        A=eye(n);
        for i=1:n
            if (d(i) ~= 0)
                A(:,i)=A(:,1);
                A(:,1)=d;
                break
            end
        end
        [q,r]=gramscb(A); % Se llama a la función que aplica el metodo de Gram-
            Schmidt
        if (sign(det(q)) == -1)
            a=q(:,2);
            q(:,2)=q(:,3);
            q(:,3)=a;
```

```
        end
        di=zeros((4*k)*(k-1)^(n-2)+1,length(d));
        varphi=ones(1,n-1);
        e1=zeros(1,n); e1(1)=1; di(1,:)=e1;
        for i=1:n-2
            varphi(i)=acos(e1(i)/sqrt(sum(e1(i:n).^2)));
            if (sum(e1(i:n).^2) == 0)
                varphi(i)=pi/2;
            end
        end
        if (e1(n) >= 0)
            varphi(n-1)=e1(n-1)/sqrt(e1(n)^2+e1(n-1)^2);
            if ((e1(n)^2+e1(n-1)^2) == 0)
                varphi(n-1)=pi/2*sign(e1(n-1));
            end
        else
            varphi(n-1)=2*pi-acos(e1(n-1)/sqrt(e1(n)^2+e1(n-1)^2));
        end
        i=1:k-1; VARPHI=cell(n-2,1);
        for l=1:n-2
            VARPHI(l,:)={varphi(l)+i*pi/(2*k)};
        end
        j=1:4*k; VARPHIfin=varphi(n-1)+j*2*pi/(4*k);
        combinaciones=combvec(VARPHI{1:n-2,:},VARPHIfin)';
        for i=1:(4*k)*(k-1)^(n-2)
            angulos=combinaciones(i,:);
            di(i+1,1)=cos(angulos(1));
            for j=2:n-1
                di(i+1,j)=prod(sin(angulos(1:j-1)))*cos(angulos(j));
            end
            di(i+1,n)=prod(sin(angulos(1:n-1)));
        end
        di=(q*di')';
    end
end
```

En Código 2 se expone la función empleada para calcular la discretización del hemisferio de la esfera $\mathbf{S}^n$ cuyo polo es un vector determinado. Los datos de entrada son d, el vector que determina el polo en torno al que se desea discretizar la esfera, y el valor de k que se corresponde con el número de puntos de la discretización. De este modo, para un k mayor se obtienen más puntos de la discretización, mientras que si se reduce su valor, también lo hace el número de direcciones a comprobar. Por otra parte, se utiliza una función con el nombre "gramscb" que lo que hace es aplicar el método de ortogonalización de Gram-Schmidt explicado en el Algoritmo 1 (dado que el método de Gram-Schmidt es de fácil aplicación, se deja su programación al lector).

Código 3: Función que verifica la factibilidad de un punto obtenido a partir del anterior mediante una dirección y un valor α concreto

```
function z=conjunto(u,alpha,d)
    y=g1(u+alpha*d); % La funcion g1 devuelve un vector cuyas componentes son la
        evaluacion de las restricciones
    for i=1:length(y)
        if (y(i) < eps) % Se utiliza el epsilon de la maquina como comparativa
            z=1;
        else
            z=0;
            return
        end
    end
end
```

En Código 3 se expone el programa utilizado para comprobar la factibilidad de un nuevo elemento obtenido a partir de un punto arbitrario, una dirección y un valor "alpha" que se corresponde con el desplazamiento desde el punto arbitrario en la dirección dada. Este programa verifica si el siguiente iterante del Algoritmo 2 es admisible para el problema (2) tomando como valor de ε el propio épsilon de la máquina.

De esta forma se concluiría con los aspectos computacionales referentes al Algoritmo 2. Es importante resaltar que la programación aquí mostrada es un ejemplo de cómo se podría llevar a la práctica, pero en realidad podría haber otras versiones. Un ejercicio interesante resultaría de reflexionar la forma en la que se podría abaratar el coste computacional de esta programación. Estos aspectos se dejan a reflexión de los lectores y a juicio del público para mejorar el método y la programación planteados.

5. Pruebas y ejecuciones

Con el objetivo de testear el buen funcionamiento del Algoritmo 2, se ha probado su ejecución en una serie de problemas de sencillo cálculo empleando la programación expuesta en la Sección 4. Para ello se seleccionaron una serie de problemas, extraídos de [11], con los que corregir el método creado y comprobar su funcionamiento, además de incluir el Contra-Ejemplo 1. Así mismo, esos mismos problemas se resolverán con otros algoritmos ya conocidos para comparar los resultados obtenidos. En este sentido, en [1] se pueden apreciar una recopilación de métodos ya conocidos que el lector puede consultar como ampliación de esta sección. Sin embargo, en este artículo se mostrarán los resultados obtenidos con la función *fmincon* de MatLab y el método del Lagrangiano aumentado presentado en el artículo [14].

- **Solución al problema del Contraejemplo 1:**
  Se considera el problema (6), que se tomó en el Contraejemplo 1, para ver el funcionamiento del algoritmo en este caso:

$$\begin{aligned} &\min_{\mathbf{v}\in\mathbb{R}^2} \ J(\mathbf{v}) = v_1^2 + v_2^2 \\ &\text{Sujeto a:} \\ &\qquad \varphi_1(\mathbf{v}) = v_1^2 - v_2 + 1 \, . \end{aligned} \tag{7}$$

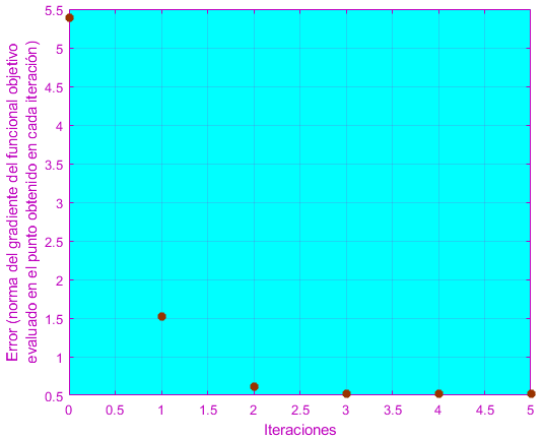
  El iterante inicial tomado es $\mathbf{u}^{(0)} = \begin{pmatrix} 1 & 5/2 \end{pmatrix}^{\mathsf{T}}$ de tal forma que $J(\mathbf{u}^{(0)}) = 29/4$. La solución exacta al problema es $\mathbf{u} = \begin{pmatrix} 0 & 1 \end{pmatrix}^{\mathsf{T}}$, lo que implica que el valor del funcional objetivo en el mínimo es $J(\mathbf{u}) = 1$.

Tabla 1: Resultados obtenidos al problema (7)

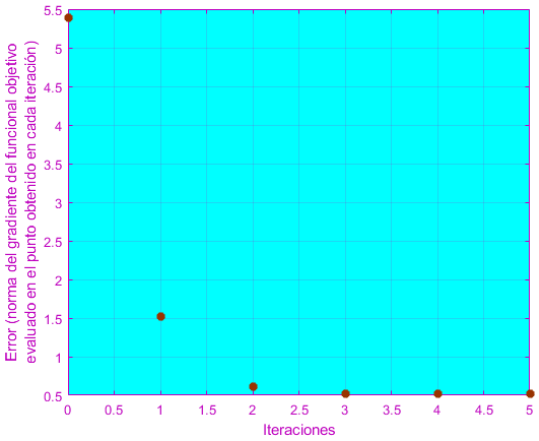
Resultados	Algoritmo 2	Función *fmincon* en MatLab	Lagrangiano aumentado
Solución	$\begin{pmatrix} -0.0006 \\ 1.0000 \end{pmatrix}$	$\begin{pmatrix} 0.0000 \\ 1.0000 \end{pmatrix}$	$\begin{pmatrix} 0.0000 \\ 1.0000 \end{pmatrix}$
Valor de la función objetivo	1.0000	1.0000	1.0000
Iteraciones	5	8	7

- **Problema de 2 dimensiones:**
  Se considera el problema (8), que se corresponde con el ejemplo número 22 en [11],

$$\begin{aligned} &\min_{\mathbf{v}\in\mathbb{R}^2} J(\mathbf{v}) = (v_1 - 2)^2 + (v_2 - 1)^2 \\ &\text{Sujeto a:} \\ &\qquad v_1 + v_2 - 2 \leq 0 \\ &\qquad v_1^2 - v_2 \leq 0 \, . \end{aligned} \tag{8}$$



(**a**) Representación del error en cada iteración



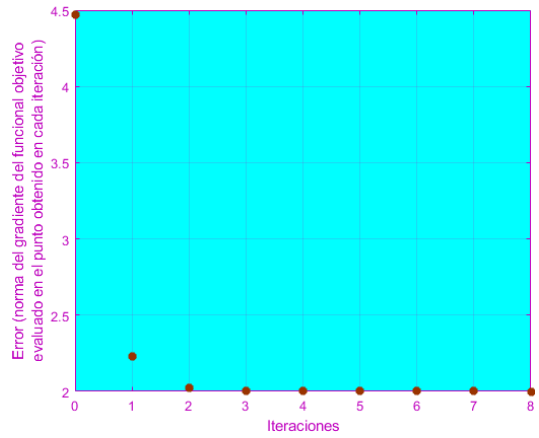
(**b**) Representación del valor del funcional objetivo en cada iteración

Figura 6: Resultados por iteraciones del problema (7).

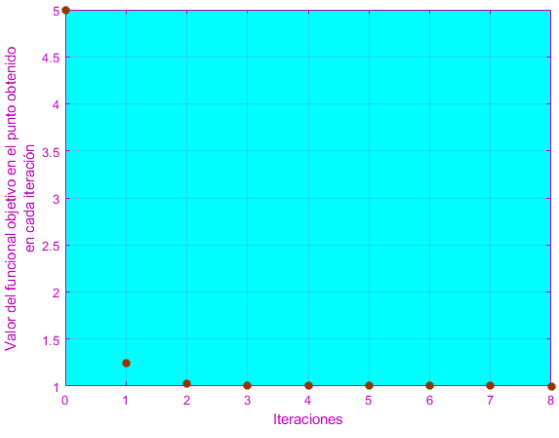
El iterante inicial tomado es $\mathbf{u}^{(0)} = \begin{pmatrix} 0 & 0 \end{pmatrix}^{\mathsf{T}}$ de tal forma que $\mathrm{J}(\mathbf{u}^{(0)}) = 0$. La solución exacta al problema es $\mathbf{u} = \begin{pmatrix} 1 & 1 \end{pmatrix}^{\mathsf{T}}$ lo que implica que el valor del funcional en el mínimo es $\mathrm{J}(\mathbf{u}) = 1$.

Tabla 2: Resultados obtenidos al problema (8).

Resultados	Algoritmo 2	Función *fmincon* en MatLab	Lagrangiano aumentado
Solución	$\begin{pmatrix} 1.0000 \\ 1.0000 \end{pmatrix}$	$\begin{pmatrix} 1.0000 \\ 1.0000 \end{pmatrix}$	$\begin{pmatrix} 1.0000 \\ 1.0000 \end{pmatrix}$
Valor de la función objetivo	1.0000	1.0000	1.0000
Iteraciones	8	10	11



(**a**) Representación del error en cada iteración



(**b**) Representación del valor del funcional objetivo en cada iteración

Figura 7: Resultados por iteraciones del problema (8).

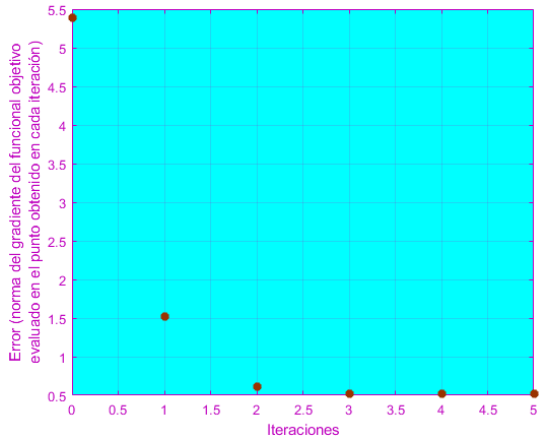
- **Problema de 3 dimensiones:**
  Se considera el problema (9), que se corresponde con el ejemplo número 35 en [11],

$$\begin{aligned} &\min_{\mathbf{v}\in\mathbb{R}^2} J(\mathbf{v}) = 9 - 8v_1 - 6v_2 - 4v_3 + 2v_1^2 + 2v_2^2 + v_3^2 + 2v_1v_2 + 2v_1v_3 \\ &\text{Sujeto a:} \\ &\qquad v_1 + v_2 + 2v_3 - 3 \leq 0 \\ &\qquad v_1 \leq 0 \\ &\qquad v_2 \leq 0 \\ &\qquad v_3 \leq 0\,. \end{aligned} \tag{9}$$

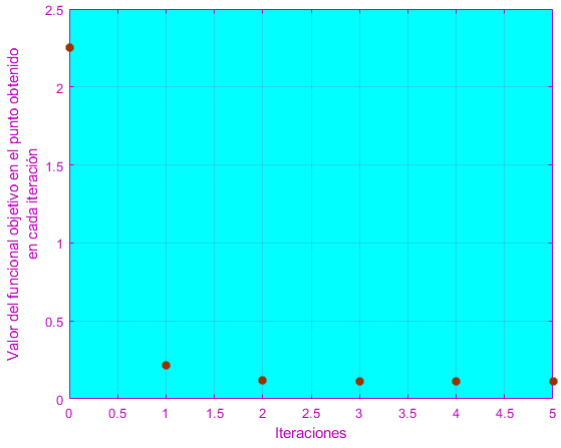
  El iterante inicial tomado es $\mathbf{u}^{(0)} = \begin{pmatrix} 1/2 & 1/2 & 1/2 \end{pmatrix}^T$, de tal forma que $J(\mathbf{u}^{(0)}) = 2.25$. La solución exacta al problema es $\mathbf{u} = \begin{pmatrix} 4/3 & 7/9 & 4/9 \end{pmatrix}^T$, lo que implica que el valor del funcional en el mínimo es $J(\mathbf{u}) = 1/9$.

Tabla 3: Resultados obtenidos al problema (9).

Resultados	Algoritmo 2	Función *fmincon* en MatLab	Lagrangiano aumentado
Solución	$\begin{pmatrix} 1.3452 \\ 0.7698 \\ 0.4425 \end{pmatrix}$	$\begin{pmatrix} 1.3333 \\ 0.7778 \\ 0.4444 \end{pmatrix}$	$\begin{pmatrix} 1.3333 \\ 0.7777 \\ 0.4444 \end{pmatrix}$
Valor de la función objetivo	0.1113	0.1111	0.1111
Iteraciones	5	11	6



(**a**) Representación del error en cada iteración



(**b**) Representación del valor del funcional objetivo en cada iteración

Figura 8: Resultados por iteraciones del problema (9).

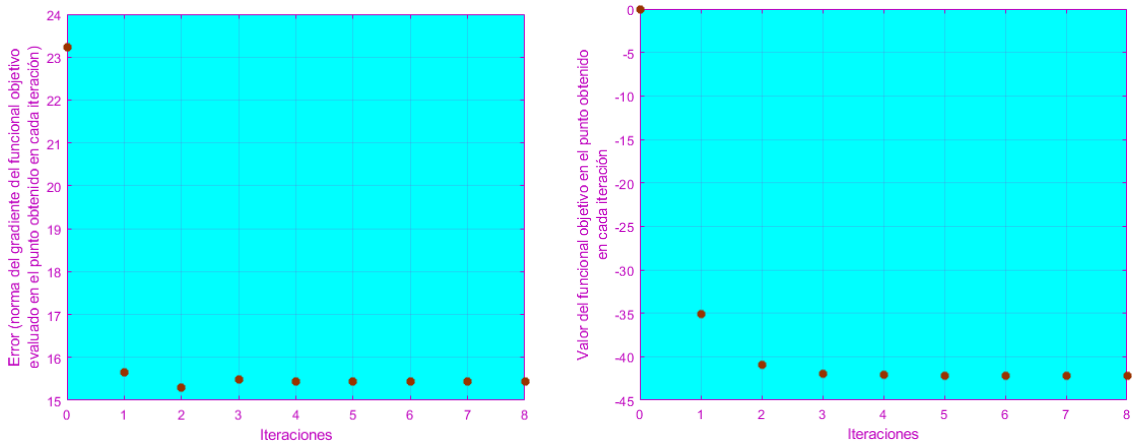
- **Problema de 4 dimensiones:**
  Se considera el problema (10), que se corresponde con el ejemplo número 43 en [11],

$$\begin{aligned} &\min_{\mathbf{v}\in\mathbb{R}^2} J(\mathbf{v}) = v_1^2 + v_2^2 + 2v_3^2 + v_4^2 - 5v_1 - 5v_2 - 21v_3 + 7v_4 \\ &\text{Sujeto a:} \\ &\qquad v_1^2 + v_2^2 + v_3^2 + v_4^2 + v_1 - v_2 + v_3 - v_4 - 8 \leq 0 \\ &\qquad v_1^2 + 2v_2^2 + v_3^2 + 2v_4^2 - v_1 - v_4 - 10 \leq 0 \\ &\qquad 2v_1^2 + v_2^2 + v_3^2 + 2v_1 - v_2 - v_4 - 5 \leq 0\,. \end{aligned} \tag{10}$$

  El iterante inicial tomado es $\mathbf{u}^{(0)} = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}^T$, de tal forma que $J(\mathbf{u}^{(0)}) = 0$. La solución exacta al problema es $\mathbf{u} = \begin{pmatrix} 0 & 1 & 2 & -1 \end{pmatrix}^T$, lo que implica que el valor del funcional en el mínimo es $J(\mathbf{u}) = -44$.

Tabla 4: Resultados obtenidos al problema (10).

Resultados	Algoritmo 2	Función *fmincon* en MatLab	Lagrangiano aumentado
Solución	$\begin{pmatrix} 0.1480 \\ 0.3497 \\ 1.9444 \\ -1.1070 \end{pmatrix}$	$\begin{pmatrix} 0.0000 \\ 1.0000 \\ 2.0000 \\ -1.0000 \end{pmatrix}$	$\begin{pmatrix} -0.0744 \\ 1.0178 \\ 2.0439 \\ -0.9417 \end{pmatrix}$
Valor de la función objetivo	-42.1388	-44	-43.9488
Iteraciones	8	12	648



(**a**) Representación del error en cada iteración

(**b**) Representación del valor del funcional objetivo en cada iteración

Figura 9: Resultados por iteraciones del problema (10).

A la vista de los resultados obtenidos se puede comprobar como el testeo del Algoritmo 2 ha respondido de forma positiva. Se podrían hacer más testeos, e incluso más complejos, pero se deja como ejercicio al lector. Lo que se ha querido mostrar con estas ejecuciones es el correcto funcionamiento del método planteado que se aproxima al valor del mínimo del problema original.

Por otra parte, un aspecto importante son los tiempos de ejecución, que no se incluyen en los resultados debido a que estos varían en función de cada computadora. Sin embargo, hablando de los tiempos obtenidos mediante el Algoritmo 2, es conveniente decir que en el problema (8) se obtiene el resultado en un tiempo razonable, algo que incluso sucede con el problema (9). Sin embargo, en la resolución del problema (10) el tiempo de ejecución se dispara a más de un minuto, algo extraño en un problema de sólo cuatro dimensiones. Este hecho se debe al número de direcciones que el algoritmo debe de evaluar en cada iteración. Con dimensiones pequeñas el número de direcciones de descenso, creadas con la discretización, no es demasiado elevado, pero este número crece de forma exponencial según se aumenta la dimensión del problema.

Esta carencia es importante tenerla en cuenta. De hecho, para que el Algoritmo 2 aporte resultados concluyentes, es importante construir una discretización amplia para poder comparar el mayor número de direcciones de descenso posibles. Por este motivo, el algoritmo se ve limitado en el tiempo de ejecución y, al mismo tiempo, en la memoria requerida en cada iteración para almacenar las direcciones que se deben de comprobar.

Por consiguiente, una futura línea de investigación para el método planteado en este artículo es la forma en la que se podría reducir la memoria y el tiempo de ejecución empleado por el algoritmo. En este sentido se podría hacer un estudio acerca de la forma en la que se podría reducir el número de direcciones de descenso en la discretización sin que esto implique menor precisión del método.

6. Conclusiones y futuras líneas de investigación

En base al trabajo desarrollado en el presente artículo, se obtienen las siguientes conclusiones:

1. El método expuesto se enmarca dentro del conjunto de métodos de direcciones factibles, permitiendo resolver una serie de problemas de optimización con restricciones cuyos conjuntos admisibles sean cerrados y conexos.
2. El interés prioritario del algoritmo planteado se basa en el aporte teórico de su deducción. A lo largo del artículo se muestran diferentes interpretaciones geométricas y se relacionan distintas disciplinas matemáticas para obtener el método final.
3. Una de las ventajas del algoritmo es que permite crear una sucesión de puntos factibles que descienden el valor del funcional objetivo en cada iteración. Así pues, siempre se está descendiendo el valor del funcional en cada iteración.
4. La principal desventaja que presenta el método es su coste computacional, lo que implica altos tiempos de ejecución en la práctica según aumenta la dimensión de los problemas a resolver. Por otra parte, el coste de memoria de la programación expuesta es también una desventaja del método según se aumenta el de la dimensión de los problemas a resolver.

Una vez expuestas las principales conclusiones, se muestran las principales líneas de investigación para mejorar el método expuesto en el artículo:

1. Resulta primordial poder reducir la memoria requerida por la programación del método. Sería muy interesante descartar, en cada iteración, una serie de direcciones en lugar de tener que considerarlas.
2. La reducción de direcciones de descenso debe de hacerse de tal forma que la discretización no se vea reducida, para no menguar el nivel de precisión del método.
3. En lugar de reducir el valor de α a la mitad cómo se hace en el Código 1, sería conveniente desarrollar un método más efectivo de calcular el valor de α. Para ello se podría investigar acerca de cómo aplicar la regla del Armijo, de Goldestein o de Wolfe-Powell a este algoritmo, las cuales se pueden estudiar en [17, Sec. 2.5], que podrían aportar un mejor método de reducir el valor inicial de α.

7. Bibliografía

[1] Ancau, M. *Practical Optimization with MatLab*. Cambridge Scholars Publishing. 2019.

[2] Boglárka G.-Tóth and Eligius M.T. Hendrix. *Introduction to Nonlinear and Global Optimization*. Springer. 2010.

[3] Crilly, A.J, and Pérez, E.H. *50 cosas que hay que saber sobre matemáticas*, Ariel, 2014.

[4] De Burgos J. *álgebra lineal y geometría cartesiana*. Ed. MacGraw-Hill, Madrid, 1999.

[5] Eiselt, H.A. and Sandblom, C..L. *Nonlinear Optimization. Methods and Applications*. Springer. 2019.

[6] Eremeev, A. Khachay, M., Kochetov, Y. and Pardalos, P. *Optimization Problems and Their Applications*. Springer. 2018.

[7] Escudero Andino, F. F.*Diseño de un modelo matemático para optimizar las rutas de recorrido del proceso de recolección de desechos sólidos para el Cantón Valencia*. Master's thesis, Universidad Técnica de Ambato. Facultad de Ingeniería en Sistemas, Electrónica e Industrial. Maestría en Matemática Aplicada. 2021

[8] Gallier, J. and Quaintance, J. *Fundamentals of Optimization TheoryWith Applications to Machine Learning*. University of Pennsylvania Philadelphia. 2019.

[9] González Castaño, F. A. *Análisis de ciclo de vida y optimización matemática como herramientas para la producción sustentable de energía y de productos químicos.* 2019

[10] Hernández, E. *álgebra y geometría*. Ed. Addison Wesley, Madrid, 1994.

[11] Hock W. and Schittkowski K. *Test Examples for Nonlinear Programming Codes*, Lecture Notes in Economical and Mathematical Systems 187, Springer-Verlag, Berlin, 1981.

[12] Nocedal, Jorge and Wright, Stephen J. *Numerical Optimization.* Springer. 1999.

[13] Peressini, A.L; Sullivan, F.E and Uhl, J.J. *The mathematics of nonlinear programming*. Springer-Verlag. 1988

[14] Vázquez, M. *Resolución de los problemas de optimización con restricciones mediante los métodos de penalización y del Lagrangiano aumentado.* Revista digital Matemática, Educación e Internet, 21(2). 2021. Recuperado de: https://tecdigital.tec.ac.cr/revistamatematica/

[15] Silva Melgarejo, M. A., and Zevallos Diaz, A. J. E.*Programación lineal para optimizar separación de grasas del proceso de fabricación de harina de pescado.* Corporación Hayduk SA Coishco. 2019.

[16] Yang, X-S. *Optimization Techniques and Applications with Examples*. John Wiley and Sons. 2018.

[17] Sun W. and Y-X Yuan. *Optimization Theory and Methods. Nonlinear Programming*, Springer, 2006.