Universidad de La Habana
Facultad de Matemática y Computación

Un generador de problemas prueba para evaluar la calidad de la solución de los algoritmos de problemas de optimización de dos niveles

Autor:
Francisco Vicente Suárez Bellón

Tutora:
Dr. C. Gemayqzel Bouza Allende

Trabajo de Diploma
presentado en opción al título de
Licenciado en Ciencia de la Computación

Fecha

github.com/FVSB/Tesis

Agradecimientos

Agradecimientos

El camino recorrido hasta este momento ha sido posible gracias al apoyo y dedicación de personas extraordinarias que han estado presentes en cada paso de mi formación académica. Deseo expresar mi más sincero agradecimiento a todos los profesores que han contribuido a mi desarrollo profesional durante mi trayectoria universitaria.

Mi más profunda gratitud a la profesora Dra. C. Gemayqzel, mi tutora, por su paciencia, dedicación y capacidad para transmitir conocimientos en el fascinante mundo de la optimización binivel. Su apoyo y comprensión han sido fundamentales para la culminación exitosa de este trabajo.

A mi familia, quiero agradecerles especialmente por su infinita paciencia y dedicación en los momentos más difíciles de este proceso. Su apoyo constante y su aliento para no abandonar nunca me han motivado a seguir adelante y alcanzar mis metas.

A mis amigos y compañeros de estudio, gracias por su tiempo, dedicación y palabras de aliento, que enriquecieron esta experiencia académica.

Un agradecimiento especial a mis mascotas, por su compañía y alegría en los momentos desafiantes, y a ChatGPT, cuya asistencia fue invaluable en la redacción y organización de mis ideas.

Este trabajo no habría sido posible sin el soporte y la guía de cada uno de ustedes. Gracias por ser parte fundamental de este capítulo de mi vida.

Opinión del tutor

Opiniones de los tutores

Resumen

El problema de optimización binivel se define como minimizar una función sobre un conjunto determinado por los puntos óptimos de un modelo de programación matemática. La optimización en el nivel inferior depende de las decisiones tomadas en el nivel superior, creando así una relación de interdependencia entre ambos niveles.

Para abordar este problema, se considera la creación de un problema relacionado en el cual el nivel inferior se sustituye por las condiciones necesarias de optimalidad, siendo este un problema con restricciones de complementariedad.

En este trabajo se propone una forma de generar problemas de dos niveles cuyo problema con restricciones de complementariedad (MPEC) tiene un punto estacionario perteneciente a una de las siguientes clases: fuertemente estacionario, M-estacionario o C-estacionario, dependiendo de los multiplicadores.

Palabras clave: Optimización binivel, MPECs, Condiciones necesarias de optimalidad, Punto estacionario.

Abstract

The bilevel optimization problem is defined as minimizing a function on a set determined by the optimal points of a mathematical programming model. The optimization at the lower level depends on the decisions made at the upper level, thus creating an interdependent relationship between both levels.

To address this problem, the lower-level problem is replaced by the necessary optimality conditions and the resulting (relaxed) optimization problem with complementarity constraints is solved.

This work proposes a way to generate two-level problems whose relaxed problem has a stationary point belonging to one of the following classes: strongly stationary, M-stationary, or C-stationary, depending on the multipliers.

Keywords: Bi-level optimization, MPECs, Necessary optimality conditions, Stationary point.

Índice general

Índice de figuras

Ejemplos de código

Introducción

La optimización de dos niveles, un área fundamental en la investigación operativa y la teoría de juegos, presenta desafíos significativos debido a su complejidad inherente. Este tipo de problemas se caracterizan por la interacción entre un líder y un seguidor, donde las decisiones del líder afectan las respuestas del seguidor. Uno de los aspectos más críticos de esta problemática es garantizar la existencia de soluciones óptimas, lo cual se ve complicado por la naturaleza no convexa del problema, incluso cuando las funciones y los conjuntos factibles son convexos. A menudo, los algoritmos utilizados en este contexto solo logran identificar puntos estacionarios o críticos, que no necesariamente representan soluciones locales o globales, ver Dempe y Zemkoho 2020a.

El modelo de dos niveles es:

$$
\begin{array}{ll}
\text{mín}_x & F(x,y) \\
s.a & \left\{ \begin{array}{l} x \in \mathcal{T}, \\ y \in S(x) = \arg\text{mín}_y \{ f(x,y) \quad s.a \quad y \in \mathcal{H}(x) \}, , \\ (x,y) \in \mathcal{M}^0, \end{array} \right.
\end{array}
$$

donde $x \in R^n$, $y \in R^m$, $F : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, $\mathcal{T} \subseteq \mathbb{R}^n$, $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, $\mathcal{H}(x) \subseteq \mathbb{R}^m$, para todo $x \in \mathbb{R}^n$ and $\mathcal{M}^0 \subseteq \mathbb{R}^{n+m}$. En otras palabras, el problema de optimización binivel se centra en que el líder (nivel superior) debe tomar decisiones (x) que optimicen su objetivo $F(x,y)$, anticipando que el seguidor (nivel inferior) responderá de manera óptima con respecto a su propio objetivo $f(x,y)$, dado el valor de x elegido por el líder. Esta interacción jerárquica entre ambos niveles añade una gran complejidad al problema en comparación con los problemas de optimización de un solo nivel.

Los problemas de optimización de dos niveles son muy utilizados para modelar y analizar mercados eléctricos complejos, ofreciendo una perspectiva única sobre las interacciones estratégicas entre diversos agentes económicos. En el trabajo de Aussel, Cervinka y Marechal 2016 se desarrolló un modelo innovador que aborda los mercados de electricidad desregulados. Su enfoque se distingue por incorporar restricciones de producción y pérdidas térmicas, lo que permite una modelización más precisa y realista. Mediante el uso de modelos binivel, los investigadores pueden explorar escenarios más complejos y representativos del funcionamiento real de los mercados energéticos. Otro ejemplo en esta línea se encuentra en Aussel, Bendotti y Pistek 2017, se profundiza en el análisis de mercados de electricidad de pago por oferta, explorando cómo un productor puede ajustar su estrategia considerando las acciones de sus competidores. El estudio destaca la aplicación de conceptos de equilibrio de Nash y técnicas de mejor respuesta, proporcionando una metodología sofisticada para optimizar la participación de un productor en el mercado.

Los modelos binivel también tiene aplicaciones fundamentales en la selección de hiperparámetros en aprendizaje automático, como lo demuestra el trabajo de Dempe y Zemkoho 2020b. El capítulo 6 del libro aborda la optimización de hiperparámetros en problemas de clasificación y regresión, presentando algoritmos innovadores para manejar funciones objetivo no suaves y no convexas. La razón del uso de esta radica en su capacidad para minimizar errores en modelos complejos, mejorando así la precisión general del aprendizaje automático. Además, se implementan algoritmos especializados para abordar problemas no convexos.

La optimización de dos niveles es una herramienta clave en el diseño y operación de redes industriales sostenibles. Ejemplos notables se incluyen en los estudios de Ramos, Boix et al. 2016 donde se optimiza el uso del agua a partir de modelar la situación mediante juegos de múltiples líderes-seguidores, priorizando objetivos ambientales y económicos. Los resultados mostraron que las empresas participantes lograron beneficios significativos con las formulaciones **KKT** (Karush-Kuhn-Tucker) del modelo **MLFG** (Multi-Leader-Follower Game) utilizado. Además, en Ramos, Boix et al. 2016 se destaca la influencia de la estructura del juego en la configuración óptima, sugiriendo la necesidad de un diseño óptimo para cada planta dentro del EIP. Los enfoques **SLMFG** (Single-Leader-Multifollower Game) y **MLSFG** (Multi-Leader-Single-Follower Game) presentan variaciones en el rol de los participantes: en SLMFG, las empresas son seguidoras y la autoridad es líder, mientras que en MLSFG, ocurre lo contrario. Se resalta que el enfoque MLFG logra equilibrar objetivos económicos y ambientales, generando ahorros significativos mediante la reutilización de recursos. Los resultados indican una reducción en el consumo de agua fresca gracias a las estrategias implementadas, utilizando herramientas como GAMS para modelar los problemas de optimización, ver Ramos, Boix et al. 2016. Además, en Ramos, Rocafull et al. 2018 los autores introducen el concepto de autoridad ambiental en el diseño de redes de servicios públicos, utilizando juegos de múltiples líderes-seguidores y reformulaciones KKT. En el ámbito del despacho energético bajo restricciones de carbono, en Gu et al. 2020 se modela incentivos de precios de energía en un parque industrial, demostrando que un enfoque binivel puede simultáneamente mejorar el impacto ambiental y los beneficios económicos, utilizando un procedimiento iterativo primal-dual.

Además estudios como los de Bhavsar y Verma 2021 investigan sobre la aplicación de una política de subsidios para gestionar el riesgo de materiales peligrosos en una red de transporte ferroviario. En este modelo, el gobierno actúa como líder, ofreciendo subsidios para incentivar al operador ferroviario (el seguidor) a usar rutas alternativas que eviten los enlaces de alto riesgo en la red, utilizando el enfoque **SLSF** (Single-Leader-Single-Follower). Los autores utilizan una reformulación de KKT para resolver el problema y aplican su método a un caso real en los Estados Unidos. Se demuestra que incluso subsidios modestos pueden resultar en una reducción significativa del riesgo.

El estudio de los problemas de dos niveles es de interés de la comunidad científica no solo porque modelan las situaciones antes mencionadas, sino porque es complejo obtener propiedades de sus soluciones así como su cálculo numérico. El concepto mismo de solución del problema binivel es complejo. La decisión del líder es solo respecto a un grupo de variables, mientras que las otras influyen en la función objetivo, pero no son decisión de él. Si para un mismo valor de las variables del líder el problema del seguidor tiene diferentes soluciones óptimas, el valor de la función objetivo del líder no estará determinado, sino que depende de cuál de los óptimos escogió el otro agente.

Para obtener las condiciones de optimalidad y los algoritmos de solución de los modelos binivel se reportan dos enfoques fundamentales. En Dempe y Zemkoho 2020a se usa la función valor extremal. Esto significa que para todo valor de x, se considera la minimización de la función objetivo del líder en el conjunto dado por las restricciones de ambos individuos y la condición de que la función objetivo del seguidor es menor o igual que el valor más pequeño que alcanza en el conjunto de soluciones factibles del seguidor.

Otro enfoque clásico consiste en sustituir el problema del nivel inferior por la condición de KKT, lo que permite transformar problemas de optimización binivel en programas matemáticos con restricciones de equilibrio (MPEC), facilitando así su resolución, ver Caselli et al. 2024; Dempe y Freiberg 2003; Dempe y Zemkoho 2020a. Conocido como *enfoque KKT* en la literatura, es una de las formas más utilizada para la resolución de problemas de dos niveles en la actualidad, ver Bouza-Allende, Aussel et al. 2021.

Dado que los problemas de optimización de ese tipo son inherentemente difíciles de resolver debido a su naturaleza **NP-hard**, ver Bard 1991; Jeroslow 1985 o incluso $\Sigma P2-hard$, ver Cerulli 2021, diciembre; Dempe y Zemkoho 2020a, se han desarrollado diversos enfoques para abordar su complejidad computacional. Sin embargo, estos enfoques suelen ser computacionalmente intensivos para problemas de gran escala, ver Cerulli 2021, diciembre. En paralelo, los algoritmos metaheurísticos, como los evolutivos, han ganado relevancia al proporcionar aproximaciones eficientes en casos no lineales o no convexos, donde las soluciones exactas son inalcanzables en tiempos razonables, ver Sinha et al. 2017. Otro enfoque destacado es el uso de métodos de descomposición, los cuales dividen el problema en subproblemas más manejables que pueden resolverse iterativamente, ver Floudas y Pardalos 1990.

Estas técnicas son particularmente útiles en aplicaciones prácticas, como los mercados de energía o los modelos de sostenibilidad, ver Siddiqui y Gabriel 2012. A pesar de estos avances, existen desafíos abiertos. La escalabilidad sigue siendo un problema crítico, ya que el crecimiento exponencial de las opciones en problemas de gran tamaño limita la aplicabilidad de los métodos exactos, ver Dempe y Zemkoho 2020a. Asimismo, los problemas no convexos carecen de garantías de convergencia hacia el óptimo global, lo que los hace especialmente difíciles de abordar. Finalmente, la incorporación de incertidumbre en los modelos agrega una capa adicional de complejidad, lo que demanda nuevos enfoques híbridos que combinen algoritmos exactos y heurísticos para mejorar la eficiencia computacional sin sacrificar la calidad de las soluciones, ver Cerulli 2021, diciembre; Sinha et al. 2017. Estos avances y desafíos reflejan la importancia de diseñar algoritmos personalizados que aprovechen las estructuras particulares de cada problema binivel. Las aplicaciones industriales, como el diseño de redes ecoindustriales y la gestión de mercados energéticos, destacan la necesidad de enfoques que equilibren precisión y tiempo de cálculo, haciendo de la optimización de dos niveles un área de investigación activa con un impacto significativo en la práctica.

Los algoritmos que se basan en las condiciones KKT incluyen una variedad de métodos, como técnicas de branch-and-bound y métodos de suavizado, ver Dempe y Zemkoho 2020a. Además, se emplean algoritmos SQP (Sequential Quadratic Programming) para resolver problemas suaves con restricciones al que se le aplica el enfoque KKT. Los problemas de optimización de dos niveles y la formulación KKT son inherentemente no convexos, lo que implica que los métodos de optimización convexa no son directamente aplicables. Esta no convexidad puede llevar a soluciones subóptimas y a dificultades para encontrar una solución global. Aunque en muchos casos las funciones involucradas en la definición del modelo sean convexas, la estructura general del problema sigue siendo no convexa.

En resumen, obtener garantías sobre soluciones en problemas de optimización de dos niveles es un desafío complejo debido a que por su no convexidad inherente, se presentan dificultades para escapar de óptimos locales, la posible falta de unicidad y la inestabilidad en las soluciones. Los algoritmos frecuentemente hallan puntos estacionarios, que no siempre corresponden a las soluciones óptimas deseadas. Por lo tanto, es crucial desarrollar métodos especializados que aborden estos problemas y permitan encontrar soluciones globales o aproximaciones adecuadas, ver Dempe y Zemkoho 2020a.

En este contexto, se ha estudiado la estructura genérica de los problemas de complementariedad (MPCC) que surgen del enfoque KKT y Fritz-John (FJ) aplicado a un problema de dos niveles. Se ha demostrado que, para una clase amplia de estos, la condición de que independencia lineal (MPCC-LICQ) se cumple en todos los puntos factibles. Sin embargo, las condiciones de complementariedad estricta (MPCC-SC) y las condiciones de segundo orden (MPCC-SOC) pueden fallar en puntos críticos (estacionarios), incluso en situaciones genéricas, ver Bouza-Allende y Still 2012. Esta situación complica aún más la obtención de garantías sobre la solución.

Es importante señalar que existen casos singulares donde los puntos estacionarios pueden ser problemáticos, especialmente cuando el multiplicador (α) asociado a la condición de KKT del problema de nivel inferior es igual a cero. En tales circunstancias, la condición MPCC-SC puede no cumplirse, lo que podría llevar a que el método KKT no funcione adecuadamente, ver Bouza-Allende y Still 2012.

Basándose en la caracterización de los diferentes tipos de puntos estacionarios establecida por Flegel y Kanzow 2003, esta tesis propone desarrollar un generador de problemas que, dado un punto y las funciones que definen un problema de dos niveles, agregarles funciones polinomiales de primer o segundo grado de forma tal que el punto inicial dado sea un punto crítico del problema creado. Este generador facilitará el estudio del comportamiento de algoritmos conocidos en problemas con ahora al menos un punto estacionario conocido. El usuario además podrá decidir si quiere un punto crítico con multiplicadores arbitrarios o si $\alpha = \mathbf{0}$, lográndose estudiar las clases de puntos críticos que aparecen en el caso genérico, o sea en un clase amplia y significativa de los problemas generados.

La tesis está compuesta de 3 capítulos. Luego del capítulo de introducción, se mostrará la notación que se empleará, se define el problema de dos niveles con un líder y un seguidor, y se explica la teoría matemática para su transformación en un problema MPEC, así como los algoritmos de Julia que se utilizarán en ella. En el tercer capítulo se explicará la implementación algorítmica propuesta anteriormente y su correcta utilización. En el cuarto capítulo se analizarán los resultados obtenidos por el algoritmo propuesto y su comparación con algoritmos implementados en el entorno Julia. Finalmente, se presentarán las conclusiones y recomendaciones del trabajo realizado.

Capítulo 1

Preliminares

La optimización binivel es un problema de optimización donde un subconjunto de variables debe ser la solución óptima de otro problema de optimización, parametrizado por las variables restantes. Este problema tiene dos niveles jerárquicos de decisión: el nivel superior (líder) y el nivel inferior (seguidor). Este tiene dos características principales: primero, el problema del nivel inferior actúa como una restricción para el nivel superior; segundo, la solución del nivel inferior depende de las variables del nivel superior, creando una interdependencia entre ambos niveles. Por ello, el líder debe anticipar la respuesta óptima del seguidor al tomar decisiones. En términos abstractos, la optimización binivel busca minimizar una función objetivo de nivel superior, $F(x, y)$, donde x son las variables de decisión del líder y y son las variables del seguidor.

En este capítulo se abordará los conocimientos necesarios para el desarrollo de esta tesis. La primera seccioón trata sobre Optimización Binivel: la definición de solución basada en la existencia de una cierta cooperación entre lideres y seguidores (caso optimista) y la reformulación KKT. Luego se presentan los conceptos fundamentales de solución, regularidad y estacionariedad para Problemas Matemáticos con Restricciones de Equilibrio (MPEC). La tercera sección contiene las principales transformaciones que se aplican a las restricciones de complementariedad para la solución numérica del modelo. Por último, se presenta la modelación de los problemas binivel en el lenguaje de programación Julia y los métodos seleccionados implementados en este lenguaje para su resolución.

1.1. Optimización Binivel

El problema de optimización binivel en general se define de la siguiente manera

$$\begin{aligned} &\text{mín}_x \quad F(x,y) \\ &s.a \begin{cases} x \in \mathcal{T} \\ y \in S(x) = \arg\text{mín}_y \{f(x,y) \quad s.a \quad y \in \mathcal{H}\} \\ (x,y) \in \mathcal{M}^0 \end{cases} \end{aligned} \tag{1.1}$$

En la mayoría de las aplicaciones, los problemas de optimización del nivel inferior y superior son modelos de programación matemática, por lo que, sin pérdida de generalidad, se asume que $\mathcal{T} = \mathbb{R}^n$, $\mathcal{H}$ es $\{y \in \mathbb{R}^m \in v_j(x,y) \leq 0,\ j = 1,\ldots,q\}$ y $\mathcal{M}^0 = \{(x,y) \in \mathbb{R}^{n+m},\ g_i(x,y) \leq 0,\ i = 1,\ldots,q\}$. El problema es

$$\begin{aligned} &\min_x \ F(x,y) \\ &\text{sujeto a} \\ &g_i(x,y) \leq 0,\ i = 1,\ldots,q, \\ &y \in \underset{y}{\operatorname{argmin}} \{f(x,y) \mid v_j(x,y) \leq 0,\ j = 1,\ldots,q\} \end{aligned} \tag{1.2}$$

donde $x \in R^n$, $y \in R^m$, $F(x,y) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, $g_i(x,y) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R},$, $i = 1,\ldots,q$, $f(x,y) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, $v_j(x,y) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, $j = 1,\ldots,s$. En el enfoque optimista, se asume que el seguidor, que actúa en el nivel inferior, elegirá la solución más favorable para el líder, quien toma decisiones en el nivel superior. Este es considerado más tratable y, en ciertas situaciones favorables, puede simplificarse a un problema convexo. Además, en el contexto de múltiples objetivos, el enfoque optimista permite alcanzar el mejor frente de Pareto posible, ver Dempe y Zemkoho 2020a.

$$\begin{gathered} \text{mín}_x \ \text{ínf}_{y \in S(x)} F(x,y) \\ \text{con } S(x) = \operatorname{argmin}_y \{f(x,y) \mid v(x,y) \leq 0\} \end{gathered} \tag{1.3}$$

Figura 1.1: Problema de optimización bajo el enfoque optimista.

Por otro lado, el enfoque pesimista asume que el seguidor seleccionará la opción menos favorable para el líder entre las soluciones óptimas disponibles, el cual es más complejo de resolver y puede incluso no tener solución. A menudo, se requieren reformulaciones para abordar estos problemas, lo que lo convierte en un reto teórico y computacional significativo en situaciones de múltiples objetivos, conduce al peor frente de Pareto posible, ver Sinha et al. 2017.

$$\begin{aligned} &\min_x \sup_{y \in S(x)} F(x,y) \\ &\text{con } S(x) = \underset{y}{\operatorname{argmin}} \{f(x,y) \mid v(x,y) \leq 0\} \end{aligned} \tag{1.4}$$

Figura 1.2: Problema de optimización bajo el enfoque pesimista.

Es relevante destacar que la mayoría de la literatura se centra en el enfoque optimista debido a su mayor facilidad de tratamiento. Sin embargo, el otro también tiene su utilidad, especialmente en la modelación de situaciones donde se considera la aversión al riesgo, ver Dempe y Zemkoho 2020a. En este contexto, los términos "líder" y "seguidor" se utilizan para describir los roles en el modelo a optimizar; el líder toma decisiones considerando las posibles reacciones del seguidor, quien a su vez reacciona seleccionando su mejor opción, ver Sinha et al. 2017.

Dado que en la tesis trataremos sobre problemas binivel de enfoque optimista mostraremos algunos resultados referidos al modelo resultante.

Primeramente es importante notar el siguiente resultado, ver Schmidt y Beck 2021.

Proposition 1 *El problema de optimización binivel optimista* (1.3) *se formula equivalentemente como:*

$$\begin{array}{cc} \min_{x,y} & F(x,y) \\ s.t. & g_i(x,y) \leq 0, i = 1 \ldots q, \\ & y \in S(x), \end{array}$$

donde $S(x)$ es el conjunto de soluciones óptimas del problema parametrizado por x

$$\min_{y \in Y(x)} \quad f(x,y) s.t. \quad v_j(x,y) \leq 0, j = 1 \ldots s,$$

De ahí que a partir de ahora se considerará el problema (1.5) y se le llamará indistintamente problema de dos niveles. Los problemas de dos niveles pueden ser reformulados en un problema de un solo nivel al reemplazar el problema del nivel inferior por las condiciones KKT de este en las restricciones del primer nivel.

$$\begin{array}{ll} \min\limits_{x,y,\lambda_j} & F(x,y) \\ s.a & \begin{cases} g_i(x,y) \leq 0, i = 1 \ldots q, \\ \nabla_y f(x,y) + \sum_{j=1}^{s} \nabla_y v_j(x,y) \lambda_j = 0, \\ v_j(x,y) \leq 0, j = 1 \ldots s, \\ v_j(x,y) \lambda_j = 0, j = 1 \ldots s, \\ \lambda_j \geq 0, j = 1 \ldots s. \end{cases} \end{array} \tag{1.5}$$

Los tres últimos grupos de restricciones expresan que v y λ están restringidas en signo y que al menos una es 0. Estas condiciones son conocidas como **restricciones de complementariedad**. Estos modelos corresponden a la clase de problemas de programación matemática con restricciones de complementariedad (MPEC). Como se mencionó en la introducción, existe un enfoque alternativo mediante un problema de programación matemática utilizando la función del valor extremal. Sin embargo, esta función carece de una expresión explícita, exceptuando casos muy específicos. La ventaja del método KKT para MPEC radica en que proporciona una forma explícita, aunque requiere ciertas condiciones de regularidad del conjunto, como la Condición de Calificación de Restricciones de Independencia Lineal (LICQ), por sus siglas en ingles, de las restricciones activas. En caso de no cumplirse estas condiciones, podrían existir puntos óptimos que no satisfagan las condiciones KKT. A continuación, presentamos los resultados de esta área necesarios para el desarrollo de esta tesis.

1.2. Programación Matemática con Restricciones de Equilibrio (MPEC)

Un **problema de Programación'on Matemático con Restricciones de Equilibrio (MPEC)** es un tipo de problema de optimización no lineal que incluye restricciones de equilibrio, específicamente, restricciones de complementariedad.

$$
\begin{array}{ll}
\text{mín} & f(z) \\
\text{s.t.} & \begin{array}{rcll} g_i(z) & \leq & 0, & i=1,\dots,q, \\ h_k(z) & = & 0, & k=1\dots,m, \\ G_j(z) & \geq & 0, & j=1,\dots,s, \quad H_j(z) \;\; \geq \;\; 0, \quad j=1,\dots,s, \end{array} \\
& G_j(z)^T H_j(z)=0,\ j=1,\dots,s. \\
& \text{Definición de MPEC}
\end{array}
\tag{1.6}
$$

donde $f:\mathbb{R}^{\hat{n}}\to\mathbb{R}$, $g:\mathbb{R}^{\hat{n}}\to\mathbb{R}$, $h:\mathbb{R}^{\hat{n}}\to\mathbb{R}$, $G:\mathbb{R}^{\hat{n}}\to\mathbb{R}$, y $H:\mathbb{R}^{\hat{n}}\to\mathbb{R}$ son funciones continuamente diferenciables. El término de complementariedad se debe a la existencia de las restricciones homónimas

$$
G_j(z)\geq 0,\ H_j(z)\geq 0, \quad G_j(z)^T H_j(z)=0, \quad j=1,\dots,s. \tag{1.7}
$$

Debido a las restricciones de complementariedad, los MPEC no cumplen con las condiciones de regularidad estándar, lo que hace que las condiciones de KKT no sean directamente aplicables como condiciones de optimalidad de primer orden. Los MPEC son utilizados para modelar problemas donde existen restricciones de equilibrio, como problemas de ingeniería y economía, ver Dempe y Zemkoho 2020a; Flegel y Kanzow 2003.
A continuación como en Flegel y Kanzow 2003, se introduce las siguientes definiciones, útiles para la obtención de condiciones necesarias de optimalidad.

Definición 1.1 (Conjunto de Índices) *Dado un vector factible z^* del MPEC*

(1.6)*, definimos los siguientes conjuntos de índices:*

$$\begin{aligned} J_G := & \quad J_G(z^*) := \{i | G_i(z^*) = 0, H_i(z^*) > 0\} \\ J_{GH} := & \quad J_{GH}(z^*) := \{i | G_i(z^*) = 0, H_i(z^*) = 0\} \\ J_H := & \quad J_H(z^*) := \{i | G_i(z^*) > 0, H_i(z^*) = 0\} \end{aligned} \tag{1.8}$$

Para definir condiciones de regularidad para MPEC, introducimos el siguiente problema, dependiente de z^*:

Definición 1.2 (Programa No Lineal Relacionado (RNLP)) *Un Programa No Lineal Relacionado* $RNLP_{a,b} := RNLP_{a,b}(z^*)$ *es:*

$$\begin{array}{ll} \text{mín} & f(z) \\ s.t. & \begin{array}{rclll} g_i(z) & \leq & 0, & i = 1, \ldots, q, & \\ h_k(z) & = & 0, & k = 1 \ldots, m, & \\ G_j(z) & = & 0, & j \in J_G \cup J_a, & G_j(z) \ \geq \ 0, \quad j \in J_H, \\ H_j(z) & = & 0, & j \in J_H \cup J_b, & H_j(z) \ \geq \ 0, \quad j \in J_G, \end{array} \end{array} \tag{1.9}$$

Problema No Lineal Relacionado (RNLP)

donde $a \cup b = J_{GH}$.

El RLNP (1.9) puede usarse para definir variantes de regularidad y condiciones necesarias de optimalidad adecuadas para MPEC como son la restricción estándar de independencia lineal (LICQ en su forma abreviada) y la de punto de KKT .

Definición 1.3 (MPEC-LICQ) *El MPEC* (1.6) *se dice que satisface la MPEC-LICQ en un vector factible* z^* *si los correspondientes* $RNLP_{a,b}(z^*)$ *satisfacen la LICQ en ese vector* z^**.*

Cabe destacar que si la LICQ se cumple en $RNLP_{a,b}(z^*)$ para una partición (a,b) se cumple para todas.
En el contexto de los MPEC en Flegel y Kanzow 2003 se exponen varios tipos de puntos estacionarios que son cruciales para analizar la optimalidad, los cuales son los siguientes:

Definición 1.4 (Punto Factible) *Un punto factible* z^* *del MPEC se llama débilmente estacionario si existe un multiplicador de Lagrange* $(\mu, \alpha, \beta, \gamma)$ *tal que se cumplen las siguientes condiciones:*

$$
\begin{aligned}
&\nabla f(z^*)+\sum_{i=1}^{q}\mu_i\nabla g_i(z^*)+\sum_{k=1}^{q_0}\alpha_k\nabla h_k(z^*)-\sum_{i=1}^{s}[\beta_j\nabla G_j(z^*)+\gamma_j\nabla H_j(z^*)]=\mathbf{0}\\
&\quad\beta_j,\ j\in J_G \text{ libre},\quad \beta_j,\ j\in_{J_G\cup J_{GH}} \text{ libre},\quad \beta_j,\ j\in_{J_H}=0,\\
&\quad\gamma_j,\ j\in_{J_H} \text{ libre},\quad \gamma_j,\ j\in_{J_G\cup J_{GH}} \text{ libre},\quad \gamma_j,\ j\in_{J_G}=0,\\
&\quad g_i(z^*)\leq 0,\quad \mu_i\geq 0,\quad \mu_i g_i(z^*)=0, i=1,\ldots,q.
\end{aligned}
\tag{1.10}
$$

Este concepto de estacionariedad es equivalente al cumplimiento de la estacionariedad clásica en el problema $RNLP_{J_{GH},J_{GH}}$. De ahí que es una condición relativamente débil. Existen conceptos más fuertes de estacionariedad que se derivan y estudian en otros lugares. En particular, se tienen las siguientes definiciones:

Definición 1.5 (Punto C-estacionario) *El punto z^* es: C-estacionario si, para cada $i\in J_{GH}$, $\beta_j\gamma_j\geq 0$ se cumple.*

Definición 1.6 (Punto M-estacionario) *El punto z^* es: M-estacionario si, para cada $i\in J_{GH}$, o bien $\beta_j,\gamma_j>0 \vee \beta_j\gamma_j=0$.*

Definición 1.7 (Punto Fuertemente estacionario) *El punto z^* es: fuertemente estacionario si, para cada $i\in J_{GH}$, $\beta_j,\gamma_j\geq 0$.*

La C y la M estacionariedad son condiciones necesarias de optimalidad bajo ciertas condiciones de regularidad, más débiles que la MPEC-LICQ. La fuerte conlleva al siguiente resultado:

Teorema 1.1 *Sea $z^*\in\mathbb{R}^n$ un mínimo local del MPEC* (1.6)*. Si MPEC-LICQ se cumple en z^*, entonces existe un único multiplicador de Lagrange tal que $(z^*,\mu,^*)$ es fuertemente estacionario.*

O sea el punto z^* es un punto estacionario del problema relajado

$$
\begin{array}{ll}
\text{mín} & f(z)\\
\text{s.t.} & g_i(z)\leq 0,\ i=1,\ldots,q,\\
& h_k(z)=0,\ k=1\ldots,m,\\
& G_j(z)\ =\ 0,\ \ j\in J_G\quad G_j(z)\ \geq\ 0,\ \ j\in J_{GH},\quad G_j(z)\ \geq\ 0,\ \ j\in J_H,\\
& H_j(z)\ =\ 0,\ \ j\in J_H\quad H_j(z)\ \geq\ 0,\ \ j\in J_{GH},\quad H_j(z)\ \geq\ 0,\ \ j\in J_G,\\
& \text{Problema No Lineal Relajado}
\end{array}
\tag{1.11}
$$

Los algoritmos buscan al menos converger a puntos de este tipo. En la próxima sección se revisarán las reformulaciones que se resuelven las bibliotecas de Julia.

1.3. Métodos de Reformulación para Optimización Binivel

La optimización binivel presenta desafíos particulares debido a su naturaleza jerárquica y las condiciones de complementariedad resultantes. A continuación, se presentan los principales métodos de reformulación implementados en la literatura, basados en la transformación del MPEC (1.5).

1.3.1. Método Big-M

El método Big-M (Fortuny-Amat y McCarl) es una técnica fundamental para reformular problemas de optimización binivel en problemas MPEC. Este método aborda específicamente las condiciones de complementariedad que surgen en estas reformulaciones, transformando el problema original en un problema de programación lineal mixta entera (MILP).

La reformulación mediante Big-M introduce un parámetro M suficientemente grande y variables binarias para transformar las condiciones de complementariedad no lineales en restricciones lineales. Para cada condición de complementariedad $v_j(x,y)\lambda_j = 0$ en (1.5), el método introduce una variable binaria $\delta_j \in \{0,1\}$ y cotas superiores M_p, M_d bajo las siguientes restricciones:

$$
\begin{aligned}
v_j(x,y) &\geq -M_p(1-\delta_j) \\
\lambda_j &\leq M_d\delta_j \\
\delta_j &\in \{0,1\}
\end{aligned}
\tag{1.12}
$$

donde M_p y M_d son valores grandes para las variables primales y duales, respectivamente. La efectividad del método depende crucialmente de la selección apropiada de estos valores, que deben ser suficientemente grandes para no excluir la solución óptima, pero no excesivamente grandes para evitar inestabilidades numéricas Garcia et al. 2022.

1.3.2. Método SOS1

El método de Conjuntos Ordenados Especiales tipo 1 (SOS1) evita el uso de parámetros Big-M mediante restricciones de tipo conjunto. Para cada par complementario (v_j, λ_j), $j = 1, \ldots, s$:

$$
\begin{aligned}
v_j(x,y) &= s_j \\
[s_j; \lambda_j] &\in \text{SOS1}
\end{aligned}
\tag{1.13}
$$

donde $SOS1 = \{(a,b) \in \mathbb{R} : a,b \geq 0,\ ab = 0\}$
Esta restricción fuerza que al menos una variable en el par sea cero, preservando la no linealidad original sin necesidad de cotas. Note que SOS1 es un cono. Esta es una reformulación del MPEC como un problema de optimización conica. Es particularmente eficaz en MPECs con restricciones del tipo $u \in K$, donde K es un cono, pues sigue siendo un problema de optimización cónica y se pueden aplicar algoritmos específicos para la resolución de este tipo de modelos, Garcia et al. 2022. La dificultad principal es que el cono es no convexo. Una aplicación de este enfoque se encuentra en Siddiqui y Gabriel 2012.

1.3.3. Método ProductMode

El método ProductMode representa un enfoque directo para manejar las condiciones de complementariedad en su forma de producto original. Este método es particularmente útil cuando se trabaja con solucionadores de programación no lineal (NLP), aunque no garantiza la optimalidad global.
La implementación del ProductMode mantiene la restricción de complementariedad en su forma original:

$$v_j(x,y) \cdot \lambda_j \leq t \tag{1.14}$$

donde $t > 0$ es un parámetro de regularización pequeño. Esta formulación, aunque no satisface las condiciones de calificación de restricciones estándar, es útil para obtener soluciones iniciales y puede ser especialmente efectiva cuando se combina con solucionadores NLP, ver Garcia et al. 2022. sin embargo, bajo condiciones naturales, la sucesión que genera converge a puntos C estacionarios y bajo regularidad hasta fuertemente estacionarios, ver **scholtes12**.
La Tabla 1.1 resume los requisitos y características de cada método:

Tabla 1.1: Comparación de métodos de reformulación

Método	**Solver Requerido**	**Ventajas**
Big-M	MIP	Estabilidad numérica controlada
SOS1	MIP con SOS1	Sin parámetros ad-hoc
ProductMode	NLP	Manejo de no linealidades

Para casos con restricciones no lineales $v_j(x,y)$, se recomienda combinar ProductMode con técnicas de linealización por tramos Garcia et al. 2022, Apéndice B. Todos estos métodos están implementados en BilevelJuMP.jl, permitiendo experimentación con reformulaciones que se pueden encontrar en Garcia et al. 2022, Sección 4.

1.4. Modelación en Julia

El lenguaje de programación Julia se ha posicionado como una herramienta revolucionaria en el ámbito de la computación científica y la optimización matemática. Desarrollado en el MIT, Julia fue diseñado desde sus cimientos para abordar el denominado "problema de los dos lenguajes", donde tradicionalmente los investigadores se veían obligados a prototipar en un lenguaje de alto nivel como Python y luego reimplementar en C++ o Fortran para obtener rendimiento óptimo. Julia logra combinar la facilidad de uso de lenguajes dinámicos como Python con el rendimiento de lenguajes compilados como C, gracias a su sistema de tipos y su compilador LLVM. Una de las características más destacadas de Julia es su sintaxis expresiva y natural para la formulación matemática. El lenguaje permite escribir código que se asemeja notablemente a las expresiones matemáticas tradicionales, facilitando la traducción directa de modelos matemáticos a código ejecutable. Esta característica es particularmente valiosa en el campo de la optimización, donde la claridad en la expresión de modelos matemáticos es crucial. Julia alcanza velocidades de ejecución comparables a C y Fortran, superando significativamente a Python en cálculos numéricos intensivos, con mejoras de rendimiento que pueden alcanzar órdenes de magnitud en determinadas aplicaciones. La interoperabilidad de Julia con otros lenguajes de programación constituye una ventaja significativa para proyectos de optimización complejos. A través de paquetes como PyCall, RCall, y JavaCall, Julia puede integrarse perfectamente con bibliotecas establecidas en Python, R y Java. Esta capacidad permite aprovechar el extenso ecosistema de estos lenguajes mientras se mantiene el rendimiento superior de Julia en los cálculos críticos de optimización. Además, Julia puede llamar directamente a funciones de C y Fortran sin overhead adicional, permitiendo la reutilización de código legacy optimizado y la integración con sistemas empresariales existentes. El ecosistema de optimización en Julia está encabezado por JuMP (Julia for Mathematical Programming), un lenguaje de modelado algebraico embebido que permite expresar problemas de optimización de manera natural y eficiente. JuMP se destaca por su capacidad para manejar problemas de gran escala y su integración perfecta con diversos solucionadores, tanto comerciales como de código abierto. La extensión BilevelJuMP amplía estas capacidades al dominio de la optimización binivel, permitiendo la formulación y resolución de problemas jerárquicos complejos, un área de creciente importancia en la investigación de optimización moderna.
Julia facilita la integración con solucionadores de alto rendimiento como Ipopt para optimización no lineal, SCIP para programación entera mixta, y HiGHS para programación lineal. La interfaz nativa de Julia con estos solucionadores minimiza el overhead de comunicación, resultando en tiempos de resolución significativamente menores comparados con interfaces basadas en Python. Por ejemplo, en problemas de optimización no lineal de gran escala, la combinación de JuMP con Ipopt puede

ejecutarse hasta 10 veces más rápido que implementaciones equivalentes en Python utilizando Pyomo o AMPL, lo que demuestra la eficiencia del ecosistema de Julia en aplicaciones prácticas.
La capacidad de Julia para escalar eficientemente en entornos de computación distribuida y GPUs es particularmente relevante para problemas de optimización de gran escala. El lenguaje incluye soporte nativo para computación paralela y distribuida, permitiendo la paralelización de algoritmos de optimización con mínimas modificaciones al código. Paquetes como DistributedArrays y CUDA.jl facilitan la implementación de algoritmos de optimización en clusters y GPUs, respectivamente, manteniendo la sintaxis clara y expresiva característica de Julia.
En el ámbito específico de la optimización matemática, Julia presenta ventajas significativas sobre Python, particularmente en términos de rendimiento y expresividad. Mientras que Python requiere de bibliotecas como Numpy y Scipy para operaciones numéricas eficientes, Julia proporciona estas capacidades de manera nativa. En problemas de optimización complejos, las implementaciones en Julia con JuMP típicamente requieren menos código y se ejecutan más rápidamente que sus equivalentes en Python utilizando Pyomo o CVXPY. Estudios comparativos han demostrado que Julia con JuMP puede resolver problemas de optimización típicos entre 2 y 10 veces más rápido que implementaciones equivalentes en Python, dependiendo de la naturaleza y escala del problema. Esta ventaja se hace más pronunciada en problemas que requieren múltiples evaluaciones de la función objetivo o restricciones complejas, atribuyéndose principalmente a la compilación Just-In-Time de Julia, la ausencia de overhead en la llamada a funciones, y la integración más eficiente con los solucionadores de optimización.
Por los argumentos anteriores en este trabajo utilizaremos este lenguaje de programación con principalmente los módulos **JuMP** y **BilevelJuMP**, de los cuales se describe a continuación.

JuMP

JuMP (Julia for Mathematical Programming) es una biblioteca de modelado algebraico integrada en Julia que permite formular y resolver problemas de optimización matemática de manera eficiente y expresiva. Esta biblioteca está diseñada para abordar problemas de programación matemática que pueden expresarse en la forma general:

$$\min_{z} F(z) \tag{1.15}$$

sa:

$$\begin{aligned} g_i(z) &\leq 0, \quad i = 1, \ldots, q, \\ h_k(z) &= 0, \quad k = 1, \ldots, q_0, \end{aligned} \tag{1.16}$$

donde $z \in \mathbb{R}^p \times \mathbb{Z}^{\hat{n}-p}$ representa el vector de variables de decisión, que puede incluir tanto componentes continuos como discretos. La función $F(z)$ representa el objetivo a minimizar, mientras que $g_i(z)$ y $h_k(z)$ representan las restricciones de desigualdad e igualdad, respectivamente.

JuMP proporciona una interfaz intuitiva y matemáticamente rigurosa para la construcción de estos modelos mediante un conjunto de macros especializadas. La macro **@variable** se utiliza para declarar las variables de decisión, permitiendo especificar su naturaleza (continua o discreta) y sus dominios. Por ejemplo, podemos declarar variables continuas no negativas o variables enteras acotadas.

La función objetivo se define mediante la macro **@objective**, que acepta expresiones lineales, cuadráticas o no lineales. JuMP distingue automáticamente entre estos tipos de expresiones y selecciona los métodos de solución apropiados. La sintaxis es clara y cercana a la notación matemática tradicional, lo que facilita la traducción de modelos matemáticos a código ejecutable.

Para la especificación de restricciones, JuMP ofrece dos macros principales: **@constraint** para restricciones lineales y cuadráticas, y **@NLconstraint** para restricciones no lineales generales. La biblioteca identifica automáticamente la naturaleza de las restricciones y las procesa de manera eficiente. Las restricciones pueden incluir desigualdades, igualdades y expresiones más complejas que involucren funciones no lineales.

Una característica particularmente potente de JuMP es su integración con el paquete **Complementarity.jl**, que permite la formulación de problemas de complementariedad matemática (MPECs). Esto es especialmente útil para problemas de optimización jerárquica o de equilibrio, donde las condiciones de complementariedad son fundamentales. Las restricciones de complementariedad se pueden expresar de manera natural utilizando la sintaxis proporcionada por estas bibliotecas.

Una vez que el modelo está completamente especificado, la función **optimize!** se utiliza para resolver el problema utilizando el solucionador seleccionado. JuMP es compatible con una amplia gama de solucionadores, tanto de código abierto como comerciales, y selecciona automáticamente el más apropiado según la estructura del problema (lineal, cuadrático, no lineal, entero mixto, etc.).

Es importante destacar que JuMP maneja eficientemente la diferenciación automática para problemas no lineales, lo que permite calcular gradientes y Hessianas de manera precisa y eficiente, mejorando significativamente el rendimiento y la robustez de los métodos de optimización utilizados. Para más detalles sobre la biblioteca, ver Lubin et al. 2023.

BilevelJuMP

BilevelJuMP.jl es una extensión especializada de Julia diseñada específicamente para abordar problemas de optimización binivel, también conocidos como problemas de optimización jerárquica o de dos niveles. Esta biblioteca se construye sobre la base de JuMP, aprovechando su sintaxis intuitiva y capacidades de modelado mientras agrega funcionalidades específicas para la formulación y resolución de problemas jerárquicos. Este facilita la modelación de esta jerarquía propia de los problemas binivel permitiendo que el problema del seguidor se formule utilizando la sintaxis familiar de JuMP. Esto significa que los usuarios pueden especificar restricciones lineales y cuadráticas, definir variables tanto continuas como enteras, y trabajar con diferentes tipos de funciones objetivo, manteniendo la claridad y expresividad características de JuMP. De manera análoga, el problema del nivel superior se gestiona con la misma flexibilidad, permitiendo la especificación de sus propias restricciones y función objetivo.
Una característica notable de BilevelJuMP.jl es su capacidad para manejar diferentes reformulaciones de las restricciones de complementariedad en problemas MPEC (Mathematical Programs with Equilibrium Constraints). Los usuarios pueden experimentar con diversas técnicas de reformulación, incluyendo el método SOS1 (Special Ordered Sets of type 1), la reformulación de McCarl (conocida como Big-M) y ProductMode, descritos anteriormente, adaptando la resolución del problema según las necesidades específicas del caso.
La versatilidad de BilevelJuMP.jl se evidencia en su compatibilidad con diversos tipos de solucionadores. La biblioteca puede utilizar tanto solucionadores de programación lineal mixta entera (MIP) como solucionadores de programación no lineal (NLP), seleccionando el más apropiado según las características particulares del problema y la reformulación elegida. Esta flexibilidad permite abordar una amplia gama de problemas binivel con diferentes estructuras y complejidades.
Sin embargo, es importante reconocer las limitaciones inherentes a BilevelJuMP.jl. Este no permite problemas no convexos, además puede encontrar dificultades al manejar problemas altamente no lineales o con estructuras de optimización particularmente complejas que desafían la representación estándar en la sintaxis de JuMP. Algunas restricciones específicas podrían requerir transformaciones adicionales que incrementan la complejidad del modelo, y en casos de problemas de gran escala, el rendimiento del solucionador puede convertirse en un factor crítico que afecte la eficiencia de la resolución.
Adicionalmente, la formulación y resolución de problemas muy específicos o especializados podría no estar completamente optimizada dentro del paquete, lo que podría requerir adaptaciones o consideraciones especiales por parte del usuario. Estas limitaciones, aunque importantes de considerar, no disminuyen la utilidad general de BilevelJuMP.jl como una herramienta poderosa para la optimización binivel, sino que más bien definen el alcance de su aplicabilidad óptima.

Para más detalles sobre la biblioteca, ver Garcia et al. 2022.
Teniendo en cuenta la posibilidad de que los métodos de solución converjan a puntos estacionarios de distinto tipo, resulta interesante comprobar si, conocido un punto de esta naturaleza, el algoritmo logra obtener un punto con mejor evaluación de la función objetivo. Esta es la motivación para el desarrollo del generador de problemas prueba, objeto de esta tesis y que se presentará en el próximo capítulo.

Capítulo 2

Detalles de Implementación y Experimentos

En este capítulo se mostrará la forma y los pasos para generar el problema deseado en el contexto de un generador de puntos estacionarios para problemas binivel. Este capítulo aborda el diseño y la implementación del generador, explicando detalladamente las metodologías empleadas, los criterios necesarios para asegurar la validez de los puntos obtenidos y los algoritmos utilizados en cada etapa del proceso.

2.1. Nomenclatura a utilizar:

Tabla 2.1: Nomenclatura a utilizar

$F(x,y)$	Función del líder.
$g_i(x,y)$	Restricciones de desigualdad del líder $i=1\ldots q$.
$h_i(x,y)$	Restricciones de igualdad del líder $i=1\ldots r$.
$f(x,y)$	Función del seguidor.
$v_j(x,y)$	Restricciones de desigualdad del seguidor $j=1\ldots s$.
$u_j(x,y)$	restricciones de igualdad del seguidor $j=1\ldots t$.
$v_j^\star$	Restricción de desigualdad del seguidor activa después de modificarse el problema $j=1\ldots s$.
z_{est}	Punto que se desea que sea estacionario.
J_0^g	Índices activos del líder.
J_0^v	Índices activos del seguidor.
α	Vector de entrada de dimensión igual cantidad variables seguidor.
μ_i	Multiplicador asociado al $g_i(x,y)$ en el líder.
β_j	Multiplicador asociado al $v_j^\star$ en el líder.
λ_j	Multiplicador asociado al $v_j^\star$ en el seguidor.
γ_j	Valor asociado a cada $v_j^\star$ con $\lambda_j=0$.

2.2. Generación del problema

A continuación se mostrarán los pasos a seguir para generar el problema deseado donde el punto sea estacionario de la clase específica.

2.2.1. Requerimientos de entrada

Para generar un problema es necesario inicialmente introducir un problema de optimización binivel del tipo SLSF, un punto (z_{est}), el cual se desea que sea estacionario y los multiplicadores con signo o valores con ciertas condiciones en dependencia del tipo de estacionariedad requerida.
El problema de entrada tiene que cumplir con ser un programa de optimización binivel como 1.5. Debe introducirse también sus índices activos de las restricciones de desigualdad en cada nivel para z_{est} :

- Nivel Superior:
$$J_0^G = \{i|g_i(x,y) = 0\} \tag{2.1}$$

  donde estos índices activos tienen un μ_i relacionados.

- Nivel Inferior: Se tiene en cuenta con respecto a los valores del multiplicador λ_i y $v_j(x,y)$.

  - $$J_1^v = \{j|v_j(x,y) = 0 \wedge \lambda_j = 0\} \tag{2.2}$$
  - $$J_2^v = \{j|v_j(x,y) = 0 \wedge \lambda_j > 0\} \tag{2.3}$$
  - $$J_3^v = \{j|v_j(x,y) < 0 \wedge \lambda_j = 0\} \tag{2.4}$$

  Cada índice activo tiene multiplicadores β_j y en dependencia del caso γ_j asociados.

Para cada $i \in J_0^G$ (2.1) debe de asignarse su multiplicador $\mu_i \geq 0$ correspondiente. Además se debe conocer el valor del $\boldsymbol{\alpha}$, para el nivel inferior debe tenerse en cuenta si $\alpha = \mathbf{0}$ en cuyo caso afirmativo γ_j es un valor de entrada
Para obtener las diferentes clases de puntos estacionarios debe de introducirse β_j, γ_j, en caso de ser este último valor de entrada tal que, los multiplicadores β_j y γ_j de forma tal que en los indices activos J_1^v (**??**):

- **Punto C-Estacionario:**

$$\beta_j * \ \gamma_j \ >= 0 \tag{2.5}$$

- **Punto M-Estacionario**

$$\begin{aligned} &\beta_j \wedge \gamma_j > 0 \\ &\beta_j \quad \text{Libre} \quad \gamma_j = 0 \\ &\beta_j = 0 \quad \gamma_j \quad \text{Libre} \end{aligned} \tag{2.6}$$

- **Punto Fuertemente Estacionario**

$$\beta_j \wedge \gamma_j \geq 0 \tag{2.7}$$

En caso que $\boldsymbol{\alpha} \neq \mathbf{0}$ se asume que $\gamma_j = 0$ en las condiciones anteriores.

Modificación del Problema Original

Después de tener los datos de entrada necesarios se procede a la modificación del problema de entrada para que este sea estacionario del tipo requerido, z_{est}.
Primero en caso de que $\alpha \neq 0$ en los $v_j(x,y) \in J_0^v$ se procede a calcular el $\boldsymbol{b_j}$ de la siguiente forma:

- $v_j(x,y) \in J_2^v$ **2.3**:

$$\boldsymbol{b_j} = \frac{\alpha \cdot (-\nabla_y v_j(z_{est})^T \cdot \alpha)}{\|\alpha\|_2^2} \tag{2.8}$$

- $v_j(x,y) \in J_1^v \vee J_3^v$ **(??,2.4)**

$$\boldsymbol{b_j} = \frac{\alpha \cdot ((-\nabla_y v_j(z_{est})^T \cdot \alpha) + \gamma_j)}{\|\alpha\|_2^2} \tag{2.9}$$

Después de realizado el calculo del b_j se procede a modificar su índice activo correspondiente:

$$v_j^\star(z_{est}) = v_j(z_{est}) + (\boldsymbol{b_j}^T) \cdot (y_1, y_2, \ldots, y_m) \tag{2.10}$$

Luego en todas las restricciones del nivel inferior y superior se evalúa el punto y se verifica si es factible. En caso de no ser factible bajo las nuevas restricciones se añaden constantes c_i en el caso de las restricciones del nivel superior y c_j para las del inferior. Esto conlleva a que sea un punto factible en el conjunto de restricciones

de ambos niveles sin perjudicar las propiedades relacionadas con la convexidad al ser una adicción de funciones lineales.
Al hacer factible se procede a realizar el KKT del nivel inferior con las modificaciones anteriores evaluado en z_{est}.
Se plantea las condiciones KKT y se halla $\boldsymbol{bf}$:

$$\nabla_y f(x,y) + \sum_{j \in J_2^v} (\lambda_j \nabla_y v_j^\star(x,y)) + \boldsymbol{bf} = \mathbf{0} \tag{2.11}$$

KKT del problema del nivel inferior

Finalmente podemos construir la reformulación en MPEC como 1.5 calculando el $\boldsymbol{BF}$ y obteniendo el problema de forma reformulada.

KKT del MPEC

$$\tag{2.12}$$

$\nabla_{xy} F(x,y) + \sum_{j \in J_o^g|} (\mu_i \nabla_{xy} g(x,y)) + [\nabla_{x,y} \nabla_y f(x,y) + \sum_{j \in J_1^v g} \lambda_j \nabla_{xy} \nabla_y v_j^\star(x,y)]\alpha + \sum_{j \in J_1^v \cup J_2^v|} (\beta_j \nabla_{xy} v_j^\star(x,y)) + \boldsymbol{BF} = \mathbf{0}$

2.3. Implementación Algorítmica y Guía de Usuario

En la implementación computacional del generador propuesto en la sección anteriores se utilizó el lenguaje de programación **Julia**, ver JuliaLang 2025, dado a su versatilidad en expresiones y funciones matemáticas implementadas en su paquete base y sus bibliotecas externas como **BilevelJuMP**, ver Garcia et al. 2022, que permite resolver problemas binivel SLSF lineales y cuadráticos sin tener que transformar el problema original y **JuMP** Lubin et al. 2023 el cual, es una interfaz robusta para la optimización general, todo ello con unas excelentes prestaciones de cómputo. Por ello se brindará una guía de usuario para el uso del generador.
La implementación algorítmica se ha llevado a cabo mediante la creación de una biblioteca de Julia llamada **ProblemGenerator**, con una sintaxis *JuMP-like* intuitiva. Primero debe tenerse un problema de optimización Binivel SLSF planteado como el 1.5, el punto que se desea que sea estacionario (z_{est}), los índices activos descritos anteriormente según el caso así como sus multiplicadores correspondientes, que cumplirán las propiedades del tipo de estacionariedad requerida y el $\boldsymbol{\alpha}$.
Para una mayor facilidad de compresión del uso de la biblioteca se utiliza este pro-

blema ejemplo:

$$\begin{aligned}
\min_{x_1,x_2} \quad & x_1^2 y_1^2 y_2 + x_2 \\
\text{s.t.} \quad & x_1 + y_2 - y_1 \leq 9, \\
& \min_{y_1,y_2} \quad x_2^2 y_1^2 y_2 + x_1 \\
& \text{s.t.} \quad x_1^2 y_1^2 + x_2 \leq 0.
\end{aligned} \tag{2.13}$$

Se debe tener instalado en el entorno de desarrollo de Julia los siguientes módulos:

- Symbolics
- LinearAlgebra

Para la utilización de la biblioteca se procede a su importación de la siguiente forma:

```
using ProblemGenerator
```

Ejemplo de código 2.1: Importar el Módulo

Para comenzar debe llamarse a la función **GeneratorModel** para crear el modelo base inicial.

Para $\boldsymbol{\alpha} = \mathbf{0}$

Para crear un modelo donde $\boldsymbol{\alpha} = \mathbf{0}$:

```
# Crear el modelo base del generador alpha=0
model=GeneratorModel()

```

Ejemplo de código 2.2: Crear el modelo para $\boldsymbol{\alpha} = \mathbf{0}$

Para $\boldsymbol{\alpha} \neq \mathbf{0}$

Para crear un modelo donde $\boldsymbol{\alpha} \neq \mathbf{0}$, se tiene que pasar como parámetro el valor del α, el cual será un vector de *Number*:

```
# Crear el modelo base del generador alpha!=0
alpha_vec=Vector::{Number}
model=GeneratorModel(alpha_vec)
```

Ejemplo de código 2.3: Crear el modelo para $\boldsymbol{\alpha} \neq \mathbf{0}$

En caso que se requiera como entrada un **Problem**, se debe declarar de que nivel se trata de evaluar el **model** en las funciones **Upper** para el nivel superior y **Lower** para el inferior.

Para referirse al nivel superior

```
    # Referirse nivel superior
    Upper(model)
```

Ejemplo de código 2.4: Referirse al nivel superior

Para referirse al nivel inferior

```
    # Referirse nivel inferior
    Lower(model)
```

Ejemplo de código 2.5: Referirse al nivel inferior

Las variables deben de declararse bajo la siguiente macro **@myvariables**, la cual recibe una función **Upper** o **Lower** que recibe el modelo para las variables del nivel superior y las del nivel inferior respectivamente.

Introduccion de las variables del nivel superior

```
    # Se declara en el nivel superior las variables x_1, x_2
    @myvariables Upper(model) x_1, x_2
```

Ejemplo de código 2.6: Introducir las variables del nivel superior

Introducción de las variables del nivel inferior

```
   # Se declara en el nivel inferior las variables y_1, y_2
   @myvariables Lower(model) y_1,y_2
```

Ejemplo de código 2.7: Introducir las variables del nivel inferior

2.3.1. Declarar Funciones Objetivo

Para declarar las funciones objetivos debe asumirse que en ambos casos es un problema de minimización. Se utilizará la función **SetObjectiveFunction** que recibe un **Problem** y una expresión de **Num**.

Declaración de una función objetivo del nivel superior

Con:

$$\min(x_1^2 * y_1^2 * y_2) + x_2$$

```
    # Declarar la funcion objetivo del nivel superior
    # Min de ((x_1^2)*(y_1^2)*(y_2))+x_2
    SetObjectiveFunction(Upper(model),((x_1^2)*(y_1^2)*(y_2))+x_2)
```

Ejemplo de código 2.8: Declarar una función objetivo del nivel superior

Declaración de una función objetivo del nivel inferior

Con:

$$\min(x_2^2 * y_1^2 * y_2) + x_1$$

```
    # Declarar la funcion objetivo del nivel inferior
    # Min de ((x_2^2)*(y_1^2)*(y_2))+x_1
    SetObjectiveFunction(Lower(model),((x_2^2)*(y_1^2)*(y_2))+x_1)
```

Ejemplo de código 2.9: Declarar una función objetivo del nivel inferior.

2.3.2. Definición del conjunto de Índices activos

Antes de ilustrar como introducir las restricciones se va a explicar que los tipos de índices activos.

- **Ambos Niveles**:
  - **Normal** si no es un índice activo.
- **Nivel Superior**:
  - **J__0__g** Si es un índice como en 2.1.
- **Nivel Inferior**:
  - **J__0__LP__v** Si es un índice como en 2.3.
  - **J__0__L0__v** Si es un índice como en **??**.
  - **J__Ne__L0__v** Si es un índice como en 2.4.

Los **RestrictionSetType** deben expresar en código de esta forma:

```
    Normal
    J_0_g
    J_0_LP_v
    J_0_L0_v
    J_Ne_L0_v
```

Ejemplo de código 2.10: Definir el conjunto de índice activo

Los **RestrictionSetType** son un *enum* de Julia.
Para declarar las restricciones se brindan dos funciones:

- **SetLeaderRestriction:** Para el nivel superior.
- **SetFollowerRestriction:** Para el nivel inferior.
- **Nivel Superior:**

  Para declarar las restricciones del nivel superior debe por cada restricción llamarse a la función **SetLeaderRestriction** con:

  - El modelo (**model**)
  - La expresión de la restricción, del tipo **Num**
  - El tipo de restricción, del tipo **RestrictionSetType**
  - El valor de μ_i correspondiente, del tipo **Number**

  Ejemplo para introducir la restricción:

  $$x_1 + y_2 - y_1 \leq 9$$
  Índice activo del tipo J_0_g 2.1
  $$\mu_i = 0{,}3$$

```
    # Ejemplo de restriccion del nivel superior
    SetLeaderRestriction(model,x_1+y_2-y_1<=9,J_0_g,0.3)
```

  Ejemplo de código 2.11: Introducir restricción del nivel superior

- **Nivel Inferior:**
  Para declarar las restricciones del nivel inferior debe por cada restricción llamarse a la función **SetFollowerRestriction** con:

  - El modelo (**model**)
  - La expresión de la restricción, del tipo **Num**
  - El tipo de restricción, del tipo **RestrictionSetType**
  - El valor de β_j correspondiente, del tipo **Number**
  - El valor de λ_j correspondiente, del tipo **Number**
  - El valor de γ_j correspondiente en caso de ser valor de entrada, del tipo **Number**

Ejemplo para introducir la restricción:

- $\lambda_j \neq 0$**:**

$$((x_1^2) * (y_1^2)) + x_2 \leq 0$$
Índice activo del tipo J_Ne_L0_v2,4
$$\beta_j = 0{,}1$$
$$\lambda_j = 0{,}5$$

```
        # Para caso lamda_j=0
        SetFollowerRestriction(model,((x_1^2)*(y_1^2))+x_2
    <=0,J_Ne_L0_v,0.1,0.5,0)

```

- $\lambda_j = 0$:

Análogo al caso anterior pero con $\lambda_j = 0$, por lo que γ_j valor de entrada

$$\beta_j = 0{,}1, \quad \lambda_j = 0, \quad \gamma_j = 0{,}4$$

```
        # Para caso lamda_j!=0
        SetFollowerRestriction(model,((x_1^2)*(y_1^2))+x_2
    <=0,J_Ne_L0_v,0.1,0,0.4)

```

2.3.3. Introducir el Punto

Ahora debe de introducirse el valor del punto que debe ser estacionario de la clase seleccionada. Debe de tenerse en cuenta que para todas las variables declaradas en ambos niveles debe de definirse el valor de la componente.

```
    # Introducir el punto (1,1,1,1)
    SetPoint(model,Dict(x_1=>1,x_2=>1,y_1=>1,y_2=>1))
```

Ejemplo de código 2.12: Introducir el punto $(1, 1, 1, 1)$

2.3.4. Generar el Problema

Finalmente al tener todos los datos previos introducidos se llama al **CreateProblem** pasándole como parámetro el *model* y generará dicho problema imprimiendo en consola este.

```
    # Llamar para generar el problema
    problem=CreateProblem(model)
```

Ejemplo de código 2.13: Generar el problema

2.3.5. Ejemplo completo

Se muestra el ejemplo completo para el problema antes propuesto:

```
    # Importar dependencias necesarias
    using ProblemGenerator
    # Crear el modelo base del generador alpha=0
    model=GeneratorModel()
    #Declaracion de variables
    # Se declara en el nivel superior las variables x_1, x_2
    @myvariables Upper(model) x_1, x_2
    # Se declara en el nivel inferior las variables y_1, y_2
    @myvariables Lower(model) y_1,y_2
    # Declarar la funcion objetivo del nivel superior
    # Min de ((x_1^2)*(y_1^2)*(y_2))+x_2
    SetObjectiveFunction(Upper(model),((x_1^2)*(y_1^2)*(y_2))+x_2)
    # Ejemplo de restriccion del nivel superior
    SetLeaderRestriction(model,x_1+y_2-y_1>9,J_0_g,0.3)
    # Declarar la funcion objetivo del nivel inferior
    # Min de ((x_2^2)*(y_1^2)*(y_2))+x_1
    SetObjectiveFunction(Lower(model),((x_2^2)*(y_1^2)*(y_2))+x_1)
    # Para caso lamda_j!=0
    SetFollowerRestriction(model,((x_1^2)*(y_1^2))+x_2<=0,J_Ne_L0_v
   ,0.1,0,0.4)
    # Introducir el punto (1,1,1,1)
    SetPoint(model,Dict(x_1=>1,x_2=>1,y_1=>1,y_2=>1))
    # Llamar para generar el problema
    problem=CreateProblem(model)
```

Ejemplo de código 2.14: Script

2.3.6. Documentación Oficial

Para mayor información visitar la documentación oficial del generador:

https://fvsb.github.io/Tesis/.

Capítulo 3

Experimentación

En este capítulo se experimentará tomando una serie de problemas, ver Zhou et al. 2018, que sean: Lineales, Cuadráticos y No Convexos. En ella primeramente utilizaremos las bibliotecas de Julia para obtener puntos que sean mínimos locales, después añadir valores aleatorios a esos puntos y posteriormente generar problemas estacionarios del tipo: fuerte, M y C . Posteriormente estos problemas modificados serán nuevamente ejecutados por los algoritmos tradicionales de Julia para conocer su efectividad con respecto al valor de la función objetivo del nivel superior.
Para ello tomaremos 15 problemas de optimización binivel. Entre ellos habrá 3 clases: Lineales, Cuadráticos y No Convexos, donde habrá 5 problemas de cada clase. Estos han sido extraídos de Floudas 1999 para las dos primeras clasificaciones y Zhou et al. 2018 para la última.

Los problemas escogidos para la experimentación fueron:

Tabla 3.1: Problemas Seleccionados

No Convexos	**Lineales**	**Cuadráticos**
MitsosBarton2006Ex312	ex9.1.1	ex9.2.1
MitsosBarton2006Ex313	ex9.1.2	ex9.2.2
MitsosBarton2006Ex314	ex9.1.8	ex9.2.3
MitsosBarton2006Ex323	ex9.1.9	ex9.2.4
MorganPatrone2006a	ex9.1.10	ex9.2.5

3.1. Modelación de la experimentación

Se describirá el proceso de generar la experimentación. Todos los valores, en esta parte, han sido redondeados por exceso a dos cifras después de la coma.
Inicialmente se necesita obtener óptimos de los problemas con los paquetes convencionales de Julia cuya forma de resolver dependerá de la clase del problema. A continuación definiremos este tipo de clases y como se obtienen los óptimos.

- **Problemas Lineales y Cuadráticos :**
  Un problema binivel lineal es aquel el cual los problemas del nivel superior e inferior son lineales, análogamente se define el problema cuadrático.

  Con dicho problema se introduce los datos en la interfaz de **BilevelJuMP**, ver Garcia et al. 2022, con el cual se utilizan 3 técnicas entre las ofrecidas por esta:

  - **Big-M :** Con el optimizador High-Performance Solver for Linear Programming (HiGHS) y los valores primal big $\mathrm{M} = 100, \quad$ dual big $\mathrm{M} = 100$.
  - **SOS1 :** Con el optimizador Solving Constraint Integer Programs (SCIP).
  - **ProductMode :** Con el optimizador Interior Point Optimizer (Ipopt).

  Cada uno de los resultados de evaluar el problema en cada forma anterior se guarda en un formato *.xlsx* donde por cada optimizador se guarda los parámetros:

  - Estatus del Primal, el cual define si es un punto factible o no.
  - Estatus de la Finalización, si terminó porque encontró un óptimo o se estancó en un óptimo local.
  - Valor de la función objetivo del nivel superior.
  - El punto óptimo encontrado, en caso de ser hallado.

  Posteriormente se analizan los resultados de dichos métodos, se selecciona el de mejor evaluación de la función objetivo.

- **Problemas No Convexos :**
  Al **BilevelJuMP** no contar con soporte para esta clase de problemas se utiliza **JuMP**, ver Lubin et al. 2023. Por ello se utiliza la reformulación KKT como la de 1.5 y se procede a utilizar la interfaz brindada por este. Para el caso de las restricciones de complementariedad se utiliza **Complementarity**, ver Chkwon 2025, con el optimizador Ipopt y análogo al caso anterior se extraen los mismos datos.

Generación de los problemas

Se toma el problema original de entrada y el punto obtenido en el paso anterior el cual en cada componente se le hace suma un valor aleatorio entre $1e-10$ y 5, siendo este modificado: z_0^*. Se generan los 3 problemas bajo los 3 tipos de estacionariedad descritos en cada uno, además en cada uno se toma la opción de $\boldsymbol{\alpha} = \mathbf{0}$ y $\boldsymbol{\alpha} \neq \mathbf{0}$. Para los $\boldsymbol{\alpha} \neq \mathbf{0}$ se genera un vector aleatorio donde cada componente está entre $1e-10$ y 3. Luego para los conjuntos de índices activos de las $v_j s$ se dividen en 1/2 del tipo J_1^v (**??**) y 1/4 para los dos restantes. Con respecto a la selección de los multiplicadores β_j y γ_j se se generan valores aleatorios entre $1e-10$ y 10 en caso que estos no tengan que ser 0, para los casos en que haya más de una combinación de los multiplicadores con respecto a su igualdad a 0 se toma un valor aleatorio generado por una distribución uniforme discreta. Finalmente cada problema generado es guardado en un archivo *xlsx* con la siguiente designación: *(nombre del problema)_(Tipo de punto estacionario)(generator)_alpha_((non_zero) si $\alpha \neq 0$ y (zero) si $\alpha = 0$).xlsx*, donde se guarda:

- Las expresiones de las funciones objetivo de ambos niveles y su valor evaluado en el punto.
- Las restricciones de ambos niveles con sus multiplicadores respectivos, el tipo de indice activo y la evaluación de dicha función en el punto.
- El punto z_0^*.
- El vector $\boldsymbol{bf}$.
- El vector $\boldsymbol{BF}$.
- El multiplicador $\boldsymbol{\alpha}$.

Comparación de los algoritmos de Julia

Después de tener generados los problemas se utilizan los mismos métodos de Julia mencionados anteriormente para obtener óptimos de cada problema generado y se elige como representante el que más haya superado su óptimo o en caso de no superar el que mayor distancia tenga con el valor objetivo inicial.

3.1.1. Resultados:

Se presentan los resultados seleccionados bajo los criterios expuestos anteriormente en las siguientes tablas que contienen:

- El nombre del problema original desde el cual fue modificado para que fuese estacionario de la clase deseada.
- El punto al cual se forzó ser estacionario de la clase requerida y no de un subconjunto de esta, por ejemplo si es C-estacionario no es M-estacionario.
- La evaluación de la función objetivo del punto estacionario.
- El punto óptimo hallado por los algoritmos de Julia.
- La evaluación de la función objetivo del óptimo.
- Método seleccionado.

Problemas Lineales

Tabla 3.2: Problemas Lineal Seleccionados

Tipo de punto	**Nombre del problema**	**Punto estacionario**	**Valor objetivo del punto estacionario**	**Punto óptimo**	**Valor del punto óptimo**	**Método seleccionado**
$\alpha = 0$	ex9.1.1	(6.05, 6.85, 3.1)	18.59	(6.13, 6.85, -173.65)	-1820.0	Highs-BigM (100,100)
C-Estacionario	ex9.1.1	(6.05, 6.85, 3.1)	74.95	(6.25, 10.36, 6.6)	-27.94	Product Mode
M-Estacionario	ex9.1.1	(6.05, 6.85, 3.1)	-350.25	(6.1, 2.56, 0.19)	-353.19	Product Mode
Fuertemente-Estacionario	ex9.1.1	(6.05, 6.85, 3.1)	-195.55	(6.14, 6.84, 3.11)	-195.39	Highs-BigM (100,100)

Problemas Cuadráticos

Tabla 3.3: Problemas Cuadráticos Seleccionados

Tipo de punto	**Nombre del problema**	**Punto estacionario**	**Valor objetivo del punto estacionario**	**Punto óptimo**	**Valor del punto óptimo**	**Método seleccionado**
$\alpha = 0$	ex9.2.1	(1.85, 4.65)	-106.29	(2.19, 5.31)	-108.96	Product Mode
C-Estacionario	ex9.2.3	(1.55, 2.7, -5.1, -8.65)	363.83	(32.97, -5.86, 15.95, 13.31)	-1201.53	SOS1
M-Estacionario	ex9.2.3	(1.55, 2.7, -5.1, -8.65)	405.85	(1.48, 2.65, -5.08, -8.66)	404.57	Highs-BigM (100,100)
Fuertemente-Estacionario	ex9.2.3	(1.55, 2.7, -5.1, -8.65)	293.57	(-6.9, -7.39, 0.87, -12.87)	111.17	Highs-BigM (100,100)

Problemas No Convexos

Tabla 3.4: Problemas No Convexos Seleccionados

Tipo de punto	**Nombre del problema**	**Punto estacionario**	**Valor objetivo del punto estacionario**	**Punto óptimo**	**Valor del punto óptimo**	**Método seleccionado**
$\alpha = 0$	MitsosBarton2006Ex323	(1.94, 3.7)	60.69	(3.7, 3.7)	17.53	Reformulacion KKT
C-Estacionario	MitsosBarton2006Ex323	(1.94, 3.7)	179.12	(3.7, 3.7)	179.08	Reformulacion KKT
M-Estacionario	MorganPatrone2006a	(2.85, 4.45)	-1.45	(-89783940.52, -89783940.52)	-1.45	Reformulacion KKT
Fuertemente-Estacionario	MitsosBarton2006Ex312	(1.9, 0.15,)	2.72	(0.0, 0.0)	2.26	Reformulacion KKT

Análisis de los resultados

En el caso de los problemas lineales los 3 algoritmos generalmente dieron con puntos con similar valor de la función objetivo aunque se tomó como referencia Big-M dado que fue el más eficiente en tiempo y si redondeamos a 2 números después de la coma son prácticamente todos iguales. En los problemas cuadráticos se encontró con mayor diferencia entre el valor de la función en el punto estacionario que en el óptimo local alcanzado, en algunos de estos problemas Big-M no pudo hallar un óptimo con calidad posiblemente debido a sus parámetros y continua la tendencia del problema que parte desde un punto fuertemente estacionario de tener el mayor rango de mejora. En los no convexos continua esta hipótesis dado que los puntos M-estacionarios y fuertemente estacionarios son los de mayor requerimientos, fueron los que lograron encontrar mejores óptimos.

Conclusiones

En esta tesis se desarrolló un algoritmo para la generación de problemas de optimización binivel con características específicas de estacionariedad en puntos determinados. El mismo tiene la capacidad de modificar problemas originales para garantizar la factibilidad y estacionariedad en puntos dados, aprovechando las capacidades del lenguaje de programación Julia, que destaca por su alto rendimiento computacional y sintaxis adaptada a problemas de optimización.

La experimentación se llevó a cabo sobre tres categorías fundamentales de problemas: lineales, cuadráticos y no convexos. El proceso experimental comenzó con la obtención de puntos mínimos utilizando bibliotecas establecidas como BilevelJuMP y JuMP. Posteriormente, se generaron problemas estacionarios mediante la adición de componentes aleatorias a estos puntos para cada clase de problema definida.

Los problemas modificados fueron sometidos nuevamente a algoritmos tradicionales implementados en Julia para evaluar su efectividad frente a los puntos estacionarios generados. Se realizó un análisis comparativo destacando los casos más relevantes de cada categoría de puntos estacionarios, considerando la evaluación de la función objetivo del nivel superior en el punto donde se garantizó la estacionariedad, contrastándola con los resultados obtenidos por las bibliotecas convencionales.

Recomendaciones

Como resultado de la investigación realizada, se han identificado varias líneas de trabajo futuro que permitirían expandir y mejorar los resultados obtenidos. En primer lugar, se recomienda ampliar el alcance de la experimentación numérica para incluir una mayor diversidad de problemas de optimización binivel. Esta expansión permitiría validar la robustez y versatilidad del algoritmo propuesto en diferentes contextos y escenarios de aplicación.

En segunda instancia, se sugiere profundizar en la investigación sobre la implementación del algoritmo desarrollado como criterio de parada en nuevos métodos de optimización binivel. Esta línea de investigación podría contribuir significativamente al desarrollo de algoritmos más eficientes y confiables, mejorando la capacidad de detectar y verificar puntos estacionarios durante el proceso de optimización.

Finalmente, se propone el desarrollo de una interfaz gráfica más intuitiva y funcional que facilite la generación automática de puntos según el tipo de estacionariedad requerida. Esta mejora en la usabilidad del software permitiría que usuarios con diferentes niveles de experiencia puedan aprovechar las capacidades del algoritmo de manera más efectiva, ampliando así su aplicabilidad práctica en diversos campos de estudio.

Bibliografía

Aussel, D., Bendotti, P., & Pistek, M. (2017). Nash equilibrium in a pay-as-bid electricity market Part 2 - best response of a producer. *Optimization*, *66*, 1027-1053. https://api.semanticscholar.org/CorpusID:18648572 (vid. pág. 2).

Aussel, D., Cervinka, M., & Marechal, M. (2016). Deregulated electricity markets with thermal losses and production bounds: models and optimality conditions. *RAIRO Oper. Res.*, *50*, 19-38. https://api.semanticscholar.org/CorpusID:41625879 (vid. pág. 1).

Bard, J. F. (1991). Some properties of the bilevel programming problem. *Journal of Optimization Theory and Applications*, *68*(2), 371-378. https://doi.org/10.1007/BF00941574 (vid. pág. 3).

Bhavsar, N., & Verma, M. (2021). A subsidy policy to managing hazmat risk in railroad transportation network. *Eur. J. Oper. Res.*, *300*, 633-646. https://api.semanticscholar.org/CorpusID:238706367 (vid. pág. 2).

Bouza-Allende, G., Aussel, D., Dempe, S., & Lepaul, S. (2021). Genericity Analysis of Multi-Leader-Disjoint-Followers Game. *SIAM J. Optim.*, *31*, 2055-2079. https://api.semanticscholar.org/CorpusID:238717899 (vid. pág. 3).

Bouza-Allende, G., & Still, G. (2012). Solving bilevel programs with the KKT-approach. *Mathematical Programming*, *138*, 309-332. https://api.semanticscholar.org/CorpusID:18500519 (vid. pág. 5).

Caselli, G., Iori, M., & Ljubi'c, I. (2024). Bilevel optimization with sustainability perspective: a survey on applications. https://api.semanticscholar.org/CorpusID:270379993 (vid. pág. 3).

Cerulli, M. (2021, diciembre). *Bilevel optimization and applications* [Tesis doctoral]. (Vid. págs. 3, 4).

Chkwon. (2025). Complementarity.jl. https://github.com/chkwon/Complementarity.jl (vid. pág. 32).

Dempe, S., & Freiberg, T. (2003). Annotated Bibliography on Bilevel Programming and Mathematical Programs with Equilibrium Constraints. *Optimization*, *52*. https://doi.org/10.1080/0233193031000149894 (vid. pág. 3).

Dempe, S., & Zemkoho, A. (2020a). *Bilevel Optimization: Advances and Next Challenges*. (Vid. págs. 1, 3, 4, 7-9).

Dempe, S., & Zemkoho, A. (2020b). *Bilevel Optimization: Advances and Next Challenges*. (Vid. pág. 2).

Flegel, M. L., & Kanzow, C. (2003). A Fritz John Approach to First Order Optimality Conditions for Mathematical Programs with Equilibrium Constraints. *Optimization*, *52*, 277-286. https://api.semanticscholar.org/CorpusID:26882450 (vid. págs. 5, 9, 10).

Floudas, C. A. (1999). Handbook of Test Problems in Local and Global Optimization. https://api.semanticscholar.org/CorpusID:117898119 (vid. pág. 30).

Floudas, C. A., & Pardalos, P. M. (1990). A Collection of Test Problems for Constrained Global Optimization Algorithms. *Lecture Notes in Computer Science*. https://api.semanticscholar.org/CorpusID:139191 (vid. pág. 3).

Garcia, J. D., Bodin, G., & Street, A. (2022). BilevelJuMP.jl: Modeling and Solving Bilevel Optimization in Julia. *ArXiv*, *abs/2205.02307*. https://api.semanticscholar.org/CorpusID:248524800 (vid. págs. 12, 13, 18, 23, 32).

Gu, H., Li, Y., Yu, J., Wu, C., Song, T., & Xu, J. (2020). Bi-level optimal low-carbon economic dispatch for an industrial park with consideration of multi-energy price incentives. *Applied Energy*, *262*, 114276. https://api.semanticscholar.org/CorpusID:213998633 (vid. pág. 2).

Jeroslow, R. G. (1985). The polynomial hierarchy and a simple model for competitive analysis. *Mathematical Programming*, *32*, 146-164. https://api.semanticscholar.org/CorpusID:39987722 (vid. pág. 3).

JuliaLang. (2025). *Julia Documentation*. https://docs.julialang.org/en/v1/ (vid. pág. 23).

Lubin, M., Dowson, O., Dias Garcia, J., Huchette, J., Legat, B., & Vielma, J. P. (2023). JuMP 1.0: Recent improvements to a modeling language for mathematical optimization. *Mathematical Programming Computation*. https://doi.org/10.1007/s12532-023-00239-3 (vid. págs. 16, 23, 32).

Ramos, M. A., Boix, M., Aussel, D., Montastruc, L., & Domenech, S. (2016). Water integration in eco-industrial parks using a multi-leader-follower approach. *Comput. Chem. Eng.*, *87*, 190-207. https://api.semanticscholar.org/CorpusID:26463725 (vid. pág. 2).

Ramos, M. A., Rocafull, M., Boix, M., Aussel, D., Montastruc, L., & Domenech, S. (2018). Utility network optimization in eco-industrial parks by a multi-leader follower game methodology. *Comput. Chem. Eng.*, *112*, 132-153. https://api.semanticscholar.org/CorpusID:4003323 (vid. pág. 2).

Schmidt, M., & Beck, Y. (2021). A Gentle and Incomplete Introduction to Bilevel Optimization. https://www.lamsade.dauphine.fr/poc/sites/default/files/bilevel-optimization.pdf (vid. pág. 8).

Siddiqui, S., & Gabriel, S. (2012). An SOS1-Based Approach for Solving MPECs with a Natural Gas Market Application. *Networks and Spatial Economics*, *13*. https://doi.org/10.1007/s11067-012-9178-y (vid. págs. 4, 13).

Sinha, A., Malo, P., & Deb, K. (2017). A Review on Bilevel Optimization: From Classical to Evolutionary Approaches and Applications. *IEEE Transactions on Evolutionary Computation*, *22*, 276-295. https://api.semanticscholar.org/CorpusID:4626744 (vid. págs. 3, 4, 7, 8).

Zhou, S., Zemkoho, A. B., & Tin, A. (2018). BOLIB: Bilevel Optimization LIBrary of Test Problems. *Bilevel Optimization*. https://api.semanticscholar.org/CorpusID:119636540 (vid. pág. 30).

Anexos

Problema Binivel Alpha-Zero: ex9.1.1

mín $5{,}5y_1$

sujeto a:
mín $-y_1$

sujeto a:

$$
\begin{aligned}
5{,}0 - y_1 &\leq 0 \\
y_1 - 5{,}0 &\leq 0 \\
-2x + y_1 + 4y_2 + 3{,}0 &\leq 0 \\
8x + y_1 - 70{,}46 &\leq 0 \\
-2x - 2y_1 - 2{,}2 &\leq 0
\end{aligned}
$$

Problema Binivel C-Estacionario: ex9.1.1

mín 0

sujeto a:
mín $-y_1$

sujeto a:

$$\begin{aligned}
-0{,}49y_1+0{,}5y_2+1{,}42 &\leq 0\\
0{,}49y_1-0{,}5y_2-1{,}42 &\leq 0\\
-2x-1{,}51y_1+1{,}55y_2+20{,}58 &\leq 0\\
8x+0{,}49y_1-0{,}5y_2-67{,}02 &\leq 0\\
-2x-0{,}98y_1+1{,}0y_2+4{,}97 &\leq 0
\end{aligned}$$

Problema Binivel M-Estacionario: ex9.1.1

mín $12{,}6x+0{,}98y_1-0{,}68y_2$

sujeto a:
mín $-y_1$

sujeto a:

$$\begin{aligned}
-0{,}46y_1+0{,}79y_2+0{,}67 &\leq 0\\
0{,}68y_1-0{,}47y_2-2{,}42 &\leq 0\\
-2x-1{,}2y_1+0{,}83y_2+20{,}48 &\leq 0\\
8x+0{,}86y_1-0{,}21y_2-69{,}32 &\leq 0\\
-2x-1{,}07y_1+1{,}34y_2+5{,}19 &\leq 0
\end{aligned}$$

Problema Binivel Fuertemente-Estacionario: ex9.1.1

mín $\quad -55{,}4x - 39{,}65y_1 - 23{,}19y_2$

sujeto a:
mín $\quad -y_1$

sujeto a:

$$
\begin{aligned}
1{,}48y_1 + 2{,}02y_2 - 11{,}54 &\leq 0 \\
2{,}03y_1 + 0{,}84y_2 - 11{,}89 &\leq 0 \\
-2x - 1{,}56y_1 + 1{,}91y_2 + 20{,}08 &\leq 0 \\
8x + 2{,}11y_1 + 0{,}91y_2 - 77{,}6 &\leq 0 \\
-2x + 0{,}91y_1 + 2{,}38y_2 + 6{,}27 &\leq 0
\end{aligned}
$$

Problema Binivel Alpha-Zero: ex9.2.2

mín $\quad x^2 - 28{,}5x - 16{,}1y + (y-10)^2$

sujeto a: $-x + y + 0{,}05 \leq 0$ **mín** $\quad (x + 2y - 30)^2$

sujeto a:

$$
\begin{aligned}
x + y - 28{,}85 &\leq 0 \\
14{,}4 - y &\leq 0 \\
y - 25{,}6 &\leq 0
\end{aligned}
$$

Problema Binivel C-Estacionario: ex9.2.2

mín $x^2 - 30{,}24x - 77{,}66y + (y-10)^2$

sujeto a: $-x + y + 0{,}05 \leq 0$ **mín** $(x + 2y - 30)^2$

sujeto a:

$$\begin{aligned} x - 3{,}37y + 34{,}04 &\leq 0 \\ 1{,}91y - 27{,}5 &\leq 0 \\ 1{,}11e - 16y - 40{,}0 &\leq 0 \end{aligned}$$

Problema Binivel M-Estacionario: ex9.2.2

mín $x^2 - 34{,}62x - 24{,}12y + (y-10)^2$

sujeto a: -$x + y + 0{,}05 \leq 0$ **mín** $(x + 2y - 30)^2$

sujeto a:

$$\begin{aligned} x + 0{,}58y - 22{,}8 &\leq 0 \\ 0{,}38y - 5{,}47 &\leq 0 \\ 1{,}11e - 16y - 40{,}0 &\leq 0 \end{aligned}$$

Problema Binivel Fuertemente-Estacionario: ex9.2.2

mín $x^2 - 38{,}94x - 183{,}46y + (y-10)^2$

sujeto a: $-x + y + 0{,}05 \leq 0$ **mín** $(x + 2y - 30)^2$

sujeto a:

$$\begin{aligned} x + 15{,}0y - 230{,}45 &\leq 0 \\ 4{,}9 - 0{,}34y &\leq 0 \\ 7{,}12y - 103{,}32 &\leq 0 \end{aligned}$$

Problema Binivel Alpha-Zero: ex9.1.8

mín 0

sujeto a:
mín $x_1 + x_2 - 4y_1 + y_2$

sujeto a:

$$\begin{aligned} -2x_1 + y_1 - y_2 + 4{,}45 &\leq 0 \\ x_1 - 3x_2 + y_2 - 1{,}25 &\leq 0 \\ x_1 + x_2 - 6{,}43 &\leq 0 \end{aligned}$$

Problema Binivel C-Estacionario: ex9.1.8

mín 0

sujeto a:
mín $x_1 + x_2 - 4y_1 + y_2$

sujeto a:

$$\begin{aligned} -2x_1 + 0{,}1y_1 - 1{,}08y_2 + 10{,}08 &\leq 0 \\ x_1 - 3x_2 - 0{,}09y_1 + 0{,}99y_2 - 0{,}69 &\leq 0 \\ x_1 + x_2 - 6{,}31 &\leq 0 \end{aligned}$$

Problema Binivel M-Estacionario: ex9.1.8

mín $\quad -2{,}9x_1 - 2{,}9x_2$

sujeto a:
mín $\quad x_1 + x_2 - 4y_1 + y_2$

sujeto a:

$$
\begin{aligned}
-2x_1 + 1{,}57y_1 + 1{,}93y_2 - 3{,}58 &\leq 0 \\
x_1 - 3x_2 - 0{,}19y_1 + 0{,}04y_2 + 1{,}39 &\leq 0 \\
x_1 + x_2 - 6{,}6 &\leq 0
\end{aligned}
$$

Problema Binivel Fuertemente-Estacionario: ex9.1.8

mín $\quad -3{,}8x_1 - 8{,}8x_2 - 24{,}37y_1 - 27{,}42y_2$

sujeto a:
mín $\quad x_1 + x_2 - 4y_1 + y_2$

sujeto a:

$$
\begin{aligned}
-2x_1 + 2{,}36y_1 + 0{,}96y_2 - 6{,}9 &\leq 0 \\
x_1 - 3x_2 - 0{,}47y_1 + 0{,}33y_2 + 2{,}65 &\leq 0 \\
x_1 + x_2 + 1{,}68y_1 + 2{,}41y_2 - 20{,}71 &\leq 0
\end{aligned}
$$

Problema Binivel Alpha-Zero: ex9.1.9

mín $-0{,}2x+0{,}8y$

sujeto a:
mín $-5x-y$

sujeto a:

$$\begin{aligned} -x-0{,}5y+7{,}87 &\leq 0 \\ -0{,}25x+y-4{,}97 &\leq 0 \\ x+0{,}5y-7{,}87 &\leq 0 \\ x-2y-11{,}56 &\leq 0 \\ -y-6{,}17 &\leq 0 \end{aligned}$$

Problema Binivel C-Estacionario: ex9.1.9

mín $-9{,}8x-50{,}74y$

sujeto a:
mín $-5x-y$

sujeto a:

$$\begin{aligned} -x-4{,}09y+30{,}03 &\leq 0 \\ 1{,}2-0{,}25x &\leq 0 \\ x-4{,}78 &\leq 0 \\ x-5{,}6 &\leq 0 \\ -2{,}66y-16{,}4 &\leq 0 \end{aligned}$$

Problema Binivel M-Estacionario: ex9.1.9

mín $\quad 0$

sujeto a:
mín $\quad -5x-y$

sujeto a:

$$
\begin{aligned}
-x+1{,}21y-2{,}67 &\leq 0\\
-0{,}25x+3{,}28y-19{,}02 &\leq 0\\
x-4{,}78 &\leq 0\\
x+1{,}9y-16{,}69 &\leq 0\\
-0{,}17 &\leq 0
\end{aligned}
$$

Problema Binivel Fuertemente-Estacionario: ex9.1.9

mín $\quad -10{,}07x-80{,}17y$

sujeto a:
mín $\quad -5x-y$

sujeto a:

$$
\begin{aligned}
-x+3{,}28y-15{,}45 &\leq 0\\
-0{,}25x+3{,}6y-21{,}03 &\leq 0\\
x-4{,}78 &\leq 0\\
x+3{,}76y-28{,}19 &\leq 0\\
2{,}74y-17{,}56 &\leq 0
\end{aligned}
$$

Problema Binivel Alpha-Zero: ex9.1.10

mín $-2{,}4y_2$

sujeto a:
mín $x_1+x_2-4y_1+y_2$

sujeto a:

$$\begin{aligned}
-2x_1+y_1-y_2+1{,}1 &\leq 0\\
x_1-3x_2+y_2+2{,}55 &\leq 0\\
102{,}2-y_1 &\leq 0\\
-y_2-0{,}2 &\leq 0
\end{aligned}$$

Problema Binivel C-Estacionario: ex9.1.10

mín 0

sujeto a:
mín $x_1+x_2-4y_1+y_2$

sujeto a:

$$\begin{aligned}
-2x_1+1{,}13y_1-0{,}18y_2-12{,}22 &\leq 0\\
x_1-3x_2-0{,}15y_1+0{,}02y_2+18{,}33 &\leq 0\\
-0{,}98y_1+0{,}15y_2+99{,}73 &\leq 0\\
0{,}15y_1-0{,}02y_2-15{,}75 &\leq 0
\end{aligned}$$

Problema Binivel M-Estacionario: ex9.1.10

mín $\quad -8{,}0x_1 + 24{,}0x_2 + 2{,}31y_1 - 7{,}26y_2$

sujeto a:
mín $\quad x_1 + x_2 - 4y_1 + y_2$

sujeto a:

$$\begin{aligned}
-2x_1 + 3{,}61y_1 - 0{,}17y_2 - 266{,}23 &\leq 0\\
x_1 - 3x_2 - 0{,}29y_1 + 0{,}91y_2 + 32{,}1 &\leq 0\\
-0{,}09y_1 + 0{,}29y_2 + 9{,}34 &\leq 0\\
0{,}5y_1 - 0{,}84y_2 - 50{,}92 &\leq 0
\end{aligned}$$

Problema Binivel Fuertemente-Estacionario: ex9.1.10

mín $\quad -1{,}2x_1 + 30{,}0x_2 - 179{,}95y_1 - 56{,}82y_2$

sujeto a:
mín $\quad x_1 + x_2 - 4y_1 + y_2$

sujeto a:

$$\begin{aligned}
-2x_1 + 6{,}23y_1 + 0{,}74y_2 - 534{,}1 &\leq 0\\
x_1 - 3x_2 + 8{,}37y_1 + 3{,}79y_2 - 853{,}08 &\leq 0\\
-0{,}1y_1 + 0{,}3y_2 + 10{,}16 &\leq 0\\
8{,}63y_1 + 1{,}88y_2 - 883{,}47 &\leq 0
\end{aligned}$$

Problema Binivel Alpha-Zero: ex9.2.1

mín $\quad 27{,}0x - 58{,}55y + (x-5)^2 + (2y+1)^2$

sujeto a:
mín $\quad -1{,}5xy + (y-1)^2$

sujeto a:

$$
\begin{aligned}
-3x + y + 0{,}9 &\leq 0 \\
x - 0{,}5y + 0{,}47 &\leq 0 \\
x + y - 7{,}5 &\leq 0
\end{aligned}
$$

Problema Binivel C-Estacionario: ex9.2.1

mín $\quad 46{,}92x - 47{,}48y + (x-5)^2 + (2y+1)^2$

sujeto a:
mín $\quad -1{,}5xy + (y-1)^2$

sujeto a:

$$
\begin{aligned}
5{,}55 - 3x &\leq 0 \\
x + 0{,}51y - 4{,}2 &\leq 0 \\
x - 0{,}95y - 16{,}55 &\leq 0
\end{aligned}
$$

Problema Binivel M-Estacionario: ex9.2.1

mín $\quad 14{,}13x - 39{,}51y + (x-5)^2 + (2y+1)^2$

sujeto a:

mín $\quad -1{,}5xy + (y-1)^2$

sujeto a:

$$\begin{aligned} 1{,}11e - 3x - 16y + 5{,}55 &\leq 0 \\ x - 0{,}46y + 0{,}27 &\leq 0 \\ 1{,}11e + x - 16y - 12{,}15 &\leq 0 \end{aligned}$$

Problema Binivel Fuertemente-Estacionario: ex9.2.1

mín $\quad 19{,}95x - 114{,}54y + (x-5)^2 + (2y+1)^2$

sujeto a:

mín $\quad -1{,}5xy + (y-1)^2$

sujeto a:

$$\begin{aligned} -3x + 6{,}89y - 26{,}48 &\leq 0 \\ x - 0{,}05y - 1{,}62 &\leq 0 \\ x + 0{,}67y - 9{,}05 &\leq 0 \end{aligned}$$

Problema Binivel Alpha-Zero: ex9.2.2

mín $\quad x^2 - 28{,}5x - 16{,}1y + (y-10)^2$

sujeto a: $-x + y + 0{,}05 \leq 0 \quad$ **mín** $\quad (x + 2y - 30)^2$

sujeto a:

$$\begin{aligned} x + y - 28{,}85 &\leq 0 \\ 14{,}4 - y &\leq 0 \\ y - 25{,}6 &\leq 0 \end{aligned}$$

Problema Binivel C-Estacionario: ex9.2.2

mín $\quad x^2 - 30{,}24x - 77{,}66y + (y-10)^2$

sujeto a: $-x + y + 0{,}05 \leq 0 \quad$ **mín** $\quad (x + 2y - 30)^2$

sujeto a:

$$\begin{aligned} x - 3{,}37y + 34{,}04 &\leq 0 \\ 1{,}91y - 27{,}5 &\leq 0 \\ 1{,}11e - 16y - 40{,}0 &\leq 0 \end{aligned}$$

Problema Binivel M-Estacionario: ex9.2.2

mín $\quad x^2 - 34{,}62x - 24{,}12y + (y-10)^2$

sujeto a: $-x + y + 0{,}05 \leq 0 \quad$ **mín** $\quad (x + 2y - 30)^2$

sujeto a:

$$\begin{aligned} x + 0{,}58y - 22{,}8 &\leq 0 \\ 0{,}38y - 5{,}47 &\leq 0 \\ 1{,}11e - 16y - 40{,}0 &\leq 0 \end{aligned}$$

Problema Binivel Fuertemente-Estacionario: ex9.2.2

mín $\quad x^2 - 38{,}94x - 183{,}46y + (y-10)^2$

sujeto a: $-x + y + 0{,}05 \leq 0 \quad$ **mín** $\quad (x + 2y - 30)^2$

sujeto a:

$$\begin{aligned} x + 15{,}0y - 230{,}45 &\leq 0 \\ 4{,}9 - 0{,}34y &\leq 0 \\ 7{,}12y - 103{,}32 &\leq 0 \end{aligned}$$

Problema Binivel Alpha-Zero: ex9.2.3

mín $\quad -0{,}86x_1 - 0{,}86x_2 - 3{,}76y_1 - 6{,}18y_2 - 60$

sujeto a: $x_1 + x_2 + y_1 - 2y_2 - 16{,}45 \leq 0 \quad$ **mín** $\quad (-x_1 + y_1 + 20)^2 + (-x_2 + y_2 + 20)^2$

sujeto a:

$$\begin{aligned} -x_1 + 2y_2 + 18{,}85 &\leq 0 \\ -x_2 + 2y_2 + 20{,}0 &\leq 0 \\ -y_1 - 5{,}1 &\leq 0 \\ y_1 + 5{,}1 &\leq 0 \\ -y_2 - 11{,}35 &\leq 0 \\ y_2 - 48{,}65 &\leq 0 \end{aligned}$$

Problema Binivel C-Estacionario: ex9.2.3

mín $\quad -8{,}96x_1+6{,}88x_2-19{,}87y_1-36{,}74y_2-60$

sujeto a: $x_1+x_2+y_1-2y_2-16{,}45\leq 0 \quad$ **mín** $\quad (-x_1+y_1+20)^2+(-x_2+y_2+20)^2$

sujeto a:

$$\begin{aligned}
-x_1-1{,}38y_1-1{,}54y_2-18{,}85&\leq 0\\
-x_2-0{,}68y_1+0{,}26y_2+1{,}53&\leq 0\\
-0{,}87y_1+0{,}34y_2-1{,}49&\leq 0\\
0{,}73y_1-0{,}7y_2-2{,}33&\leq 0\\
1{,}08y_1+1{,}77y_2-40{,}78&\leq 0\\
0{,}44y_1+2{,}12y_2-60{,}58&\leq 0
\end{aligned}$$

Problema Binivel M-Estacionario: ex9.2.3

mín $\quad 10{,}72x_1+14{,}18x_2-17{,}4y_1-37{,}25y_2-60$

sujeto a: $x_1+x_2+y_1-2y_2-16{,}45\leq 0 \quad$ **mín** $\quad (-x_1+y_1+20)^2+(-x_2+y_2+20)^2$

sujeto a:

$$\begin{aligned}
-x_1+0{,}09y_1+2{,}1y_2+20{,}13&\leq 0\\
-x_2-0{,}99y_1+0{,}87y_2+5{,}16&\leq 0\\
0{,}98y_1+2{,}25y_2+24{,}47&\leq 0\\
-0{,}54y_1-1{,}76y_2-17{,}99&\leq 0\\
2{,}02y_1+1{,}3y_2-41{,}56&\leq 0\\
-0{,}46y_1+0{,}48y_2-41{,}79&\leq 0
\end{aligned}$$

Problema Binivel Fuertemente-Estacionario: ex9.2.3

mín $\quad 11{,}32x_1 + 6{,}7x_2 - 20{,}6y_1 - 24{,}61y_2 - 60$

sujeto a: $x_1 + x_2 + y_1 - 2y_2 - 16{,}45 \leq 0$ **mín** $\quad (-x_1 + y_1 + 20)^2 + (-x_2 + y_2 + 20)^2$

sujeto a:

$$\begin{aligned} -x_1 + 0{,}01y_1 + 2{,}01y_2 + 18{,}96 &\leq 0 \\ -x_2 - 0{,}6y_1 + 1{,}55y_2 + 13{,}09 &\leq 0 \\ 1{,}14y_1 + 1{,}61y_2 + 19{,}73 &\leq 0 \\ -1{,}11y_1 - 1{,}58y_2 - 19{,}37 &\leq 0 \\ 2{,}23y_1 + 0{,}67y_2 - 37{,}14 &\leq 0 \\ 0{,}66y_1 + 1{,}49y_2 - 56{,}29 &\leq 0 \end{aligned}$$

Problema Binivel Alpha-Zero: ex9.2.4

mín $\quad -8{,}45y_1 - 12{,}4y_2 + 0{,}5\,(y_1 - 2)^2 + 0{,}5\,(y_2 - 2)^2$

sujeto a:
mín $\quad 0{,}5y_1^2 + y_2$

sujeto a:

$$\begin{aligned} -x + y_1 + y_2 &\leq 0 \\ 4{,}85 - y_1 &\leq 0 \\ -y_2 - 4{,}7 &\leq 0 \end{aligned}$$

Problema Binivel C-Estacionario: ex9.2.4

mín $\quad 0{,}9x - 11{,}02y_1 + 1{,}71y_2 + 0{,}5\,(y_1 - 2)^2 + 0{,}5\,(y_2 - 2)^2$

sujeto a:
mín $\quad 0{,}5y_1^2 + y_2$

sujeto a:

$$\begin{aligned} -x + y_1 + y_2 &\leq 0 \\ 1{,}26y_1 + 0{,}3y_2 - 7{,}51 &\leq 0 \\ 0{,}13y_1 - 0{,}98y_2 - 3{,}99 &\leq 0 \end{aligned}$$

Problema Binivel M-Estacionario: ex9.2.4

mín $\quad 0{,}4x - 8{,}2y_1 - 4{,}83y_2 + 0{,}5\,(y_1 - 2)^2 + 0{,}5\,(y_2 - 2)^2$

sujeto a:
mín $\quad 0{,}5y_1^2 + y_2$

sujeto a:

$$\begin{aligned} -x + y_1 + y_2 &\leq 0 \\ 1{,}63y_1 + 1{,}45y_2 - 14{,}69 &\leq 0 \\ 1{,}63y_1 - 0{,}11y_2 - 7{,}63 &\leq 0 \end{aligned}$$

Problema Binivel Fuertemente-Estacionario: ex9.2.4

mín $5{,}2x - 1{,}8y_1 - 9{,}46y_2 + 0{,}5\,(y_1 - 2)^2 + 0{,}5\,(y_2 - 2)^2$

sujeto a:
mín $0{,}5y_1^2 + y_2$

sujeto a:

$$\begin{aligned} -x + y_1 + y_2 &\leq 0 \\ -1{,}0y_1 + 0{,}06y_2 + 4{,}56 &\leq 0 \\ 0{,}04y_1 + 0{,}16y_2 - 1{,}9 &\leq 0 \end{aligned}$$

Problema Binivel Alpha-Zero: ex9.2.5

mín $-6{,}1x + 6{,}2y + (x - 3)^2 + (y - 2)^2$

sujeto a:
mín $(y - 5)^2$

sujeto a:

$$\begin{aligned} x + 2y - 12{,}85 &\leq 0 \\ x - 2y + 3{,}35 &\leq 0 \\ -2x + y - 7{,}45 &\leq 0 \end{aligned}$$

Problema Binivel C-Estacionario: ex9.2.5

mín $\quad -4{,}2x - 7{,}43y + (x-3)^2 + (y-2)^2$

sujeto a:
mín $\quad (y-5)^2$

sujeto a:

$$\begin{aligned} x - 4{,}75 &\leq 0 \\ x + 2{,}26y - 13{,}9 &\leq 0 \\ -2x - 4{,}27y - 28{,}8 &\leq 0 \end{aligned}$$

Problema Binivel M-Estacionario: ex9.2.5

mín $\quad -3{,}3x - 28{,}3y + (x-3)^2 + (y-2)^2$

sujeto a:
mín $\quad (y-5)^2$

sujeto a:

$$\begin{aligned} x + 1{,}23y - 9{,}75 &\leq 0 \\ x + 1{,}24y - 9{,}77 &\leq 0 \\ -2x + 3{,}58y - 5{,}07 &\leq 0 \end{aligned}$$

Problema Binivel Fuertemente-Estacionario: ex9.2.5

mín $\quad 2{,}1x - 52{,}94y + (x-3)^2 + (y-2)^2$

sujeto a:
mín $\quad (y-5)^2$

sujeto a:

$$x + 6{,}83y - 32{,}42 \leq 0$$
$$x - 0{,}8y - 1{,}51 \leq 0$$
$$-2x + 1{,}17y - 6{,}78 \leq 0$$

Problema Binivel Alpha-Zero: MitsosBarton2006Ex312

mín $\quad xy + 0{,}6x + 10y^2 - 4{,}9y$

sujeto a: $1{,}9 - x \leq 0$
$x - 1{,}9 \leq 0$ **mín** $\quad -xy^2 + 0{,}5y^4$

sujeto a:

$$0{,}15 - y \leq 0$$
$$y - 0{,}15 \leq 0$$

Problema Binivel C-Estacionario: MitsosBarton2006Ex312

mín $\quad xy + 1{,}32x + 10y^2 - 11{,}63y$

sujeto a: $1{,}9 - x \leq 0$
$x - 1{,}9 \leq 0$ **mín** $\quad -xy^2 + 0{,}5y^4$

sujeto a:

$$\begin{aligned} 0{,}51 - 3{,}38y &\leq 0 \\ 0 &\leq 0 \end{aligned}$$

Problema Binivel M-Estacionario: MitsosBarton2006Ex312

mín $\quad xy + 1{,}35x + 10y^2 + 4{,}23y$

sujeto a: $1{,}9 - x \leq 0$
$x - 1{,}9 \leq 0$ **mín** $\quad -xy^2 + 0{,}5y^4$

sujeto a:

$$\begin{aligned} 2{,}53y - 0{,}38 &\leq 0 \\ 1{,}11e - 16y &\leq 0 \end{aligned}$$

Problema Binivel Fuertemente-Estacionario: MitsosBarton2006Ex312

mín $xy + 1{,}19x + 10y^2 - 0{,}35y$

sujeto a: $1{,}9 - x \leq 0$
$x - 1{,}9 \leq 0$ **mín** $-xy^2 + 0{,}5y^4$

sujeto a:

$$2{,}0y - 0{,}3 \leq 0$$
$$0 \leq 0$$

Problema Binivel Alpha-Zero: MitsosBarton2006Ex313

mín $-0{,}48x$

sujeto a: $1{,}05 - x \leq 0$
$1x - 1{,}05 \leq 0$ **mín** $-x^3y + 0{,}5xy^2$

sujeto a:

$$y - 2{,}85 \leq 0$$
$$2{,}85 - y \leq 0$$

Problema Binivel C-Estacionario: MitsosBarton2006Ex313

mín $0{,}55x - 2{,}36y$

sujeto a: $1{,}05 - x \leq 0$
$1x - 1{,}05 \leq 0$ **mín** $-x^3y + 0{,}5xy^2$

sujeto a:

$$0 \leq 0$$
$$0 \leq 0$$

Problema Binivel M-Estacionario: MitsosBarton2006Ex313

mín $0{,}88x - 3{,}12y$

sujeto a: $1{,}05 - x \leq 0$
$1x - 1{,}05 \leq 0$ **mín** $-x^3y + 0{,}5xy^2$

sujeto a:

$$0 \leq 0$$
$$0 \leq 0$$

Problema Binivel Fuertemente-Estacionario: MitsosBarton2006Ex313

mín $0{,}3x - 2{,}88y$

sujeto a: $1{,}05 - x \leq 0$
$1x - 1{,}05 \leq 0$ **mín** $-x^3y + 0{,}5xy^2$

sujeto a:

$$0{,}35y - 1 \leq 0$$
$$-4{,}44e - 16y - 16 \leq 0$$

Problema Binivel Alpha-Zero: MitsosBarton2006Ex314

mín $-1{,}7x + y^2 - 5{,}4y + (x - 0{,}25)^2$

sujeto a: $x - 0{,}9 \leq 0$

$$0{,}9 - x \leq 0$$

mín $-xy + 0{,}33y^3$

sujeto a:

$$y - 2{,}7 \leq 0$$
$$2{,}7 - y \leq 0$$

Problema Binivel C-Estacionario: MitsosBarton2006Ex314

mín $\quad -0{,}92x + y^2 - 9{,}61y + (x - 0{,}25)^2$

sujeto a: $x - 0{,}9 \leq 0$

$$0{,}9 - x \leq 0$$

mín $\quad -xy + 0{,}33y^3$

sujeto a:

$$0 \leq 0$$
$$0 \leq 0$$

Problema Binivel M-Estacionario: MitsosBarton2006Ex314

mín $\quad -1{,}43x + y^2 - 6{,}86y + (x - 0{,}25)^2$

sujeto a: $x - 0{,}9 \leq 0$

$$0{,}9 - x \leq 0$$

mín $\quad -xy + 0{,}33y^3$

sujeto a:

$$\begin{aligned} 10{,}37y - 28{,}0 &\leq 0 \\ 0 &\leq 0 \end{aligned}$$

Problema Binivel Fuertemente-Estacionario: MitsosBarton2006Ex314

mín $\quad -0{,}89x + y^2 - 17{,}08y + (x - 0{,}25)^2$

sujeto a: $x - 0{,}9 \leq 0$

$$0{,}9 - x \leq 0$$

mín $\quad -xy + 0{,}33y^3$

sujeto a:

$$\begin{aligned} 9{,}14y - 24{,}67 &\leq 0 \\ 0 &\leq 0 \end{aligned}$$

Problema Binivel Alpha-Zero: MitsosBarton2006Ex323

mín $\quad x^2 + 11{,}34x + 9{,}44y$

sujeto a: $1{,}94 - x \leq 0$

$x - 1{,}94 \leq 0$

$-9x^2 + x - y + 35{,}63 \leq 0$ **mín** $\quad y$

sujeto a:

$$\begin{aligned} y^2(x - 0{,}5) - 19{,}71 &\leq 0 \\ y - 3{,}7 &\leq 0 \\ -y - 5{,}7 &\leq 0 \end{aligned}$$

Problema Binivel C-Estacionario: MitsosBarton2006Ex323

mín $x^2 + 120{,}35x - 15{,}71y$

sujeto a: $1{,}94 - x \leq 0$
$x - 1{,}94 \leq 0$
$-9x^2 + x - y + 35{,}63 \leq 0$ **mín** y

sujeto a:

$$y^2 (x - 0{,}5) - 12{,}3y + 25{,}8 \leq 0$$
$$0 \leq 0$$
$$-2 \leq 0$$

Problema Binivel M-Estacionario: MitsosBarton2006Ex323

mín $x^2 + 5{,}99x - 1{,}64y$

sujeto a: $1{,}94 - x \leq 0$
$x - 1{,}94 \leq 0$
$-9x^2 + x - y + 35{,}63 \leq 0$ **mín** y

sujeto a:

$$y^2 (x - 0{,}5) - 10{,}11y + 17{,}69 \leq 0$$
$$0 \leq 0$$
$$4{,}28y - 16{,}26 \leq 0$$

Problema Binivel Fuertemente-Estacionario: MitsosBarton2006Ex323

mín $x^2 - 35{,}51x - 96{,}21y$

sujeto a: $1{,}94 - x \leq 0$
$x - 1{,}94 \leq 0$
$-9x^2 + x - y + 35{,}63 \leq 0$ **mín** y

sujeto a:

$$\begin{aligned} y^2(x - 0{,}5) + 3{,}63y - 33{,}14 \leq 0 \\ 0 \leq 0 \\ 8{,}57y - 32{,}2 \leq 0 \end{aligned}$$

Problema Binivel Alpha-Zero: MorganPatrone2006a

mín $0{,}21x + 0{,}6y$

sujeto a: $2{,}85 - x \leq 0$
$x - 2{,}85 \leq 0$ **mín** xy

sujeto a:

$$\begin{aligned} 4{,}45 - y \leq 0 \\ y - 4{,}45 \leq 0 \end{aligned}$$

Problema Binivel C-Estacionario: MorganPatrone2006a

mín $\quad -0{,}63x$

sujeto a: $2{,}85 - x \leq 0$
$x - 2{,}85 \leq 0$ **mín** $\quad xy$

sujeto a:

$$0 \leq 0$$
$$0 \leq 0$$

Problema Binivel M-Estacionario: MorganPatrone2006a

mín $\quad -0{,}51x$

sujeto a: $2{,}85 - x \leq 0$
$x - 2{,}85 \leq 0$ **mín** $\quad xy$

sujeto a:

$$3{,}33e - 16y - 16 \leq 0$$
$$-3{,}33e - 16y - 16 \leq 0$$

Problema Binivel Fuertemente-Estacionario: MorganPatrone2006a

mín $-2{,}07x - 17{,}89y$

sujeto a: $2{,}85 - x \leq 0$
$x - 2{,}85 \leq 0$ **mín** xy

sujeto a:

$$2{,}98y - 13{,}27 \leq 0$$
$$-3{,}33e - 16y - 16 \leq 0$$