Universidad Tecnológica de la Mixteca

"Algoritmo de Espacio Rango Para Programación Cuadrática."

TESIS

Que para obtener el título de:

Ingeniero en Computación

Presenta:

Ivor Acevedo Bautista.

Asesor:

M. C. Marcela Rivera Martínez.

Huajuapan de León, Oax. Julio de 2005.

DEDICATORIA

CON TODO MI CARIÑO, ADMIRACIÓN Y AGRADECIMIENTO DEDICO ESTA TESIS A MIS PADRES, TOMAS JOSÉ ACEVEDO ROSAS Y RUTH ROCÍO BAUTISTA HERNÁNDEZ, GRACIAS POR EL GRAN EJEMPLO QUE ME HAN DADO, POR TODO EL APOYO RECIBIDO PARA IR FORJANDO MI PROPIO DESTINO, POR DARME LAS HERRAMIENTAS NECESARIAS, POR ENSEÑARME EL CAMINO DEL BIEN Y SOBRE TODO POR TODO SU AMOR. GRACIAS.

CON TODO MI AMOR A CELIA, GRACIAS PRINCESA POR ESTAR A MI LADO, POR CREER EN MI, POR ENSEÑARME A LUCHAR POR LO QUE QUIERO, PERO SOBRE TODO, POR HABER LLEGADO A TIEMPO A MI VIDA.

AGRADECIMIENTOS

A MIS ABUELITOS: CONCHITA Y GONZALO.

POR SU AMOR INFINITO E INCONDICIONAL.

A MI FAMILIA.

POR TODO EL CARIÑO, CONSEJOS, ALIENTO Y APOYO EN CADA UNO DE LOS PROYECTOS QUE HE EMPRENDIDO.

A MI ASESORA: M. C. MARCELA RIVERA MARTÍNEZ.

POR BRINDARME LA OPORTUNIDAD DE HACER ESTA TESIS, POR SUS CONSEJOS Y SOBRE TODO POR LA PACIENCIA Y ORIENTACIÓN DURANTE EL DESARROLLO DE LA MISMA.

A MIS COMPAÑEROS Y AMIGOS: ERIK, CARLOS FRANCISCO, HUGO, CARLOS Y SAYDE.

POR SU APOYO INCONDICIONAL DURANTE MIS ESTUDIOS PROFESIONALES, PERO SOBRE TODO POR SU AMISTAD SINCERA.

A MIS SINODALES, CON EL RESPETO QUE SE MERECEN.

Índice general

Índice de cuadros

Índice de figuras

Introducción.

La optimización es la búsqueda de la mejor solución (la óptima) a un problema. La mejor solución depende del área sobre la cual se está trabajando, puede significar la solución óptima para minimizar costos, maximizar beneficios, o encontrar la ruta recorrida que sea mínima, toda vez que estos y otros problemas pueden ser planteados de forma numérica, mediante un modelo de programación matemática. La optimización, a su vez, se divide en dos partes muy importantes: la programación lineal y la no lineal.

La programación no lineal viene a significar la colección de metodologías asociadas con cualquier problema de optimización donde las relaciones no lineales se presentan en la función objetivo y/o en las restricciones.

La programación cuadrática es un caso particular de la programación no lineal, al que se le pueden aplicar técnicas de programación no lineal, como pueden ser: de restricciones activas, gradientes conjugados, estimación de los multiplicadores de Lagrange, etc.

La programación cuadrática forma parte esencial de la programación no lineal debido a la gran cantidad de áreas en donde se aplica.

Encontrar el punto más alto de una pirámide puede ser visto como un problema de programación no lineal; se asume que se pueden encontrar ecuaciones de los planos que contienen los lados y la base de la pirámide. Entonces, la pirámide es esencialmente la región factible, es decir, el conjunto de puntos ubicados dentro del volumen, contenida por esos planos.

Un ejemplo análogo es encontrar el punto más profundo de un lago. Se puede ver el litoral como las restricciones, la región factible como la superficie del agua y la función objetivo como la profundidad. Este ejemplo puede ser altamente no lineal.

Hancock (1960) indicó que “todos los problemas de mecánica pueden ser reducidos a problemas de máximos y mínimos.” Principios en óptica, física cuántica, astronomía, química, biología, etc., pueden usualmente ser formulados en términos de principios “extremos” (es decir máximo

o mínimo), por ejemplo, un camino de la resistencia menor, energía mínima, entropía máxima, etc.

En este trabajo de tesis se instrumenta el "Algoritmo de Espacio Rango para Programación Cuadrática", este algoritmo es un algoritmo primal basado en la estrategia de conjunto activo, los problemas que se resuelven son del tipo:

$$\begin{aligned} \text{min.} \qquad & f(x) = \tfrac{1}{2} x^T Q x + c^T x \\ s.a. \qquad & A^T x \geq b \end{aligned}$$

La función cuadrática $f(x)$ es estrictamente convexa, por lo que la matriz Hessiana Q_{nxn} es definida positiva, A_{mxn} es la matriz de restricciones, b_m el vector de términos independientes y c_n el vector de costos.

Este algoritmo trabaja con un método primal y utiliza la estrategia de conjunto activo para encontrar la solución óptima, utilizando las condiciones de Karush-Kuhn-Tucker para determinar si la solución encontrada puede ser considerada como una solución óptima.

En 2001 dentro de la Universidad Tecnológica de la Mixteca se instrumentó el "Algoritmo de Espacio Nulo para Programación Cuadrática"[1], que resuelve el mismo tipo de problemas, se realizó en Matlab ver. 5.1. Dentro del presente trabajo de tesis se corrieron los mismos problemas prueba con el algoritmo de espacio nulo, con el objetivo de comparar los resultados de ambos métodos y así poder determinar la eficiencia de cada uno de ellos, que por la naturaleza de los problemas prueba (bien condicionados, mal condicionados, con degeneración y sin degeneración), es posible que se pueda utilizar alguno de los algoritmos para cada tipo de problema en particular.

El trabajo está organizado como sigue: en el capítulo 1 se definen los conceptos básicos necesarios para comprender el trabajo realizado, así como una descripción de lo que es la programación cuadrática.

El capítulo siguiente, 2, contiene los métodos y algoritmos instrumentados: primal, de conjunto activo y el de espacio rango para programación cuadrática.

En el capítulo 3 se encuentran los resultados obtenidos al realizar las pruebas con el algoritmo de espacio rango para programación cuadrática, así como las gráficas que muestran estos resultados para una mejor interpretación.

Las conclusiones de este trabajo se encuentran en el capítulo 4.

La parte final de este trabajo contiene los apéndices. En el apéndice A se muestra un ejemplo gráfico para ilustrar cómo trabaja el algoritmo, en el B, se presenta el manual de usuario, que describe el funcionamiento general del sistema (AERPC).

Y finalmente, se presenta la bibliografía consultada durante el desarrollo del presente trabajo.

Capítulo 1

Generalidades.

El objetivo de este trabajo es resolver problemas del tipo:

$$\begin{aligned} \text{min.} \qquad & f(x) = \tfrac{1}{2} x^T Q x + c^T x \\ s.a. \qquad & A^T x \geq b, \end{aligned} \tag{1.1}$$

con la función cuadrática estrictamente convexa, lo que implica que la matriz Hessiana sea definida positiva. Para poder conocer y comprender lo que busca la programación cuadrática, es necesario conocer la definición y darle una buena interpretación a los términos que se emplean y que se tomaron en cuenta para implementar el "Algoritmo de Espacio Rango para Programación Cuadrática". Estas definiciones fueron tomadas de [2], [3], [4] y [5].

1.1. Optimización con restricciones.

A continuación se definen los términos utilizados en la solución del problema del tipo (1.1).

1.1.1. Condiciones necesarias y suficientes de optimalidad.

Para los métodos utilizados en resolver problemas cuadráticos es necesario definir condiciones suficientes y necesarias, que se deben satisfacer en el punto solución, para que pueda ser considerado como un punto de solución óptimo. La necesidad de estas condiciones nos lleva principalmente al estudio de las teorías de Lagrange y de Karush-Khun y Tucker, que caracterizan las condiciones de un punto óptimo en programación no lineal.

Multiplicadores de Lagrange. La idea principal en el desarrollo de condiciones necesarias y suficientes, para los problemas de optimización no lineal con restricciones es, el de transformar estos en problemas sin restricciones y aplicar las condiciones necesarias y suficientes para los problemas sin restricciones. Una transformación involucra la introducción de una función auxiliar, llamada la función de Lagrange $L(x, \lambda, \mu)$, definida como:

$$L(x, \lambda, \mu) = f(x) + \lambda^T h(x) + \mu g(x),\ \mu \geq 0,$$

donde $\lambda^T = (\lambda_1, \lambda_2, ..., \lambda_m)$ y $\mu^T = (\mu_1, \mu_2, ..., \mu_p)$ son los multiplicadores de Lagrange asociados con las restricciones de igualdad y desigualdad, respectivamente. Los multiplicadores λ, asociados con las igualdades $h(x) = 0$ no tienen restricciones de signo, mientras que los multiplicadores μ, asociados con las desigualdades $g(x) \geq 0$ deben ser no negativos.

La transformación a problemas sin restricciones comienza encontrando el punto estacionario de la función Lagrangiana:

$$\min_{x,\lambda,\mu \geq 0} L(x, \lambda, \mu) = f(x) + \lambda^T h(x) + \mu g(x), \mu \geq 0.$$

Los multiplicadores de Lagrange λ proveen información sobre la sensibilidad de la función objetivo con respecto al vector de perturbación b al punto óptimo x^*, del problema (1.1). La existencia de los multiplicadores de Lagrange depende de la forma de las restricciones y no siempre está garantizada.

Condiciones de Karush-Kuhn-Tucker (KKT). Las condiciones de primer orden para que un punto pueda ser considerado un mínimo local son también llamadas condiciones de Karush-Kuhn-Tucker (KKT).

El sistema KKT, fué propuesto por Karush, Kuhn y Tucker, para resolver y establecer condiciones que permitan determinar si la solución a la que se llega, al resolver problemas de programación cuadrática del tipo (1.1), es la óptima.

Supóngase que x^* es un punto regular para las restricciones. Entonces, existen un vector λ^* y un vector $\mu^* \in R^m$ que satisfacen:

$$\nabla f(x^*) + \lambda^T \nabla h(x^*) + \mu^T \nabla g(x^*) = 0 \quad 1.2$$
$$\mu^T g(x^*) = 0 \quad 1.3$$
$$\mu^T \geq 0 \quad 1.4$$

Demostración [3]:

En primer lugar, como $\mu \geq 0$ y $g(x^*) = 0$, se observa que (1.3) es equivalente a la formulación de que una componente de μ puede ser distinta de cero sólo si la restricción co– rrespondiente es activa.

Como x^* es un punto mínimo relativo en el conjunto de restricciones, también es un mínimo relativo en el subconjunto de ese conjunto definido, igualando las restricciones activas a cero. Así, para el problema resultante con restricciones de igualdad, definido en un entorno de x^*, hay multiplicadores de Lagrange. Por tanto, se concluye que (1.2) cumple con $\mu_j = 0$ si $g_j(x^*) \neq 0$, y por tanto, también se cumple (1.3).□

1.1.2. Condiciones de segundo orden.

Las condiciones de KKT, si bien es cierto que definen a un buen candidato para ser un mínimo, no garantizan que este pueda serlo debido a que sólo son consideradas como necesarias.

A continuación se definen las condiciones necesarias y suficientes de segundo orden para asegurar que el punto encontrado es un mínimo.

Teorema 4. (Condiciones necesarias de segundo orden).

Supóngase las funciones $f, g, h \in C^2$ y que x^* es un punto regular de (1.1). Si x^* es un punto mínimo relativo para el problema, entonces existe $\lambda \in R^m, \mu \in R^m, \mu \geq 0$ tales que cumplen (1.2) y (1.3) y la funcion Lagrangiana, determinada por:

$$L(x^*) = f(x^*) + \lambda^T h(x^*) + \mu^T g(x^*),$$

es semidefinida positiva en el subespacio tangente de las restricciones que se cumplen en la igualdad en x^*.

Demostración [3]:

Si x^* es un punto mínimo relativo del problema (1.1), también es un punto mínimo relativo para el problema con las restricciones de desigualdad que se cumplen en la igualdad tomadas como restricciones de igualdad.

Se puede formular una inversa del teorema de la condición necesaria de segundo orden para obtener un teorema de la condición suficiente de segundo orden. Por analogía de la situación sin restricciones, se puede esperar que la hipótesis requerida sea que $L(x^*)$ sea definida positiva en el plano tangente M. De hecho, esto es suficiente en la mayoría de las situaciones. Sin embargo, si hay restricciones de desigualdad degeneradas (esto es, restricciones de desigualdad activas que tengan cero como multiplicador de Lagrange asociado), se debe exigir que $L(x^*)$ sea definida positiva en un subespacio mayor que M.□

Teorema 5. (Condiciones suficientes de segundo orden).

Sean $f, g, h \in C^2$. Las condiciones suficientes para que un punto x^* que satisfaga las restricciones del problema (1.1) sea un punto mínimo relativo estricto del problema, es que exista $\lambda \in R^m, \mu \in R^m$, tal que:

$$\nabla f(x^*) + \lambda^T \nabla h(x^*) + \mu^T \nabla g(x^*) = 0$$
$$\mu^T g(x^*) = 0$$
$$\mu^T \geq 0,$$

y la matriz Hessiana

$$L(x^*) = f(x^*) + \lambda h(x^*) + \mu^T g(x^*) \ ,$$

sea definida positiva en el subespacio

$$M' = \{y : \nabla h(x^*)y = 0, \nabla g(x^*)y = 0 \text{ para toda } j \in J \ \},$$

donde

$$J = \big\{j : g_j(x^*) = 0, \mu_j > 0\big\}.$$

Demostración [3]:

Supóngase que x^* no es un punto mínimo relativo estricto; sea $\{y_k\}$ una sucesión de puntos factibles que converge a x^* tal que $f(y_k) \leq f(x^*)$, y escríbase cada y_k en la forma $y_k = x^* + \delta_k s_k$ con $|s_k| = 1, \delta > 0$. Se puede suponer que $\delta \to 0$ y $s_k \to s^*$. Se tiene $0 \geq \nabla f(x^*)s^*$, para cada $i = 1, ..., m$, y resulta

$$\nabla h_i(x^*)s^* = 0.$$

Además, para cada restricción activa g_j se tiene $g_j(y_k) - g_j(x^*) \leq 0$, y por tanto,

$$\nabla g_j(x^*)s^* \leq 0.$$

Si $\nabla g_j(x^*)s^* = 0$ para toda $j \in J$, entonces por el teorema de Taylor, para cada j se tiene

$$0 = h_j(y_k) = h_j(x^*) + \delta_k \nabla h_j(x^*) s_k + \frac{\delta_k^2}{2} s_k^T \nabla^2 h_j(\eta_j) s_k \tag{1.5}$$

y

$$0 \geq f_j(y_k) - f(x^*) = \delta_k \nabla f_j(x^*) s_k + \frac{\delta_k^2}{2} s_k^T \nabla^2 f_j(\eta_j) s_k, \tag{1.6}$$

donde cada η_j es un punto en el segmento de recta que une x^* e y_k. Al multiplicar (1.5) por λ_j y sumar esto a (1.6), teniendo en cuenta que

$$\nabla f(x^*) + \lambda^T \nabla h(x^*) = 0,$$

se obtiene

$$0 \geq \tfrac{\delta_k^2}{2} s_k^T \{\nabla^2 f(\eta_0) + \textstyle\sum_{i=1}^{m} \lambda_i \nabla^2 h_i(\eta_i)\} s_k,$$

lo cual produce una contradicción cuando $k \to \infty$.

Si $\nabla g_j(x^*) s^* < 0$ para al menos, una $j \in J$, entonces

$$0 \geq \nabla f(x^*) s^* = -\lambda^T \nabla h(x^*) s^* - \mu^T \nabla g(x^*) s^* > 0,$$

lo cual es una contradicción.□

Se observa, sobre todo, que si todas las restricciones de desigualdad tienen multiplicadores de Lagrange correspondientes estrictamente positivos (sin desigualdades degeneradas), entonces, el conjunto J incluye todas las desigualdades que se cumplen en la igualdad. En este caso, la condición suficiente es que el lagrangiano sea semidefinido positivo en M, el plano tangente de las restricciones que se cumplen en la igualdad.

1.2. Programación Cuadrática.

En esta sección, definiremos lo que es y las características de la programación cuadrática, además de que se describirán los algoritmos existentes para resolver este tipo de problemas. [6], [7] y [8].

La importancia de la programación cuadrática recae en que, como es un caso especial de la programación no lineal, se utiliza como una función modelo para aproximar funciones no lineales a través de modelos locales.

La programación cuadrática trabaja con una clase especial de problemas en el que una función cuadrática de variables de decisión sujeta a restricciones lineales de desigualdad requiere ser optimizada, en nuestro caso, requiere ser minimizada.

Una función cuadrática, en notación matricial, es una función de la forma

$$f(x) = \tfrac{1}{2}x^T Qx + c^T x.$$

Es de gran importancia identificar o poder definir la característica de la matriz Hessiana, ya que a partir de ésta podemos determinar ciertas características del problema, que nos serán útiles para encontrar su solución.

Existen diferentes tipos de problemas de programación cuadrática, los cuales se pueden clasificar en:

Problemas cuadráticos de minimización sin restricciones, requieren minimizar la función cuadrática $f(x)$ sobre el espacio completo.

Problemas cuadráticos de minimización sujetos a restricciones de igualdad, requieren minimizar la función objetivo $f(x)$ sujeta a restricciones lineales de igualdad $Ax = b$.

Problemas cuadráticos de minimización sujetos a restricciones lineales de desigualdad. Requieren minimizar la función objetivo $f(x)$ sujeta a restricciones lineales de desigualdad $Ax \geq b$, también puede contener restricciones de igualdad.

Problemas de optimización de redes cuadráticas. Son problemas cuadráticos en los que las restricciones son restricciones de baja conservación sobre una red pura o generalizada.

Problemas cuadráticos convexos. Son cualesquiera de los mencionados arriba, en el cual la función objetivo a ser minimizada, $f(x)$ es convexa.

Problemas cuadráticos no convexos. Son cualesquiera de los mencionados arriba, en el cual la función objetivo a ser minimizada, $f(x)$ es no convexa.

Problemas de complementariedad lineal. Son problemas especiales con un sistema de ecuaciones en variables no negativas, en el cual las variables están formadas en varios pares llamados pares complementarios.

El "Algoritmo de Espacio Rango para Programación Cuadrática" aquí implementado, trabaja con problemas cuadráticos convexos sujetos a restricciones lineales de desigualdad.

Historicamente, las funciones cuadráticas fueron prominentes porque proveían modelos locales simples para funciones no lineales generales. Una función cuadrática, es la función no lineal más simple, y cuando es usada como una aproximación para una función no lineal general, esta puede capturar la información importante de la curvatura, lo que una aproximación lineal no puede.

El uso de aproximaciones cuadráticas para resolver problemas con funciones no lineales generales se remonta mucho tiempo atrás. Entre los métodos más destacados, tenemos al método de Newton y el método de gradiente conjugado.

Para la programación cuadrática se pueden encontrar mínimos locales, mínimos globales, puntos estacionarios o de KKT, (son los que satisfacen las condiciones de KKT del problema). En problemas convexos de programación cuadrática, todo punto KKT o mínimo local, es un mínimo global.

La programación cuadrática tiene aplicaciones muy importantes, por ejemplo, en el área financiera, se pueden realizar análisis, usando modelos de programación cuadrática para determinar la selección de estrategias óptimas de inversión. En los impuestos, la programación

cuadrática juega un papel muy importante en el análisis de políticas de impuestos. Otra aplicación importante, es en la que los economistas utilizan modelos de equilibrio para analizar expectativas de cambio en condiciones económicas, predicción de precios, incremento de la inflación, etc. Estos modelos involucran el uso de programación cuadrática.

Capítulo 2

Métodos y Algoritmos.

2.1. Métodos primales.

Por un método primal [2], se entiende un método de búsqueda que opera sobre el problema original, buscando la solución óptima directamente en la región factible. Cada punto que se genera, mediante un método primal, es factible y el valor de la función objetivo decrece en cada iteración. Para un problema de n variables y solo m restricciones de igualdad, los métodos primales operan en el espacio factible, que tiene dimensión $n - m$.

Las principales ventajas de los métodos primales y que recomiendan su utilización, como procedimientos aplicables a la mayoría de los problemas de programación no lineal son tres:

1. Como cada punto generado durante el proceso es factible, si el proceso termina antes de alcanzar la solución, el punto en el que se termina es factible y probablemente está cerca de la solución óptima del problema original, por lo que puede representar una solución aceptable para el problema originado por el programa no lineal.

2. La segunda característica a tomar en cuenta es que, la tasa de convergencia global de estos métodos suele ser satisfactoria.

3. Los métodos primales no dependen de una estructura especial del problema, tales como la convexidad, de ahí que estos métodos sean aplicables a los problemas de programación no lineal en general.

Sin embargo, los métodos primales también tienen algunos inconvenientes. Necesitan de un punto inicial factible, por lo que necesitan un procedimiento de fase I y están llenos, en particular los de restricciones no lineales, de dificultades computacionales originadas por la necesidad de permanecer dentro de la región factible.

La tasa de convergencia de este tipo de métodos, puede competir con la de cualquier otro método, y para restricciones lineales, se encuentra entre los métodos más eficientes.

2.2. Método de conjunto activo.

El método de conjunto activo [2] es un método iterativo que trabaja con las restricciones que son activas en cada iteración del método. Las restricciones inactivas son simplemente ignoradas.

Considérese el problema con restricciones

$$\begin{aligned} \text{mín}. \qquad & f(x) = \tfrac{1}{2}x^T Q x + c^T x \\ s.a. \qquad & A^T x \geq b \end{aligned} \tag{2.1}$$

Las condiciones necesarias para este problema son:

$$\begin{aligned} \nabla f(x^*) + \lambda^{*T} \nabla g(x^*) &= 0 \\ g(x^*) &\geq 0 \\ \lambda^{*T} g(x^*) &= 0 \\ \lambda^* &\geq 0, \end{aligned}$$

en función del conjunto de restricciones que se cumplen en la igualdad, estas condiciones se pueden presentar de forma más sencilla. Se denota por N al conjunto de índices para restricciones que se cumplen en la igualdad. Entonces las restricciones se convierten en:

$$\begin{aligned} \nabla f(x) + \lambda^T \nabla g(x) &= 0 \\ g_i(x) &= 0, \quad i \in N \\ g_i(x) &> 0, \quad i \notin N \\ \lambda_i &\geq 0, \quad i \in N \\ \lambda_i &= 0, \quad i \notin N \end{aligned}$$

si se conociera el conjunto activo, el problema original se podría sustituir por el correspondiente que tenga solo condiciones de igualdad. Entonces, una solución correcta implica el que las otras restricciones se satisfagan y los multiplicadores sean no negativos.

Este tipo de métodos, definen en cada fase o en cada paso de un algoritmo un conjunto de restricciones activas llamado conjunto de trabajo. El conjunto de trabajo está compuesto por el subconjunto de restricciones que son activas en el punto actual y por lo tanto, este punto es factible para el conjunto de trabajo. A partir del nuevo punto obtenido, el algoritmo comienza a moverse, dentro de la región factible, hacia un punto mejor. En este nuevo punto, puede ser que el conjunto de trabajo cambie, es decir, que se vuelvan activas algunas restricciones en este punto, por lo que hay que buscar la manera de introducirlas al conjunto de trabajo, o tal vez sea necesario eliminar una restricción del conjunto de trabajo.

En cada nuevo punto o en cada fase hay que definir el nuevo conjunto de trabajo. Evidentemente, el conjunto de trabajo no necesariamente tiene que coincidir con el conjunto activo.

2.3. Método de espacio rango.

El método de espacio rango [9] es el que se utiliza para resolver problemas del tipo (2.1), determina el conjunto de trabajo, obtiene la dirección factible, el punto factible de cada iteración y los multiplicadores asociados a cada uno de ellos. Este método toma su nombre debido a que,

durante la solución del sistema KKT, se introducen los términos ya conocidos como son el rango de una matriz y sistema. Este algoritmo trabaja con matrices de rango n, es decir, que todas sus líneas (filas o columnas) son linealmente independientes y además, todos los puntos obtenidos con este método siguen formando parte del sistema definido por la función cuadrática convexa y se pueden realizar las operaciones básicas de producto y adición.

Este método recibe como entrada un punto factible (x), un conjunto de restricciones (A) y la matriz Hessiana (Q). Como se trabaja con un método primal de conjunto activo, es necesario obtener, a partir de (A), el conjunto de restricciones que en el punto inicial (x) son activas, llamado conjunto de trabajo (N). Una vez obtenido este conjunto, se calculan los multiplicadores de Lagrange (λ) asociados a este punto, y la dirección de búsqueda (d), hacia donde se deberá mover el algoritmo para poder encontrar un punto que se acerque más al óptimo. Con estos datos, es posible determinar si el punto factible (x) cumple con las condiciones de KKT, en cuyo caso se dará por terminada la búsqueda con el punto actual como el óptimo (x^*). Si no se cumplen estas condiciones, y no existe una dirección hacia donde moverse, es necesario eliminar la restriccion asociada al multiplicador de Lagrange más negativo (paso parcial) y volver a calcularla y se procede a comprobar si ya se cumplen las condiciones de optimalidad. En el caso de que exista una dirección de búsqueda, se calcula la longitud de paso (α), el nuevo punto factible (x), se agrega la restriccion mas próxima a ser activa (paso completo) y nuevamente se comprueba si el punto actual es el óptimo. De esta forma se sigue iterando hasta encontrar el punto que satisfaga las condiciones KKT, es decir, el óptimo (x^*). A continuación se escriben detalladamente cada uno de los cálculos para los pasos anteriores.

2.3.1. Cálculo de la dirección factible y los multiplicadores de Lagrange.

En el Algoritmo de Espacio Rango para Programación Cuadrática, cabe recordar que se trabaja con un método de conjunto activo, por lo que las restricciones asociadas al subproblema a resolver son de igualdad, así el problema tiene la forma:

$$\begin{aligned} \text{mín}. \qquad & f(x) = \tfrac{1}{2}x^T Q x + c^T x \\ s.a. \qquad & N^T x = b, \end{aligned} \tag{2.2}$$

donde N es la matriz de restricciones activas. Aplicando las condiciones de optimalidad de primer orden (condiciones KKT), obtenemos:

$$\begin{aligned} Qx + c - (\lambda^T N)^T &= 0. \\ Nx - b &= 0. \end{aligned} \tag{2.3}$$

Resolviendo (2.3) en términos de la solución óptima x^* y del vector de los multiplicadores de Lagrange λ^*, tenemos:

$$\begin{bmatrix} Q & -N^T \\ N & 0 \end{bmatrix} \begin{bmatrix} x^* \\ \lambda^* \end{bmatrix} = \begin{bmatrix} -c \\ b \end{bmatrix}. \tag{2.4}$$

Nuevamente se puede reescribir (2.4) de tal forma que facilite su solución haciendo $x^* = x+d$, $g = Qx + c$ y $c' = Nx - b$, donde x es una estimación de la solución óptima. Despejando y sustituyendo en (2.4) tenemos:

Despejando c para la primera ecuación de (2.4) tenemos:

$$-c = Qx^* - N^T\lambda^*. \tag{2.5}$$

Sustityendo x^* y g se obtiene:

$$\begin{aligned} Qx^* + c - N^T\lambda^* &= 0 \\ Q(x+d) + c - N^T\lambda^* &= 0 \\ Qx + Qd + c - N^T\lambda^* &= 0 \end{aligned}$$

$$Qx + c = -Qd + N^T\lambda^*$$

$$g = -Qd + N^T\lambda^*. \tag{2.6}$$

Despejando b de la segunda ecuación de (2.3) y en términos de la solución óptima:

$$b = Nx^*. \tag{2.7}$$

Sustituyendo x^* y c'en (2.7), se obtiene:

$$b = N(x + d)$$
$$b = Nx + Nd$$
$$Nx - b = -Nd$$

$$c' = -Nd. \tag{2.8}$$

reescribiendo en notación matricial tenemos:

$$\begin{bmatrix} Q & N^T \\ N & 0 \end{bmatrix} \begin{bmatrix} -d \\ \lambda^* \end{bmatrix} = \begin{bmatrix} g \\ c' \end{bmatrix}. \tag{2.9}$$

Este sistema, denominado sistema KKT, fue propuesto por Karush, Kuhn y Tucker, y es el que se resuelve para poder encontrar la solución del problema de programación cuadrática.

Para resolver el problema de programación cuadrática (2.2), existen diferentes métodos, solución directa, método de espacio nulo, método de espacio rango, método basado en conjugancia, etc. Se determinó utilizar el método de espacio rango para resolver el sistema, ya que por la característica de la matriz Hessiana Q de ser definida positiva, la matriz es fácil de invertir.

A partir del problema de programación cuadrática (2.2), y sabiendo que Q es definida positiva, multiplicamos la primera ecuación del sistema KKT (2.6) por NQ^{-1}

$$NQ^{-1}\left(-Qd + N^T\lambda^* - g\right) = 0$$
$$-NQ^{-1}Qd + NQ^{-1}N^T\lambda^* - NQ^{-1}g = 0$$

$$-Nd + NQ^{-1}N^T\lambda^* = NQ^{-1}g \tag{2.10}$$

sustituyendo 2.8 en 2.10:

$$c' + NQ^{-1}N^T\lambda^* = NQ^{-1}g$$

$$NQ^{-1}N^T\lambda^* = NQ^{-1}g - c' \tag{2.11}$$

despejando λ de (2.11) y d de (2.6), se obtiene la forma de calcular los multiplicadores de Lagrange y la dirección factible:

$$\lambda^* = \left(NQ^{-1}NT^T\right)^{-1}\left(NQ^{-1}g - c'\right) \tag{2.12}$$

$$d = Q^{-1}\left(N^T\lambda^* - g\right) \tag{2.13}$$

Donde g representa al vector gradiente y está dado por:

$$g = Qd + c'$$

Como se puede apreciar, se tiene que calcular la inversa de Q, pero como es definida positiva, se tiene la certeza que existe y que se puede llegar a una solución.

2.3.2. Cálculo de la longitud de paso.

Para obtener el siguiente punto factible del problema, hay que tomar en cuenta que, como se trata de un problema con restricciones de desigualdad, se puede ubicar fuera de la región factible; por lo tanto, es necesario determinar una longitud de paso máxima que nos asegure que esto no suceda y no viole ninguna de estas restricciones. Así, dada $x_k \epsilon \Omega$, se puede decir que un vector d_k es una dirección factible en x si existe algún $\bar{\alpha} > 0$ tal que:

$$x_k = x_{k-1} + \alpha_{k-1} d_{k-1},$$

para toda $\alpha, 0 \leq \alpha \leq \bar{\alpha}$, donde α representa la longitud de paso y d la dirección factible.

Para determinar la longitud de paso y conociendo que la función objetivo es cuadrática, se tienen dos opciones. La primera es forzar una longitud de paso igual a la unidad, si se toma este valor en la siguiente iteración se obtendrá un valor constante para el punto factible encontrado con relación a las restricciones de igualdad. En el segundo caso, la longitud de paso debe ser menor a la unidad, tomando la longitud de la restricción más cercana, con lo cual se debe integrar esta restricción al conjunto de trabajo de la siguiente iteración. El cálculo para esta longitud de paso está dado por:

$$\alpha_k = \min_{a_i^T d_k > 0} \left\{ 1, \frac{b_i - a_i^T x_k}{a_i^T d_k} \right\}.$$

2.4. Algoritmos instrumentados.

2.4.1. Algoritmo primal de conjunto activo.

En esta sección se presentan los detalles del algoritmo primal de conjunto activo, instrumentado para resolver problemas del tipo 2.1, se necesita como entrada una solución factible x, su conjunto de restricciones A, la matriz hessiana Q, el vector de costos c y el vector de términos

independientes b. Se denota al índice de las restricciones activas como vca, el de restricciones no activas como $vcna$ y el vector gradiente evaluado en en el punto x_k como g_k.

El algoritmo primal de conjunto activo trabaja con un conjunto llamado de trabajo, que contiene las restricciones que son activas, es un método iterativo, en cada iteración, obtiene un punto factible y cambia el conjunto de trabajo.

Paso 1:

Se realiza el cálculo del índice del conjunto activo vca, el índice del conjunto no activo $vcna$ y el vector gradiente g.

Paso 2:

Se resuelve el sistema KKT usando el algoritmo de espacio rango (sección 2.4.2), para obtener el valor de d_k (dirección de búsqueda) y λ_k(multiplicadores asociados a la solución factible, x_k).

Dependiendo de la naturaleza de d_k se elige:

Si $d_k = 0$, continuar con el paso 3.
Si $d_k \neq 0$, ir al paso 5.

Paso 3:

Si con los multiplicadores de Lagrange se comprueba que $\lambda_k \geq 0$, se termina el algoritmo con el punto x_k como el punto óptimo. En caso de que $\lambda_k < 0$, el algoritmo continúa con el paso 4.

Paso 4:

Se elimina la restricción del conjunto de trabajo con el λ asociado más negativo, se actualizan las variables relacionadas y se regresa al paso 2.

Paso 5:

Se calcula la longitud de paso máxima, usando el algoritmo para el cálculo de la longitud de paso (sección 2.4.3), α_k no negativa, más adecuada para asegurar la factibilidad del nuevo punto.

Paso 6:

Se calcula el punto siguiente x_{k+1}, se evalúa el vector gradiente g y se actualiza el número de iteración k.

$$x_{k+1} \leftarrow x_k + \alpha_k d_k.$$
$$g \leftarrow Q(x_{k+1}) + c.$$
$$k \leftarrow k + 1.$$

Paso 7:

Se verifica si existe una restricción que se vuelva activa. Si no se vuelve activa ninguna restricción, se mantiene el conjunto de trabajo actual y se continúa con al paso 2.

Si se vuelve activa alguna restricción se continúa con el paso 8.

Paso 8:

Si existe más de una restricción nueva, se selecciona el índice de la más proxima a cumplirse en la igualdad, se añade a *vca* y se continúa con el paso 2.

2.4.2. Algoritmo de Espacio Rango.

Los pasos que se detallan a continuación son los empleados para determinar los multiplicadores de Lagrange y encontrar la dirección factible, que servirán para determinar en qué momento el algoritmo de Espacio Rango para Programación Cuadrática puede terminar y decir que se ha encontrado el punto óptimo:

Paso1:

Se calcula el conjunto de restricciones activas (conjunto de trabajo) a partir de vca. N denota la matriz de restrcciones activas.

Paso 2:

Se calculan los multiplicadores de Lagrange utilizando el método de Espacio Rango:

$$\lambda_k \leftarrow \left(N_k Q^{-1} N_k^T\right)^{-1} \left(N_k Q^{-1} g_k - c'\right).$$

Paso 3:

Se calcula la dirección de búsqueda:

$$d_k \leftarrow Q^{-1}\left(N_k^T \lambda_k - g_k\right).$$

Paso 4:

Termina el algoritmo, retornando los valores de λ_k y d_k.

Algoritmo para el cálculo de la longitud de paso.

A continuación se describen los pasos para obtener la longitud de paso máxima a través de la dirección de búsqueda:

Paso 1:

Se calcula el conjunto de restricciones y términos independientes fuera del conjunto activo CNA y CNB respectivamente

Paso 2:

Se calcula la longitud de paso α_k:

$$\alpha_k \leftarrow \left\{\frac{b_i - a_i^T x_k}{a_i^T d_k}\right\}.$$

Paso 3:

Se determina la longitud de paso máxima:

$$\alpha_k \leftarrow \min_{a_i^T d_k > 0} \left\{1, \frac{b_i - a_i^T x_k}{a_i^T d_k}\right\}.$$

Paso 4:

Termina el algoritmo retornando el valor de α_k

2.4.3. Diagramas de flujo.

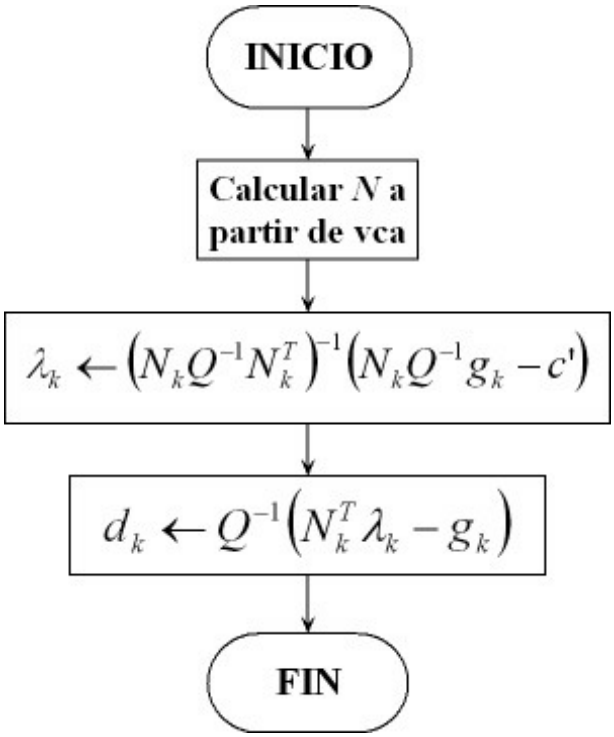


Figura 2-1: Diagrama de flujo del algoritmo de espacio rango.

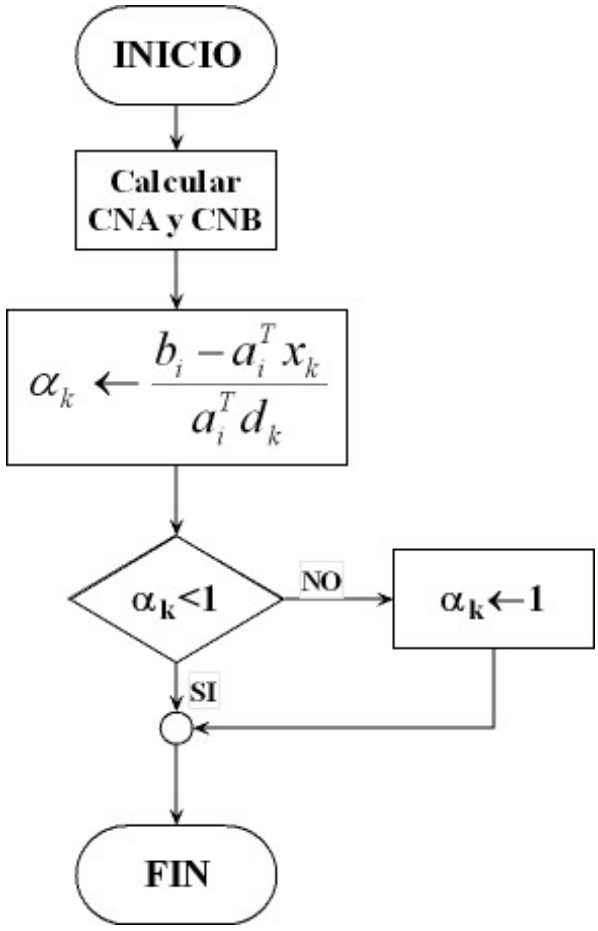


Figura 2-2: Diagrama de flujo para el cálculo de la longitud de paso

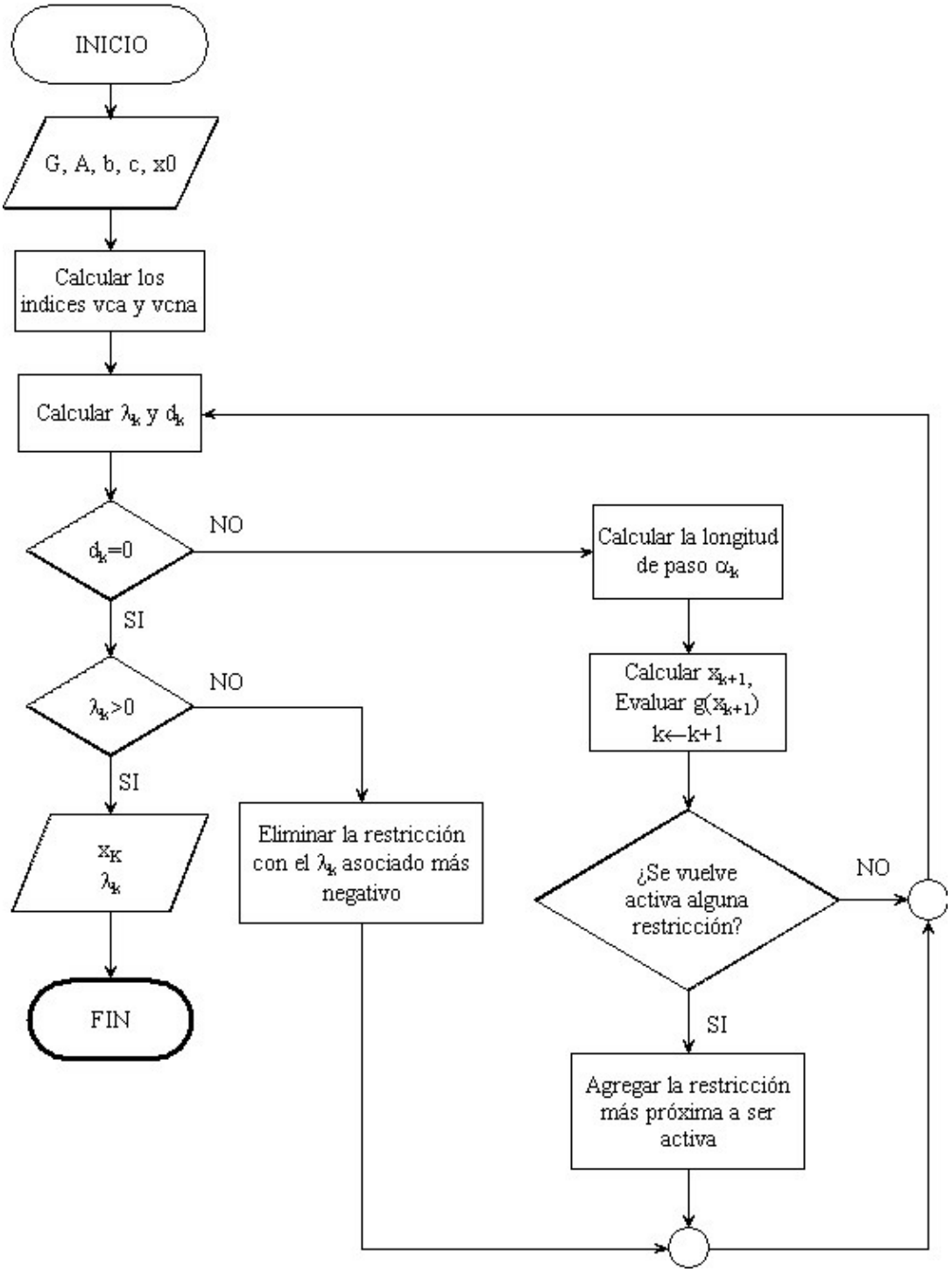


Figura 2-3: Diagrama de flujo del método primal de conjunto activo.

Capítulo 3

Pruebas

Para comprobar el funcionamiento del Algoritmo de Espacio Rango para Programación Cuadrática se utilizaron problemas prueba generados con el "generador de problemas prueba para programación cuadrática"[10]. Este generador produce 2 archivos, en el primero se encuentra la definición del problema donde se especifican: el número de variables, número de restricciones, matriz hessiana, vector de costos, matriz de restricciones, vector de términos independientes y un punto inicial factible. En el segundo archivo se encuentra la solución del problema conteniendo el punto óptimo encontrado y los multiplicadores de Lagrange asociados a este.

El generador de problemas prueba para programación cuadrática genera problemas con las siguientes características: problemas bien condicionados sin degeneración, problemas bien condicionados con degeneración, problemas mal condicionados sin degeneración y problemas mal condicionados con degeneración. La finalidad de probar los cuatro tipos de problemas, es verificar el funcionamiento del algoritmo con problemas que tengan estas características, debido a que cada uno de ellos presentan un mayor o menor grado de complejidad en su manejo. El primer tipo de problemas son los que presentan menor grado de complejidad y se espera que los resultados obtenidos sean los mejores.

El algoritmo de Espacio Rango para Programación Cuadrática se implementó en Matlab ver. 6.5. Se eligió este software debido a que es un software de alto rendimiento para computación matricial [11], especialmente aquellos con formulaciones matriciales y vectoriales, ya

que implementarlos en otro lenguaje tomaría demasiado tiempo. Integra cálculos, visualización y programación en un entorno fácil de usar, donde los problemas y soluciones son expresados en notación matemática familiar. Esta característica es muy importante para la selección del software de desarrollo, debido a que el manejo de matrices es la herramienta principal para la implementación del algoritmo, sobre todo que simplifica la manera de calcular la inversa de una matriz.

3.1. Resultados.

Las pruebas fueron realizadas en una computadora con procesador intel pentium 4 a 2.8 Ghz, con 256 Kb de memoria ram y sistema operativo Windows XP. Los resultados de las pruebas se muestran en las siguientes tablas, donde se especifican: el tiempo que consumió el algoritmo para resolver cada uno de los problemas, el error primal y el error dual norma dos. Las tablas estan divididas en 6 columnas, las primeras tres columnas especifican el número de problema, el número de variables y de restricciones, los errores primal y dual norma dos se encuentran en las columnas cuatro y cinco y el tiempo (en segundos) en la seis. La primera tabla muestra los resultados obtenidos durante las pruebas a los problemas bien condicionados y sin degeneración.

Problema	Variables	Restricciones	Error primal	Error dual	Tiempo
1	10	7	3.04E-14	5.15E-14	0.0160
2	50	30	9.33E-14	1.14E-13	0.0470
3	100	70	2.74E-13	3.16E-13	0.4690
4	150	90	4.73E-13	4.80E-13	1.0460
5	200	100	3.06E-13	2.88E-13	1.8600
6	250	150	1.23E-12	1.13E-12	6.6720
7	300	200	1.06E-12	9.66E-13	22.86
8	800	400	3.28E-12	3.02E-12	870.6870
9	1000	500	9.48E-12	7.36E-12	2489.8
10	1200	700	1.06E-10	9.81E-11	8623.5

Cuadro 3.1: Resultados para problemas bien condicionados y sin degeneración.

La tabla 3.2 presenta los resultados obtenidos durante las pruebas realizadas a los problemas bien condicionados con degeneración.

Problema	Variables	Restricciones	Error primal	Error dual	Tiempo
1	10	7	2.13E-14	4.26E-14	0.0160
2	50	30	5.68E-14	4.97E-14	0.0630
3	100	70	3.15E-13	3.59E-13	0.4060
4	150	90	5.61E-13	4.92E-13	1.2500
5	200	100	5.12E-13	4.73E-13	1.7810
6	250	150	1.66E-12	1.47E-12	7.4060
7	300	200	1.27E-12	1.16E-12	28.9220
8	800	400	6.88E-12	5.46E-12	980.4370
9	1000	500	6.13E-12	5.00E-12	2117.3
10	1200	700	8.38E-11	7.80E-11	7690.7

Cuadro 3.2: Resultados para problemas bien condicionados y con degeneración.

La tabla 3.3 muestra los resultados obtenidos durante las pruebas realizadas a los problemas mal condicionados sin degeneración.

Problema	Variables	Restricciones	Error primal	Error dual	Tiempo
1	10	7	3.96E-12	2.82E-9	0.0150
2	50	30	2.02E-12	5.56E-12	0.0310
3	100	70	2.70E-12	2.27E-11	0.4220
4	150	90	8.53E-13	2.19E-12	0.9840
5	200	100	4.41E-13	1.81E-12	1.7180
6	250	150	8.51E-13	2.06E-12	7.9690
7	300	200	2.10E-12	6.40E-12	28.1090
8	800	400	2.46E-11	4.75E-11	1010.6
9	1000	500	5.026E-11	9.33E-11	2536.6
10	1200	700	1.11E-10	1.96E-10	7484.6

Cuadro 3.3: Resultados para problemas mal condicionados y sin degeneración.

La tabla 3.4 presenta los resultados obtenidos durante las pruebas realizadas a los problemas mal condicionados con degeneración.

Problema	Variables	Restricciones	Error primal	Error dual	Tiempo
1	10	7	9.24E-14	1.35E-12	0.0160
2	50	30	1.23E-13	9.15E-13	0.0470
3	100	70	9.59E-12	3.33E-10	0.4060
4	150	90	1.25E-12	3.70E-12	0.9530
5	200	100	3.41E-13	6.28E-13	1.7340
6	250	150	4.51E-12	3.17E-11	6.4530
7	300	200	1.22E-11	5.72E-11	25.4380
8	800	400	9.92E-11	2.54E-10	1303.5
9	1000	500	8.34E-10	2.14E-9	9674.3
10	1200	700	6.82E-11	1.18E-10	7883.2

Cuadro 3.4: Resultados para problemas bien condicionados y con degeneración.

3.2. Gráficas.

En esta sección se presentan las gráficas que, ayudan a presentar y comparar de una manera más clara los resultados obtenidos. En la primer gráfica se muestran los tiempos consumidos durante las pruebas realizadas a los problemas bien condicionados y sin degeneración.

Como se aprecia en la figura (3-1), para problemas pequeños hasta de 300 variables el tiempo ocupado es pequeño, pero para problemas grandes, a partir de 800, el tiempo ocupado es mucho mayor, creciendo exponencialmente.

En las figuras (3-2 y 3-3) se muestran los errores primal y dual encontrados para los problemas bien condicionados y sin degeneración.

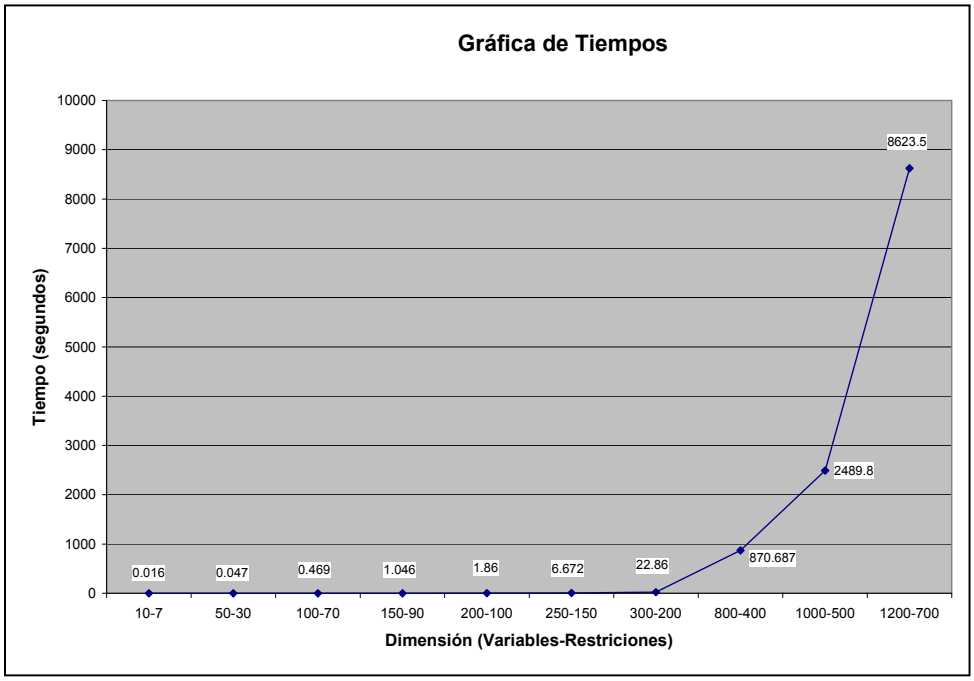


Figura 3-1: Tiempo ocupado por el algoritmo para resolver problemas bien condicionados y sin degeneración.

En estas figuras se puede observar que el error primal y el error dual encontrados no tienen un comportamiento uniforme, aunque si se puede decir que crece conforme aumentan el número de variables y restricciones, pero de manera general podemos darnos cuenta que los errores encontrados, sin importar el número de variables y restricciones, continúa siendo muy pequeño.

La figura 3-4 muestra los resultados del tiempo ocupado por el algoritmo para resolver problemas bien condicionados y con degeneración.

En esta figura se puede apreciar que aunque se esperaba que los resultados fueran peores que para el primer tipo de problemas, el algoritmo utiliza un tiempo muy cercano a los anteriores, incluso tomando menos tiempo para algunas dimensiones del problema.

Las figuras 3-5 y 3-6 muestran el error primal y dual encontrado para los problemas del tipo bien condicionado y con degeneración.

Analizando las figuras, se observa que la solución obtenida por el algoritmo es una solución óptima; al igual que para los problemas anteriores, los errores se comportan de una manera

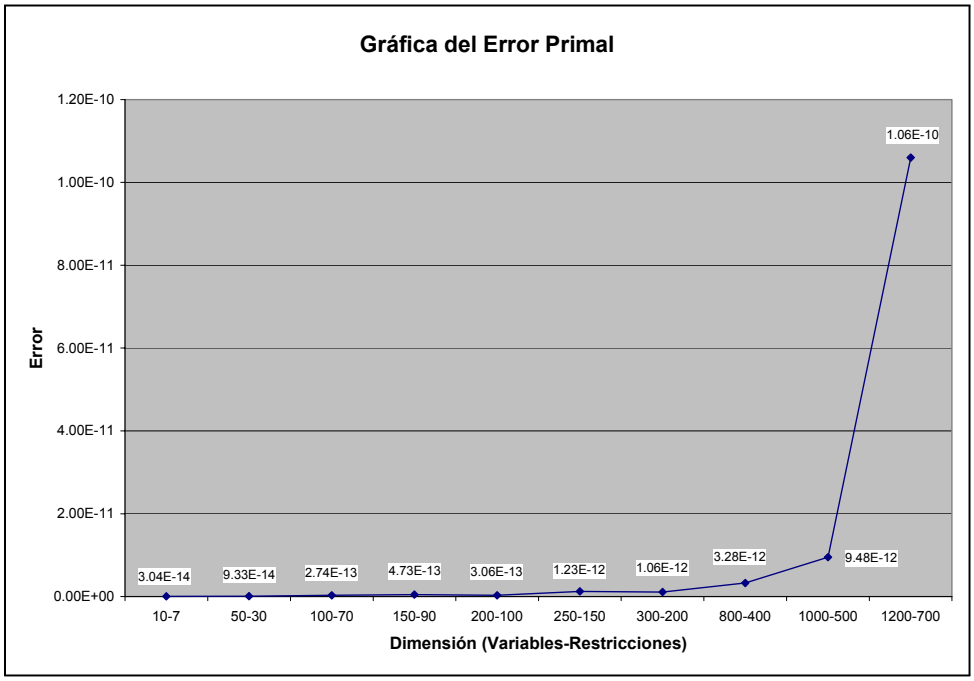


Figura 3-2: Error Primal para problemas bien condicionados y sin degeneración.

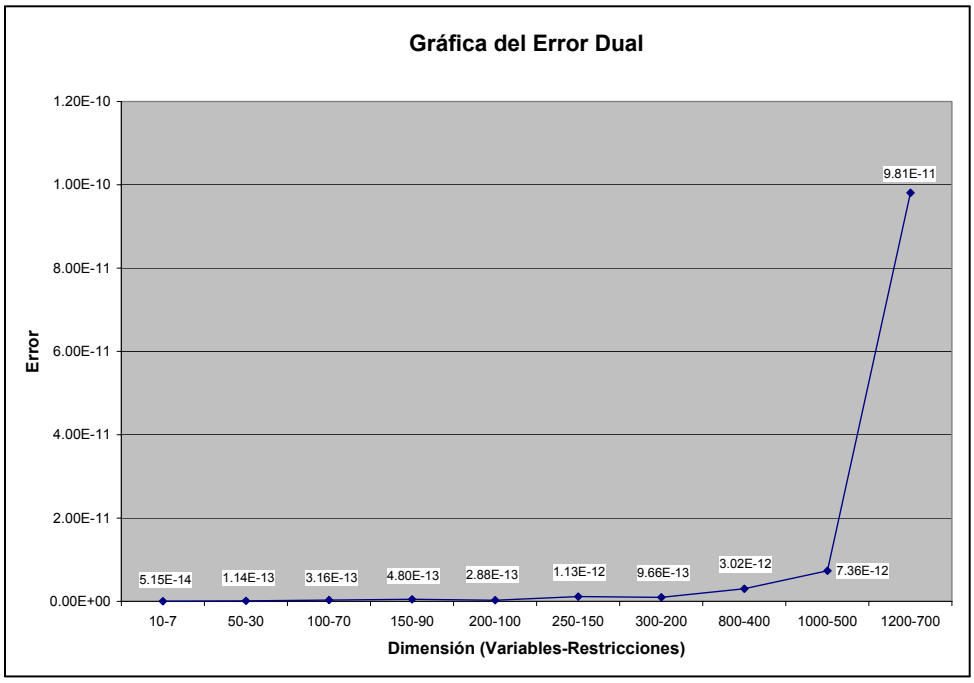


Figura 3-3: Error Dual para problemas bien condicionados y sin degeneración.

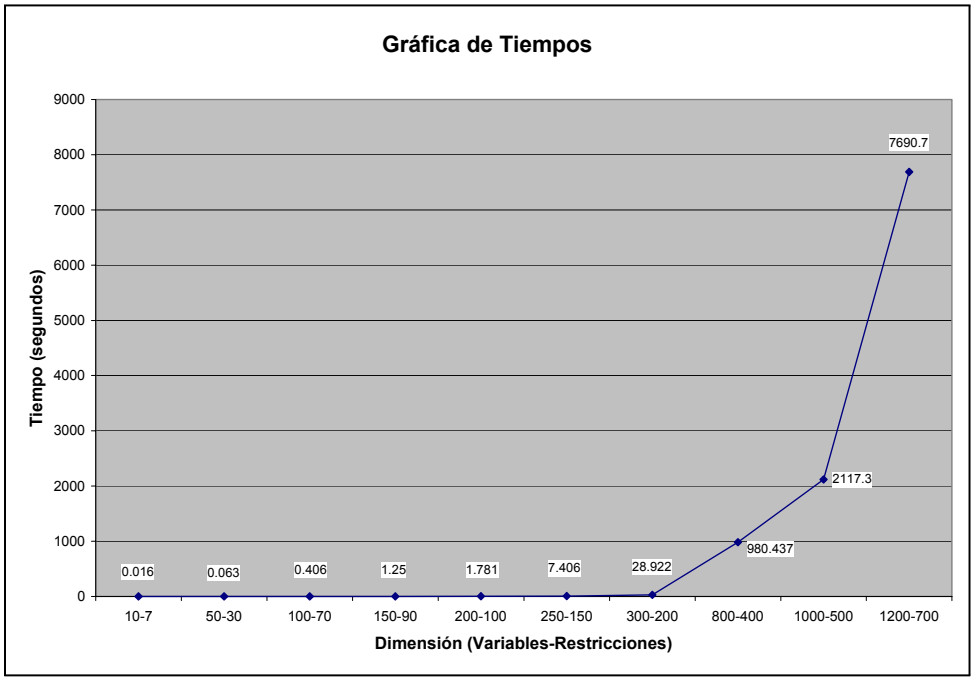


Figura 3-4: Tiempo ocupado por el algoritmo para resolver problemas bien condicionados y con degeneración.

estable, pudiendo decir que tanto para dimensiones grandes y pequeñas, los puntos encontrados son óptimos.

Para los problemas mal condicionados, el "generador de problemas prueba para programación cuadrática", utiliza una variable aleatoria para poder determinar el porcentaje de mal condicionamiento. Debido a esto, existen algunos problemas de este tipo que presentan variaciones significativas en el cálculo del tiempo y los errores dual y primal.

El tiempo consumido por el sistema para resolver problemas mal condicionados y sin degeneración se muestra en la figura 3-7.

El comportamiento mostrado en la figura 3-7 es análogo para las gráficas de tiempo anteriores, el tiempo consumido aumenta exponencialmente con respecto a la dimensión de los problemas, sin embargo, debido a la naturaleza de los problemas, se observa que el tiempo utilizado es levemente mayor a las dos figuras anteriores.

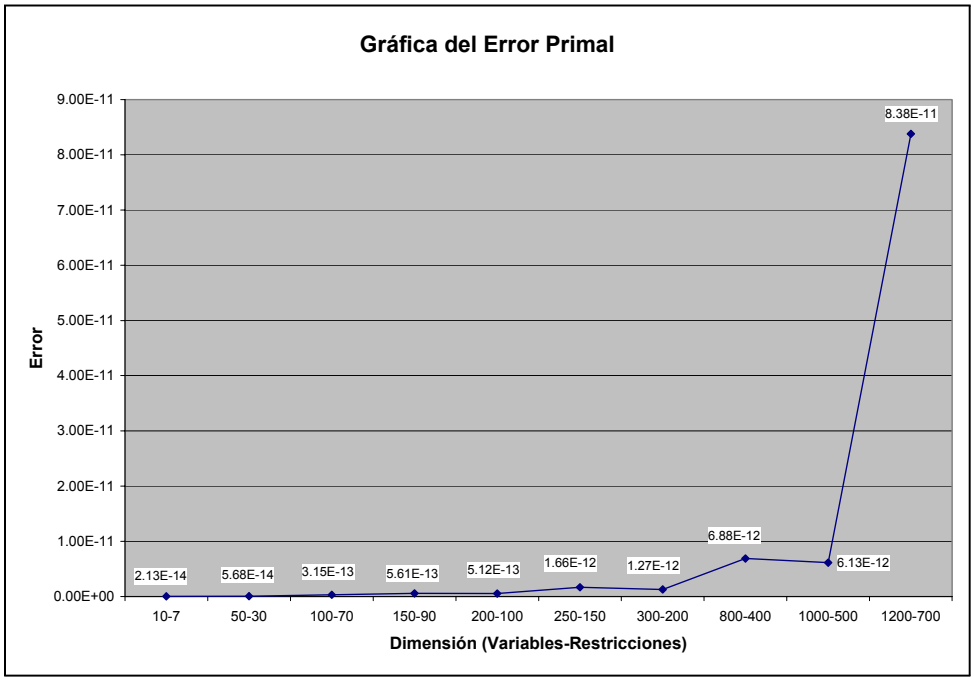


Figura 3-5: Error Primal para problemas bien condicionados y con degeneración.

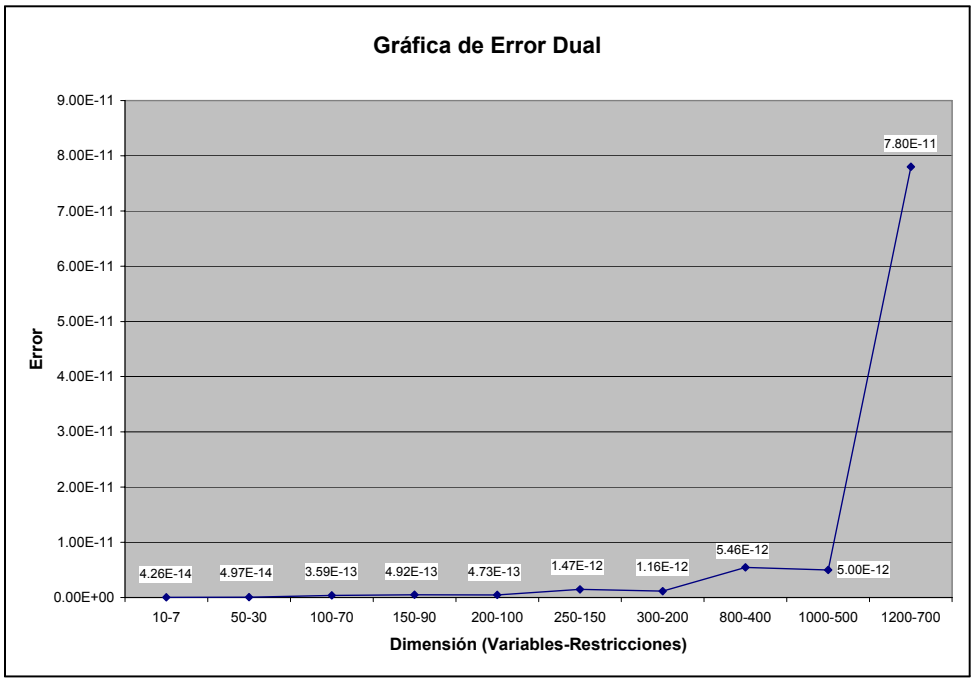


Figura 3-6: Error Dual para problemas bien condicionados y sin degeneración.

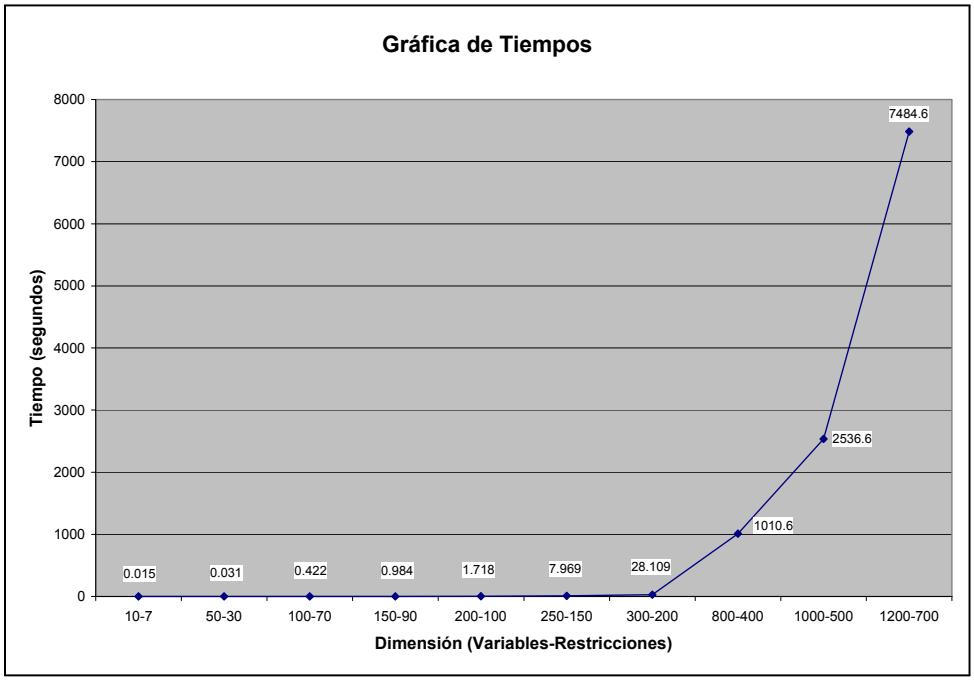


Figura 3-7: Tiempo ocupado por el algoritmo para resolver problemas mal condicionados y sin degeneración.

Para los problemas mal condicionados y sin degeneración, los errores primal y dual son mostrados en las figuras 3-8 y 3-9.

En estas figuras se observa que los errores encontrados, al igual que para los problemas anteriores, es despreciable, por lo que se puede decir que la solución encontrada es óptima. Sin embargo, para el problema de 10 variables y 7 restricciones, se ve que el error dual es más grande incluso que los de mayor dimensión, esto se debe a que el porcentaje de mal condicionamiento para este problema en particular es más grande que el de los demás. Para problemas de igual dimensión generados con un porcentaje menor, el error disminuye, por lo que se decidió presentar el peor de los casos, donde se puede apreciar esta característica.

En la figura 3-10 se observan los tiempos calculados durante las pruebas con problemas mal condicionados y con degeneración.

El comportamiento del sistema, con respecto al tiempo, para problemas mal condicionados y con degeneración, al igual que los mal condicionados y sin degeneración, crece de manera

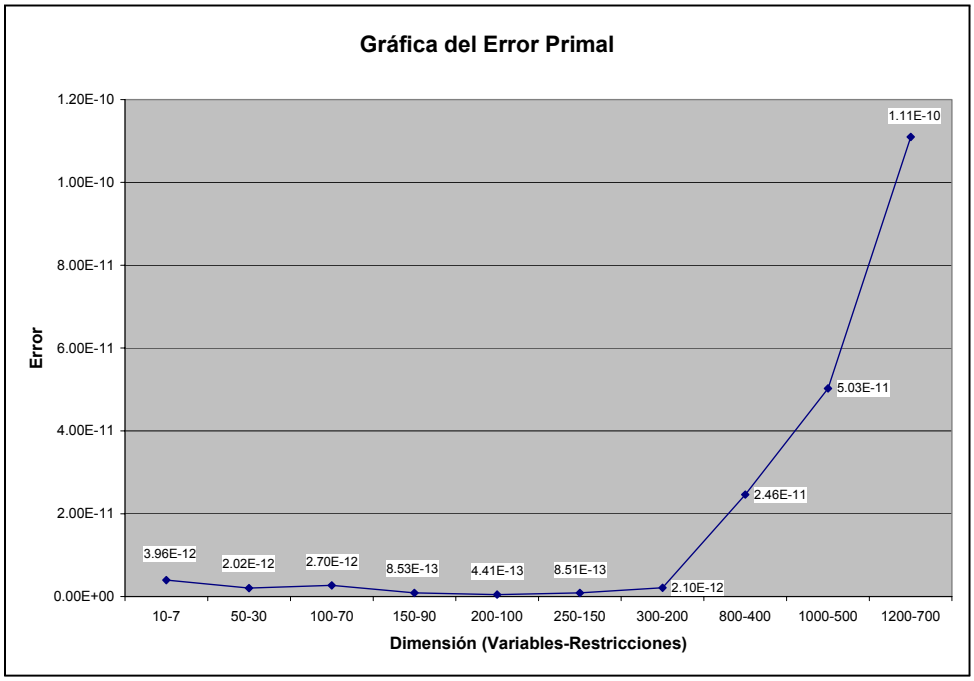


Figura 3-8: Error Primal para problemas mal condicionados y sin degeneración.

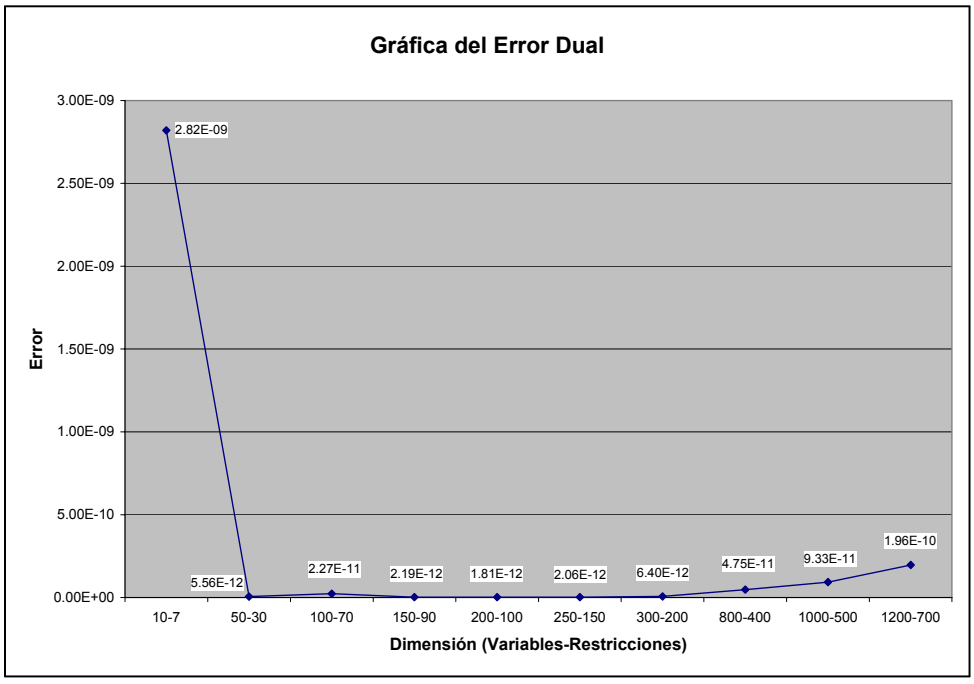


Figura 3-9: Error Dual para problemas mal condicionados y sin degeneración.

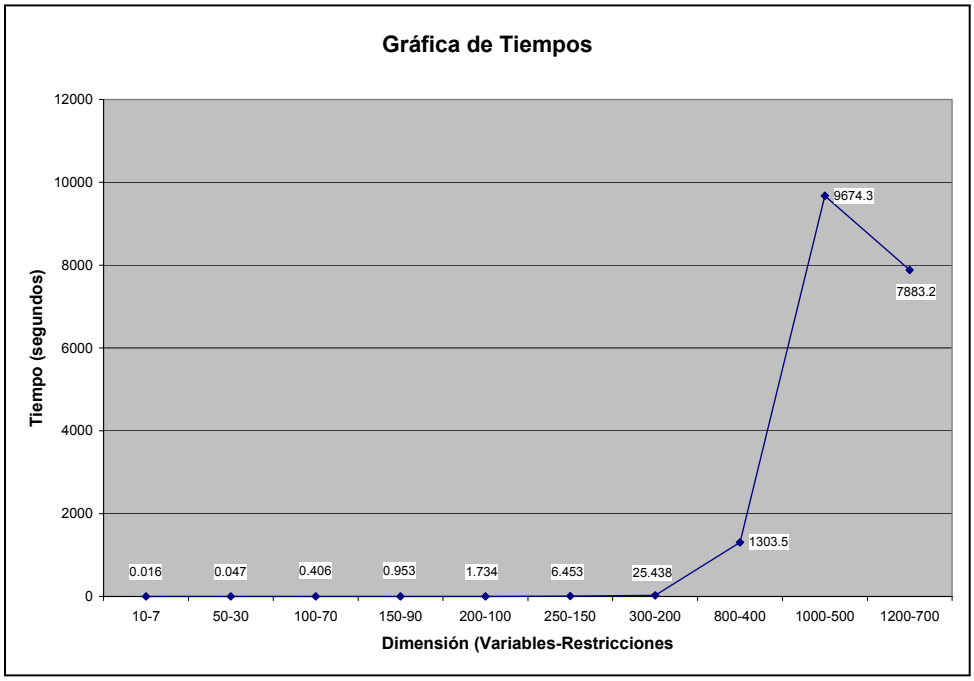


Figura 3-10: Tiempo ocupado por el algoritmo para resolver problemas mal condicionados y con degeneración.

exponencial conforme aumenta la dimensión de los problemas. Se observa, además, que los tiempos consumidos son los peores. Para el caso particular del problema de 1000 variables con 500 restricciones, se ve que el tiempo consumido es el mayor; esto se debe a que el porcentaje de mal condicionamiento es grande y para problemas de la misma dimensión y con porcentaje menor el tiempo consumido disminuye considerablemente.

Los errores primales y duales de los problemas mal condicionados y con degeneración se presentan en las figuras siguientes.

De estas figuras se observa que a pesar de que los errores, al igual que el tiempo, son los más altos, siguen siendo despreciables. Para los problemas de dimensión 100 variables y 70 restricciones y 1000 variables y 500 restricciones, el porcentaje de mal condicionamiento es más grande que para los demás, es por esto que los resultados en el cálculo del error primal y dual varían considerablemente. Para problemas de las mismas dimensiones y con un porcentaje de mal condicionamiento menor, el error calculado disminuye.

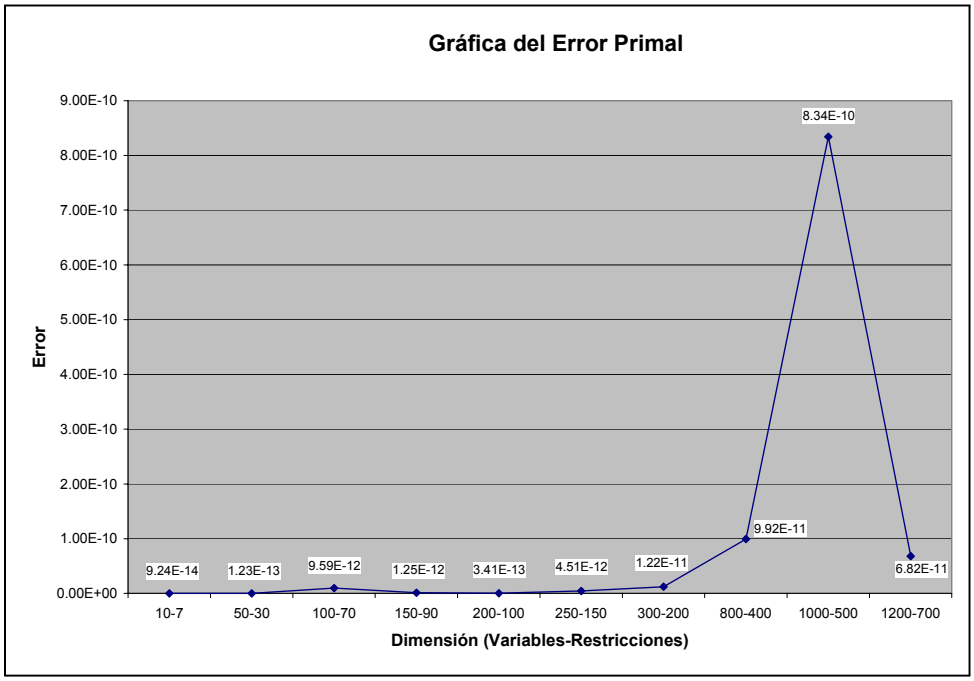


Figura 3-11: Error Primal para problemas mal condicionados y con degeneración.

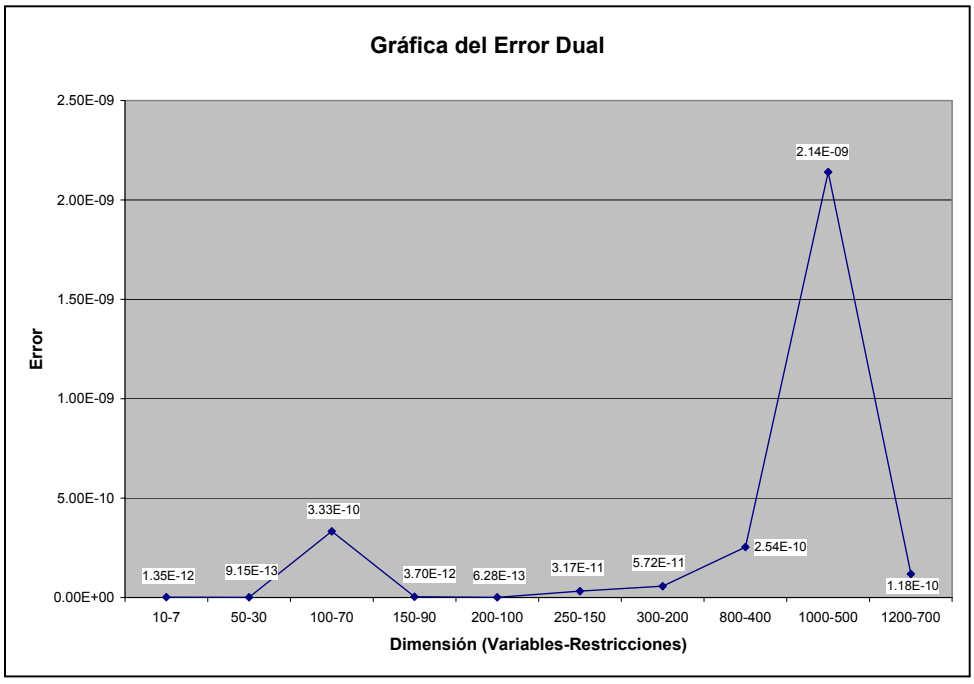


Figura 3-12: Error Dual para problemas mal condicionados y con degeneración.

3.3. AERPC vs NSAQP.

En 2001 dentro de la Universidad Tecnológica de la Mixteca se instrumentó el "Algoritmo de Espacio Nulo para Programación Cuadrática"[1] (NSAQP), dicho algoritmo resuelve problemas del mismo tipo que el "Algoritmo de Espacio Rango para Programación Cuadrática" (AERPC), por lo que es necesario compararlos para poder observar el comportamiento de cada uno de ellos al momento de resolver problemas con las diferentes características de condicionamiento y degeneración. Las siguientes gráficas muestran la comparación de los tiempos consumidos por los algoritmos AERPC y NSAQP.

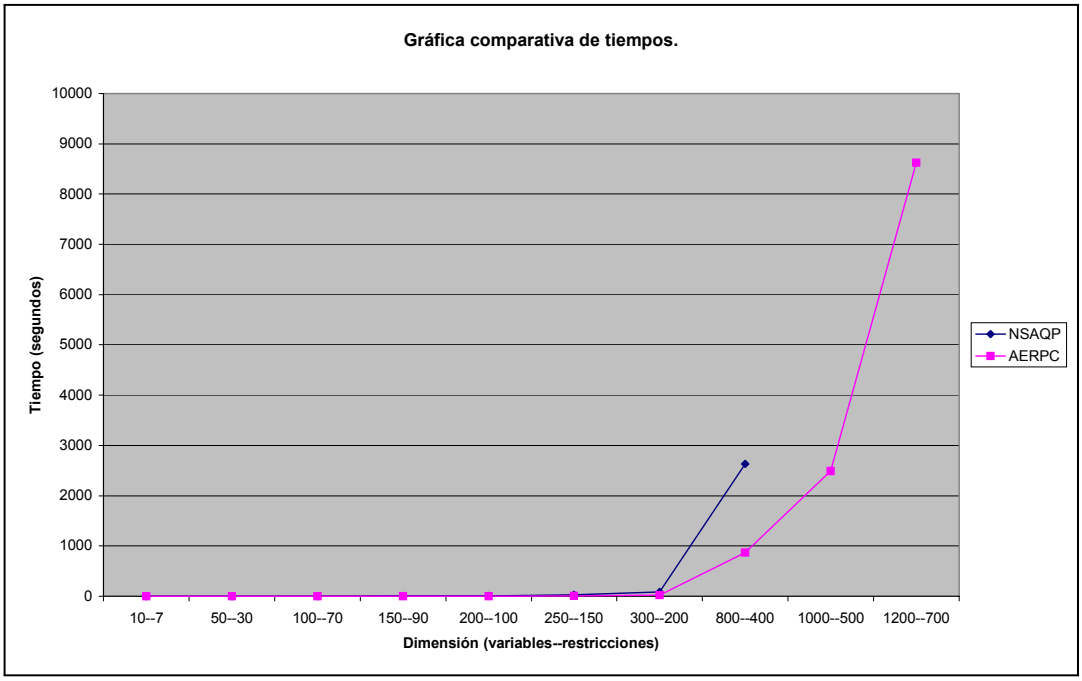


Figura 3-13: Tiempos consumidos para resolver problemas bien condicionados y sin degeneración.

En la figura 3-13 se muestran los tiempos consumidos por los dos algoritmos para resolver problemas bien condicionados y sin degeneración. Se observa que los tiempos consumidos para problemas pequeños (menores a 300 variables) existe poca diferencia entre los dos algoritmos, sin embargo, para problemas mayores el tiempo se va incrementando y los tiempos del AERPC son mucho menores que los del NSAQP. Para problemas grandes (mayores a 800 variables),

el algoritmo NSAQP, en un lapso de 72 horas, no encontró la solución, por lo que se decidió finalizar la búsqueda y el dato no aparece en la gráfica.

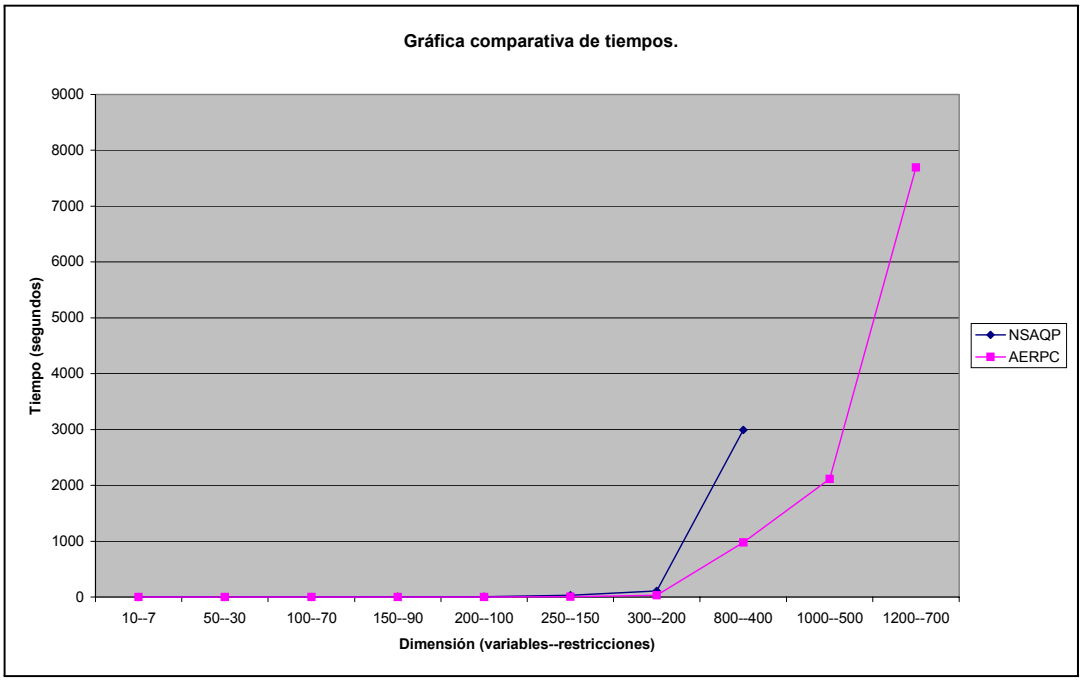


Figura 3-14: Tiempos consumidos para resolver problemas bien condicionados y con degeneración.

En la figura 3-14 se muestran los tiempos consumidos para resolver problemas bien condicionados y con degeneración. Para problemas pequeños el tiempo es similar y para problemas grandes el AERPC se comporta de una mejor manera al consumir los tiempos más bajos. De la misma forma que para los problemas bien condicionados y sin degeneración, no se pudo obtener el tiempo que tarda el NSAQP en resolver este tipo de problemas.

En la figura 3-15 se muestran los tiempos para resolver problemas mal condicionados y sin degeneración, donde nuevamente se observa que el AERPC tiene mejores tiempos de respuesta en problemas grandes. Para problemas mal condicionados y sin degeneracion se observa, nuevamente, que el NSAQP, después de 72 horas de búsqueda, no logró encontrar la solución a los problemas de este tipo, por lo que el dato no aparece en la gráfica.

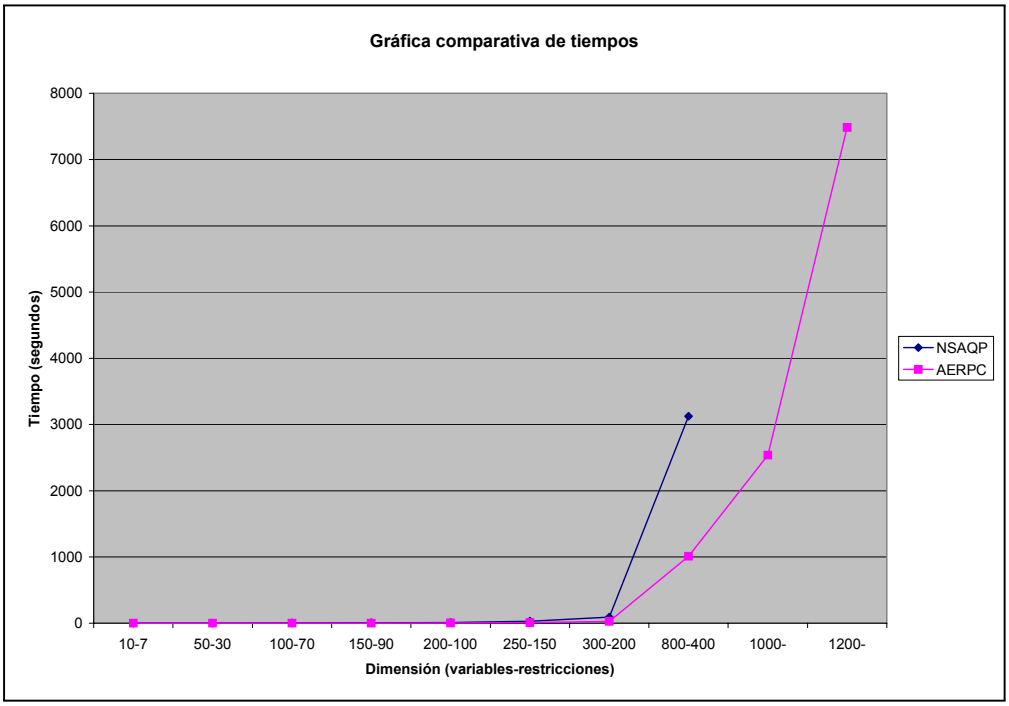


Figura 3-15: Tiempos consumidos para resolver problemas mal condicionados y sin degeneración.

Finalmente, la figura 3-16, para problemas mal condicionados y con degeneración se presenta la misma situación, el AERPC consumió menor tiempo para resolver problemas grandes. La variación que presenta en el tiempo de solución el AERPC para los problemas de 1000 variables, es debido a el porcentaje de mal condicionamiento del problema; se probaron problemas de igual dimensión, pero con un porcentaje de mal condicionamiento menor y los valores disminuyen a medida que disminuye este porcentaje. Además, se observa que el NSAQP, al cabo de 72 horas de búsqueda, no alcanza a encontrar la solución óptima para problemas grandes.

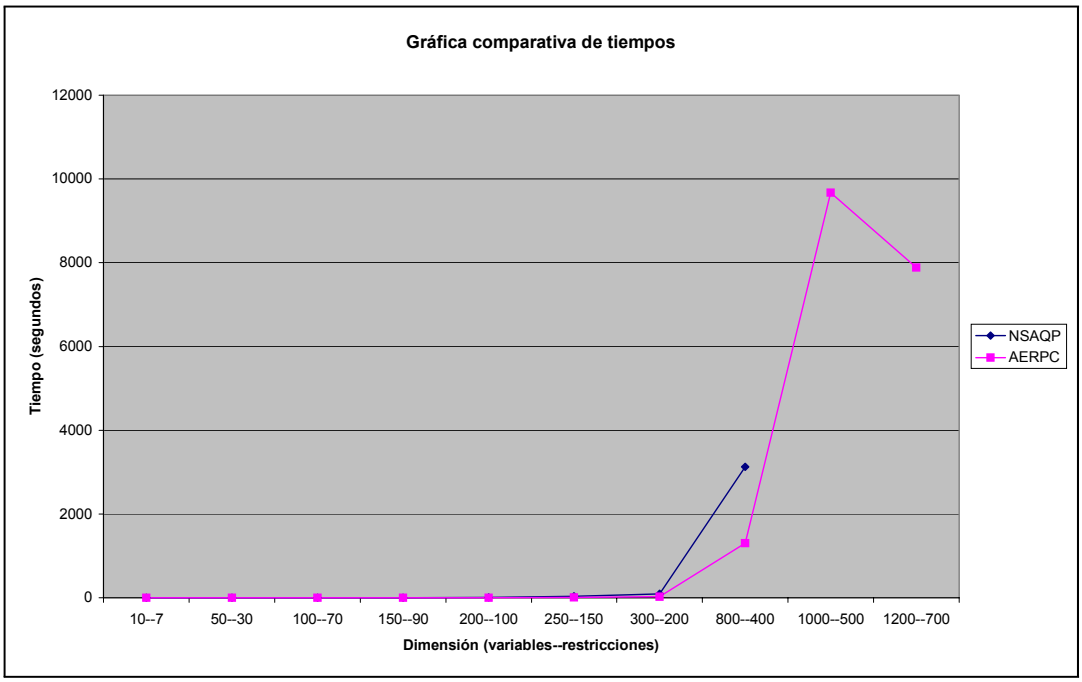


Figura 3-16: Tiempos consumidos para resolver problemas mal condicionados y con degeneración.

Capítulo 4

Conclusiones.

Una vez instrumentado el Algoritmo de Espacio Rango para Programación Cuadrática, y después de realizar pruebas con problemas bien condicionados, mal condicionados, con degeneración y sin degeneración, con dimensiones de 10, y hasta 1200 variables y con 7 y hasta 700 restricciones, se llegó a las siguientes conclusiones:

1. El algoritmo de Espacio Rango Para Programación Cuadrática (AERPC) llegó a la solución óptima en cada problema que fue probado. La dimensión de los errores fue menor a $1.0\ E-9$, pudiendo ser considerado como un buen cero, por lo tanto, se afirma que el AERPC llega a la solución óptima de cada uno de los problemas.

2. Los problemas bien condicionados y sin degeneración presentaron los mejores resultados, tanto en consumo de tiempo como en error.

3. Los problemas mal condicionados (con y sin degeneración) presentaron los peores resultados en consumo de tiempo y se obtuvieron los errores más grandes.

4. El tiempo consumido para resolver los diferentes tipos de problemas crece de manera exponencial, y el error obtenido se mantiene dentro de un rango de valores muy pequeño, ninguna prueba realizada rebasó la dimensión de error de $1.0\ E-9$.

5. Se realizaron pruebas con problemas de hasta 1200 variables y 700 restricciones, obteniendo resultados satisfactorios, por lo que se determina que para problemas de hasta éstas dimensiones, el sistema trabaja eficientemente.

6. Para las pruebas realizadas con problemas mal condicionados, se varió el mal condicionamiento de las matrices, y se obtuvo que para matrices con número de condicionamiento mayor a $1.0E+4$, el sistema no alcanzó a calcular la solución. Para matrices con condicionamiento menor, AERPC llega a la solución, manteniendo las mismas características de tiempo y error que los demás problemas.

7. Para problemas menores a 300 variables los dos sistemas (AERPC y NSAQP), se comportaron de manera similar, con tiempos de respuesta cercanos y llegando a la solución óptima.

8. Para problemas de 300 variables y más, el AERPC presentó mejores tiempos de respuesta que el NSAQP, llegando los dos sistemas a encontrar la solución óptima.

9. El NSAQP con problemas de dimensión mayor a 1000 variables no alcanzó a llegar a la solución óptima.

10. Se concluye entonces que el AERPC es un algoritmo eficaz, con mejores tiempos de respuesta que el NSAQP, calculando la solución óptima para problemas bien condicionados, mal condicionados, con degeneración y sin degeneración, con la función objetivo convexa, restricciones de desigualdad del tipo mayor o igual y con la matriz Hessiana definida positiva.

Apéndice A

Ejemplo gráfico.

En este apéndice se presenta la solución de un problema de programación cuadrática sujeto a restricciones lineales de desigualdad, utilizando el algoritmo de espacio rango para programación cuadrática.

El problema utilizado es un problema pequeño: solo dos variables de decisión, por lo que su utilización será solo ilustrativa, ya que este tipo de problemas tiene poca aplicación.

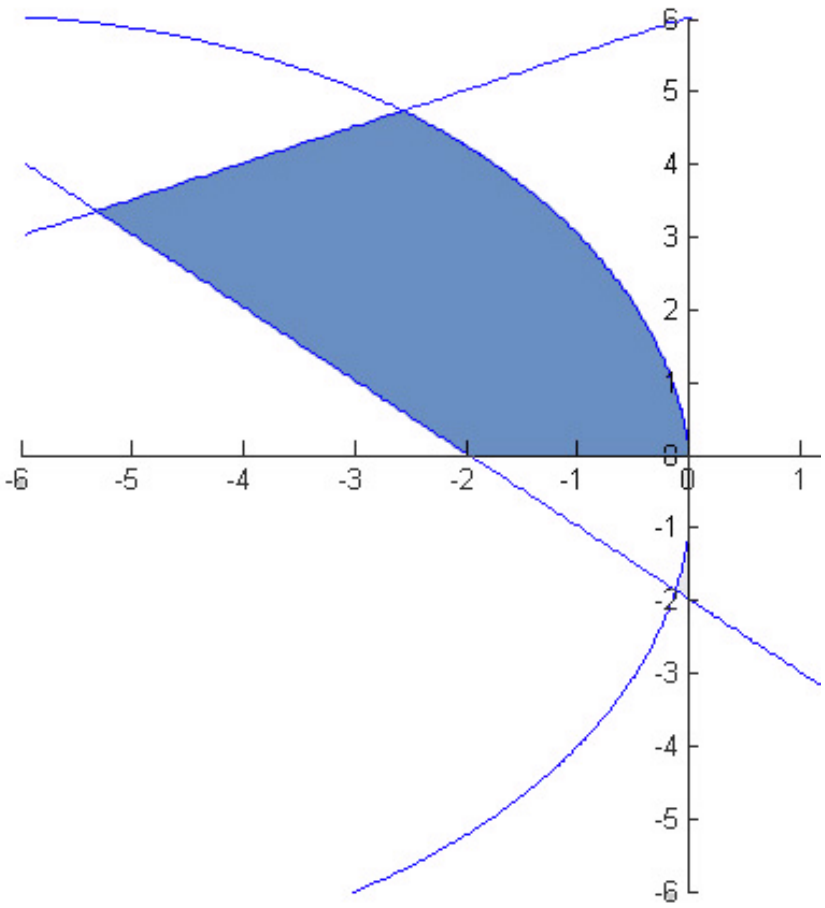
Dado el siguiente problema:

$$\begin{aligned} \text{mín}\,. f(x) &= x_1^2 + x_2^2 + 13x_1 + x_2 \\ s.a. \qquad x_1 + x_2 &\geq -2 \\ x_1 - 2x_2 &\geq -12 \\ -x_1, x_2 &\geq 0. \end{aligned}$$

La representación matricial de sus componentes es:

$$Q = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}, \qquad c = \begin{pmatrix} 13 \\ 1 \end{pmatrix}, \qquad A = \begin{pmatrix} 1 & 1 \\ 1 & -2 \\ -1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad b = \begin{pmatrix} -2 \\ -12 \\ 0 \\ 0 \end{pmatrix}.$$

Gráficamente el problema se ve de la siguiente manera:

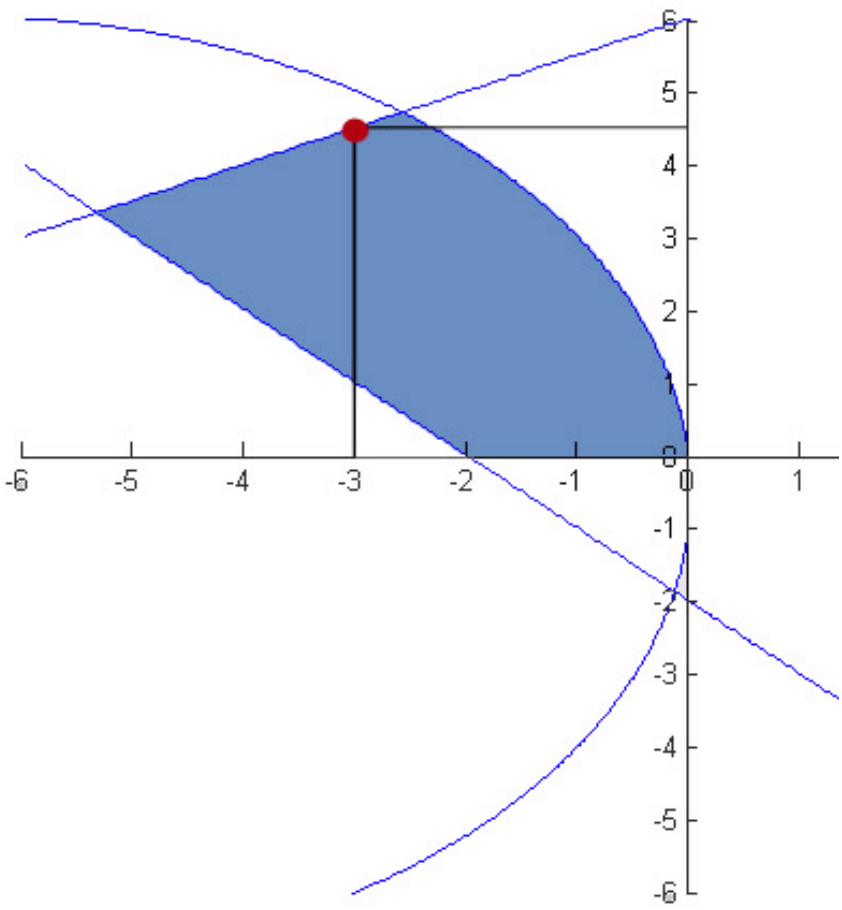


El área sombreada muestra la región factible del problema, determinada por las restriciones a las que se encuentra sujeta.

Antes de iniciar con la solución del problema, es necesario determinar un punto que sea factible, es decir, que se encuentre dentro del la región factible. Uno de estos puntos para el problema es:

$$x_0 = \begin{pmatrix} -3 \\ 4.5 \end{pmatrix}.$$

Graficando este punto factible obtenemos:



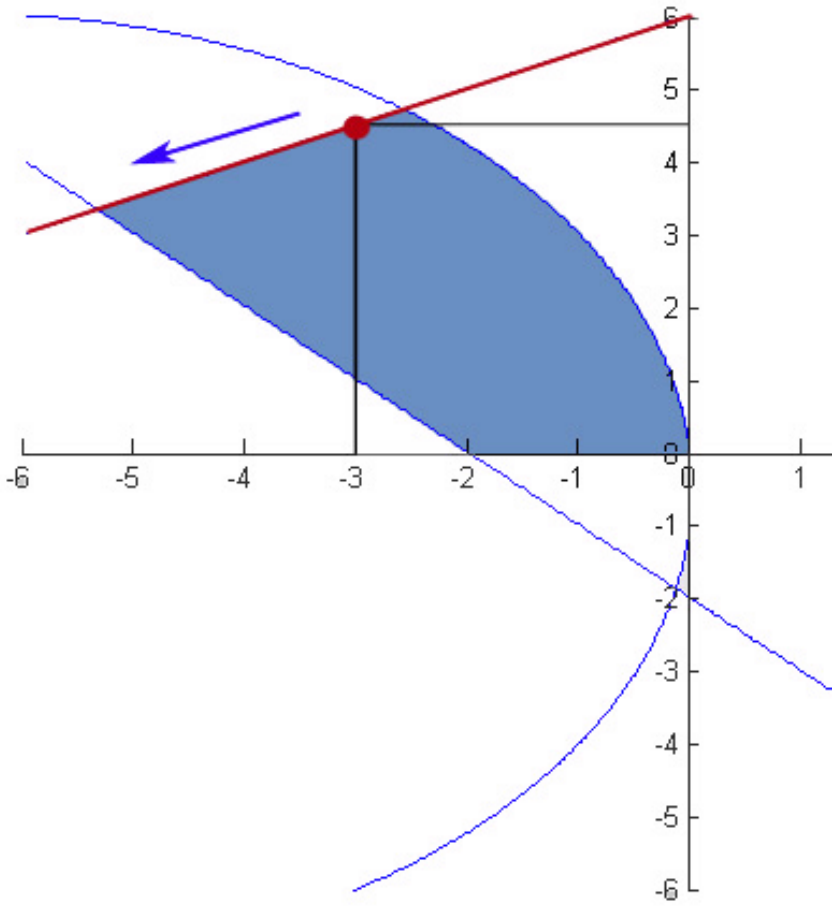
Iteración 1:

Se calcula el conjunto activo, las restricciones que evaluadas en el punto factible se cumplen en la igualdad, obteniendo que únicamente la restricción 2 se encuentra dentro de este conjunto, siendo esta añadida al conjunto de trabajo. Enseguida se evalúa el vector gradiente ($g(x_0)$), se calcula el vector de búsqueda (d_1) y los multiplicadores de Lagrage (λ_1) asociados a las restricciones:

$$d_1 = \begin{pmatrix} \text{-4.8} \\ \text{-2.4} \end{pmatrix},$$

$$\lambda_1 = \begin{pmatrix} 0 \\ \text{-2.56} \\ 0 \\ 0 \end{pmatrix}.$$

Debido a la naturaleza del vector de búsqueda ($d_1 \neq 0$), se determina que existe un punto factible mejor que el que se tiene, moviéndose hacia los puntos con abscisa negativa. Graficamente se ve:

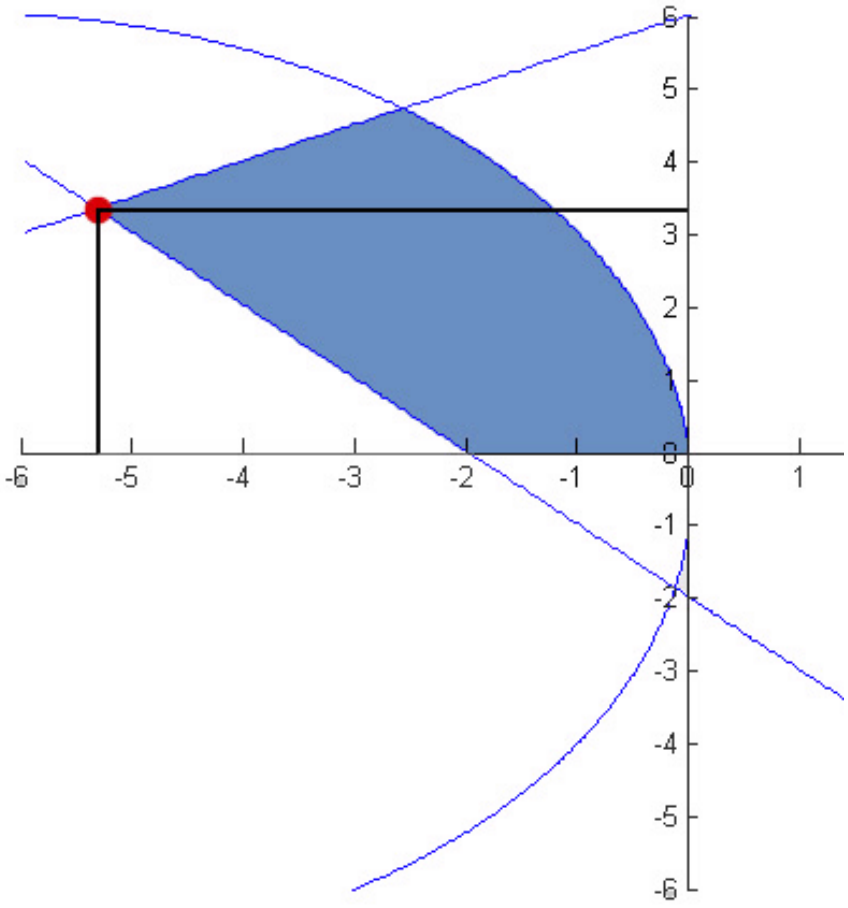


Se calcula la longitud de paso y se obtiene un nuevo valor para x:

$$\alpha_1 = 0.4861,$$

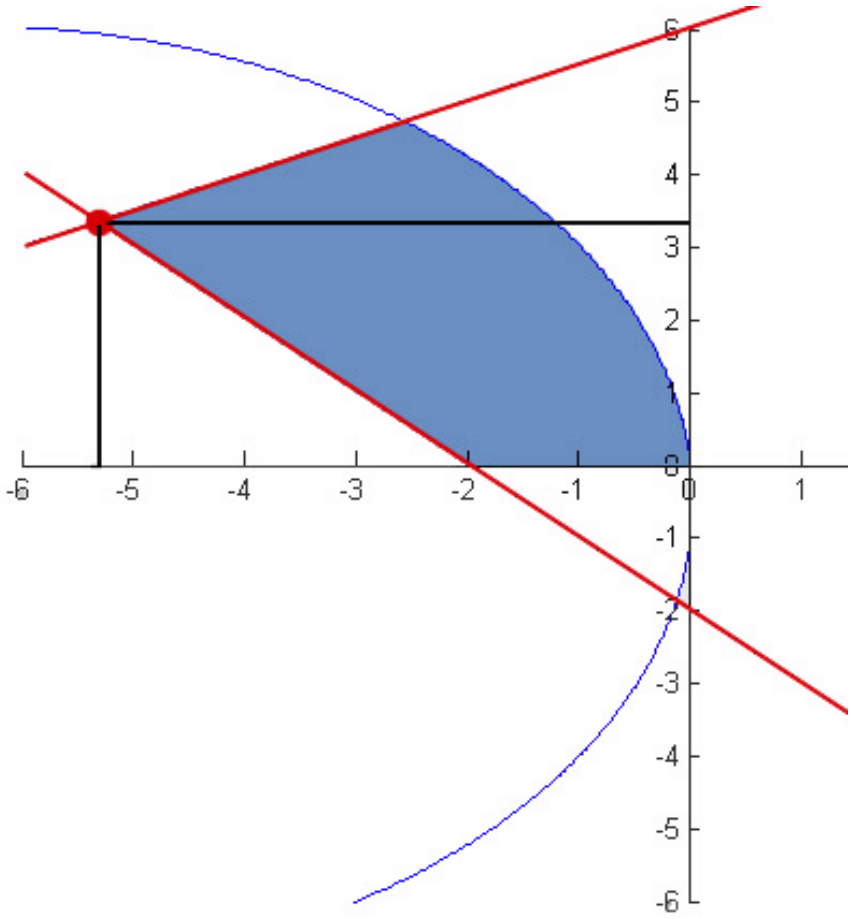
$$x_1 = \begin{pmatrix} -5.3333 \\ 3.3333 \end{pmatrix}.$$

La gráfica es:



Iteración 2:

Con el nuevo punto factible hay que verificar si cumple las condiciones de KKT para saber si es el óptimo, por lo tanto, es necesario calcular el nuevo conjunto activo. Para este punto las restricciones activas son la 1 y la 2.



Se calcula la nueva dirección de búsqueda (d_2) y los multiplicadores de Lagrange asociados a las restricciones activas (λ_2) :

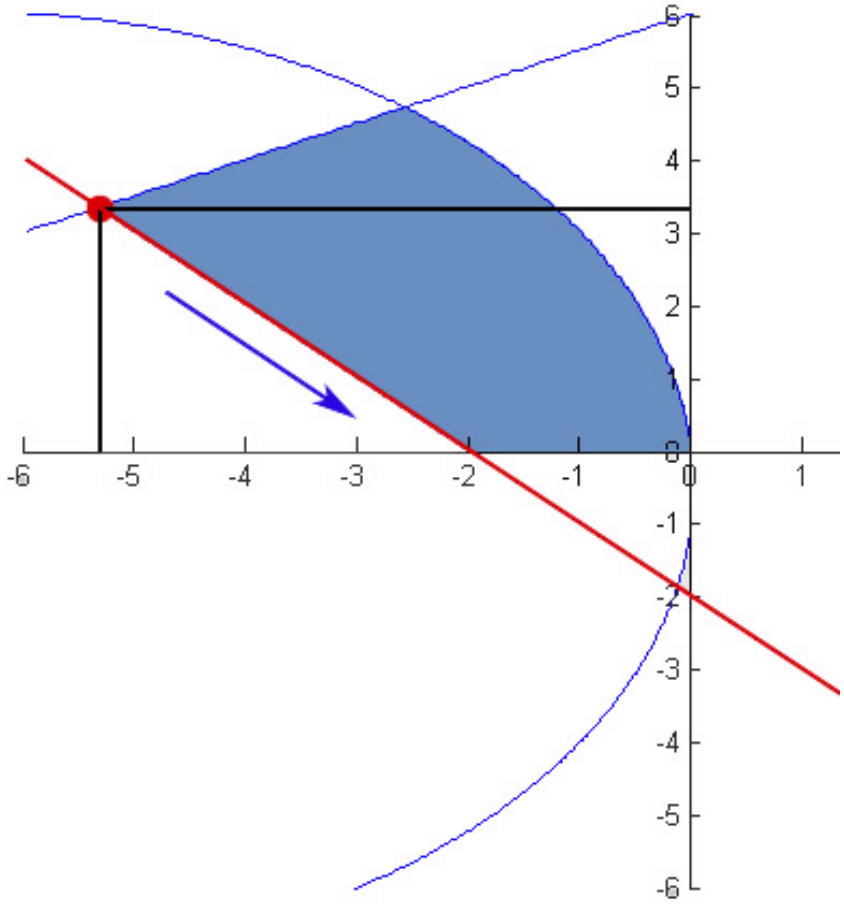
$$\lambda_2 = \begin{pmatrix} 4.1111 \\ -1.7778 \\ 0 \\ 0 \end{pmatrix},$$

$$d_2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

de acuerdo con el valor calculado para (d_2) se determina que el punto factible actual es probable que sea el óptimo, por lo que se verifica que cumpla con las condiciones de KKT, observando que por ser negativo uno de los multiplicadores, no cumple dichas condiciones y es necesario

realizar un paso parcial, para eliminar la restricción que tenga asociado al multiplicador más negativo y calcular nuevamente la dirección de búsqueda.

La restricción a eliminar es la 2, por lo que el conjunto de trabajo queda, únicamente, con la restricción 1 y la dirección de búsqueda indica la direccion hacia los positivos en $x1$ y los negativos en $x2$. Graficamente tenemos:



Iteración 3:

Se calcula la nueva direcciòn de búsqueda y los nuevos multiplicadores:

$$\lambda_3 = \begin{pmatrix} 5 \\ 0 \\ 0 \\ 0 \end{pmatrix},$$

$$d_3 = \begin{pmatrix} 1.3333 \\ \text{-}1.3333 \end{pmatrix},$$

por la naturaleza del vector de búsqueda, se determina que existe un mejor punto factible, por lo que se calcula la longitud de paso y el nuevo punto factible:

$$\alpha_2 = 1,$$
$$x_2 = \begin{pmatrix} -4 \\ 2 \end{pmatrix}.$$

Iteración 4:

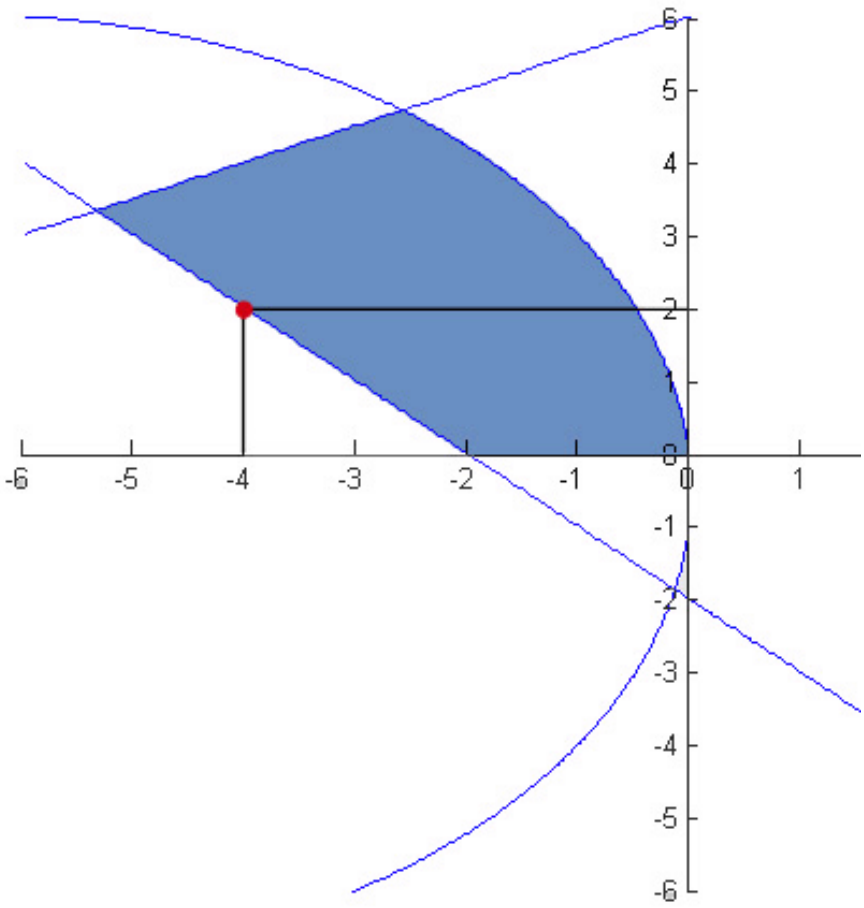
Se continúa con el cálculo de la dirección de búsqueda y los multiplicadores asociados, obteniendo:

$$\lambda_4 = \begin{pmatrix} 5 \\ 0 \\ 0 \\ 0 \end{pmatrix},$$
$$d_4 = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

con la dirección de búsqueda ($d_4 = 0$), se verifica que el punto actual cumpla las condiciones de KKT, siendo los multiplicadores de Lagrange ($\lambda_4 \geq 0$), se determina, por lo tanto, que el punto actual es el punto óptimo, quedando la solución:

$$\lambda^* = \begin{pmatrix} 5 \\ 0 \\ 0 \\ 0 \end{pmatrix},$$
$$x^* = \begin{pmatrix} -4 \\ 2 \end{pmatrix}.$$

Gráficamente la solución se ve de la siguiente manera:



Apéndice B

Manual de usuario.

Para iniciar AERPC es necesario iniciar la aplicación MATLAB, dando doble clic sobre el icono  que se encuentra sobre el escritorio o seleccionándolo desde la lista de programas instalados en la computadora. Este acceso es creado cuando se instala el programa usando el instalador de MATLAB en el directorio de instalación.

Una vez iniciado este programa hay que ubicarnos en el directorio de trabajo de AERPC. Esto lo podemos hacer siguiendo los pasos:

1. Desde la línea de comandos del programa escribir: cd c:\ruta (Ej. cd c:\aerpc), presionar enter, teclear el nombre del programa (aerpc) y presionar enter.
2. Utilizando el current directory browser de MATLAB, buscar el archivo aerpc.m y abrirlo.

Esta acción abrirá la ventana del editor de MATLAB, desde donde se puede visualizar el código del sistema.

Para ejecutar el programa se puede realizar de tres maneras diferentes, que son las siguientes:

1. Presionar una vez la tecla F5, ubicada en la parte superior del teclado.
2. Dar un clic sobre el icono  que se encuentra en la barra de herramientas del editor de MATLAB.

3. Dentro del menú Debug de MATLAB, seleccionar la opción Run.

Al ejecutar el programa, se abrirá la interfaz gráfica (figura B-1) y se podrá empezar a utilizar el sistema. La interfaz contiene los menúes Archivo, Teclado y Ayuda.

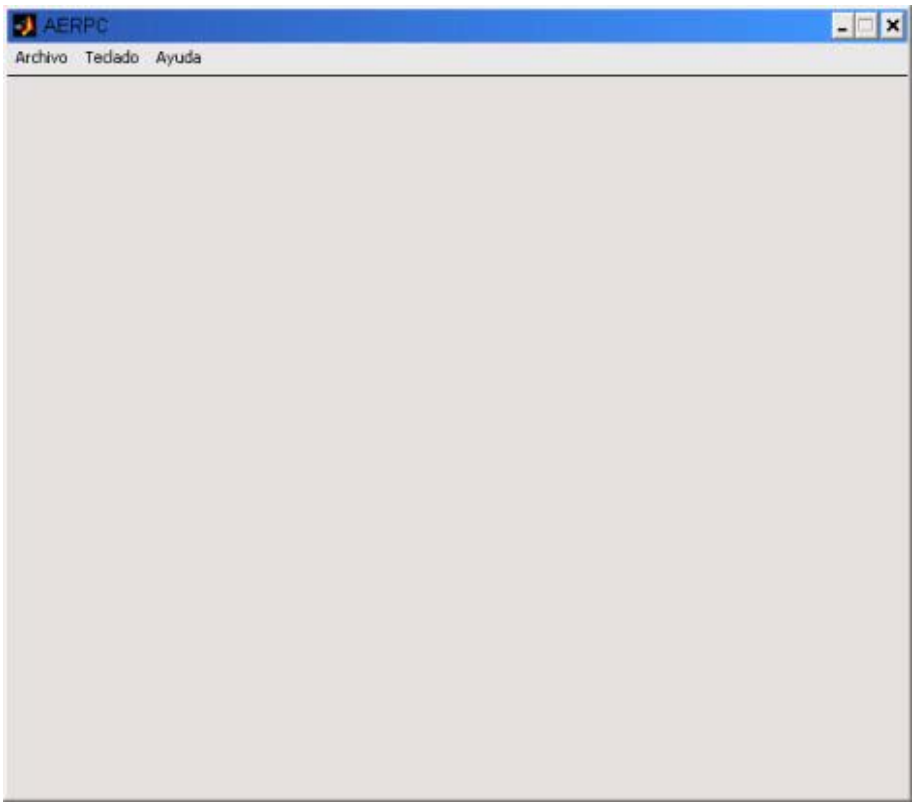


Figura B-1: Pantalla principal

Dentro del menú Archivo, se pueden abrir archivos que contengan definiciones de problemas con la especificación adecuada, resolverlos, guardar los resultados y salir del programa.

El menú Teclado contiene las opciones que permiten crear ejemplos desde el teclado, resolverlos y mostrar la solución obtenida. Se recomienda utilizar esta opción cuando los problemas son pequeños, por el espacio requerido para desplegar los resultados.

Y por último, el menú Ayuda, que permite obtener acceso a esta área, donde se encuentra la información sobre el manejo del sistema.

B.1. Abrir un archivo.

Para abrir un archivo se debe seleccionar la opción Abrir del menú Archivo como se muestra en la figura B-2. Una vez seleccionada esta opción, se abre un cuadro de diálogo que permite moverse dentro de las carpetas de Windows y seleccionar el archivo deseado (figura B-3).

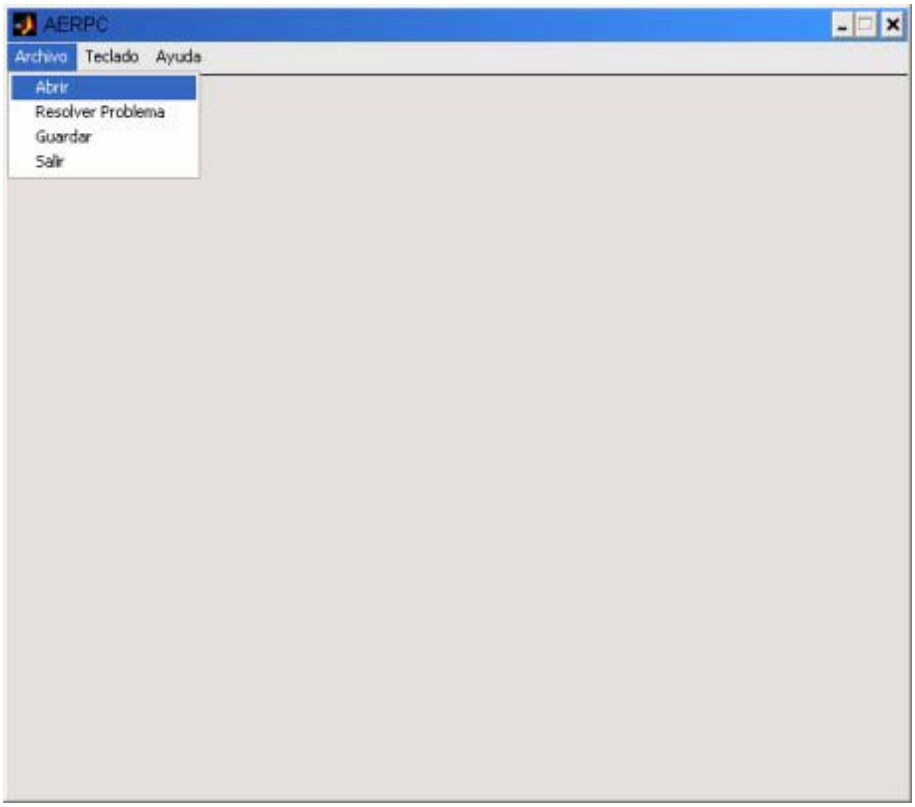


Figura B-2: Abrir archivo

Una vez seleccionado el archivo se da un clic en el botón abrir, lo que permite seleccionar el archivo que va a ser resuelto por el sistema. La ventana se cierra y se leen los datos. En caso de que el problema se haya cargado correctamente se muestra el cuadro de dialogo de la figura B-4; si existiera algún error, en los datos del problema, se muestra un mensaje indicándolo (figura B-5), y será necesario revisar los datos contenidos en el archivo y proceder nuevamente a cargarlo. Una vez cargado el problema, el siguiente paso es resolverlo.

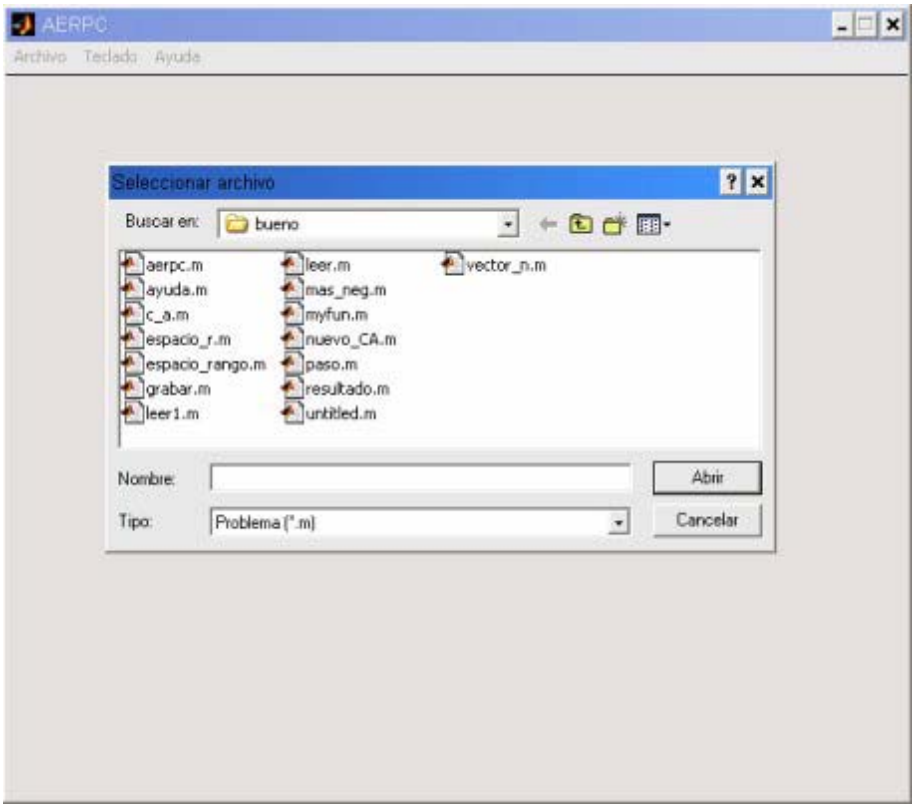


Figura B-3: Seleccionar archivo

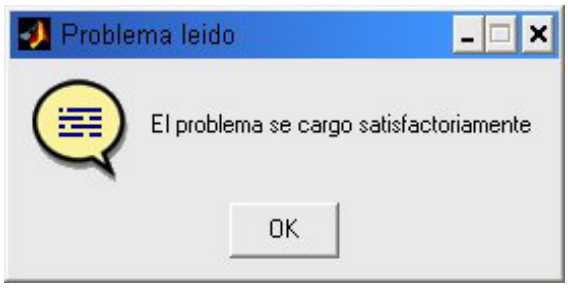


Figura B-4: Problema cargado correctamente

Figura B-5: Error al leer el archivo

B.2. Resolver el problema.

Una vez que se ha seleccionado el archivo, al escoger la opción "Resolver problema" del menú Archivo, se procede a darle solución al problema. Al finalizar se muestra el mensaje de la figura B-6, que indica que se resolvió satisfactoriamente y la solución encontrada puede ser almacenada en un archivo. En caso de que surgiera algún error durante el proceso de solución del problema será indicado por el mensaje de la figura B-7, y será necesario verificar que el archivo contenga la definición de un problema que cumpla con las especificaciones necesarias (ver Apéncide C). Una vez corregido es necesario realizar el procedimiento de selección de archivo, para que se puedan leer los datos.

Figura B-6: Problema resuelto satisfactoriamente.

En caso de intentar resolver un problema que no ha sido cargado aparecerá una ventana indicando este error (Figura B-8).

Figura B-7: Error al resolver el problema

Figura B-8: Error, falta cargar un problema

En caso de intentar resolver un problema por segunda vez consecutiva, el sistema muestra el mensaje de error de la figura B-9, indicando que el problema ya ha sido resuelto anteriormente y lo siguiente es almacenar los resultados en un archivo.

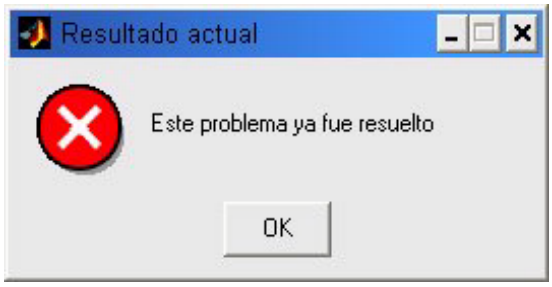


Figura B-9: Error, el resultado ya esta en memoria

B.3. Guardar resultado.

Una vez que se ha cargado desde un archivo y resuelto un problema, para poder visualizar el resultado, es necesario guardarlo en un archivo. Para hacer esto, se debe seleccionar la opción "Guardar" del menú Archivo, como se mestra en la figura B-10.

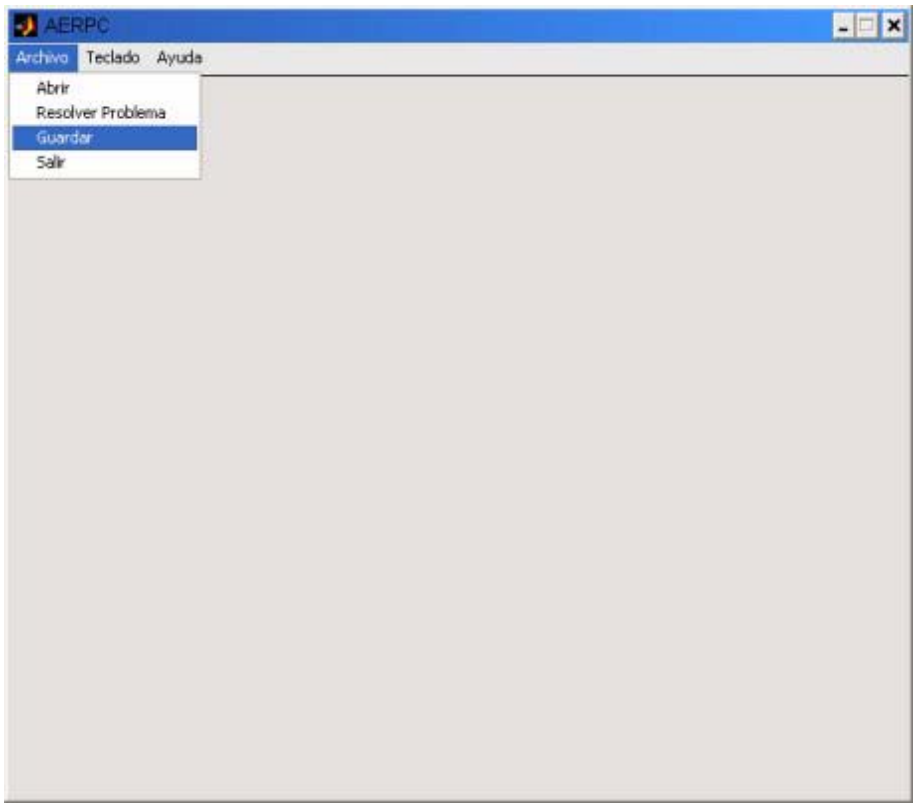


Figura B-10: Guardar resultado

Con esta opción, se abrirá el cuadro de diálogo, de la figura B-11, que permite seleccionar la carpeta donde se guardará el archivo. Se le asigna un nombre y se le agrega (recomendado) la extensión .m para que pueda visualizarse con el editor de Matlab. El formato con el que lo guarda está definido más adelante (Apéndice C).

En caso de no haber resuelto antes un problema, el sistema mostrará el mensaje de error de la figura B-12. Para poder guardar un resultado, primero hay que resolver un problema cargado

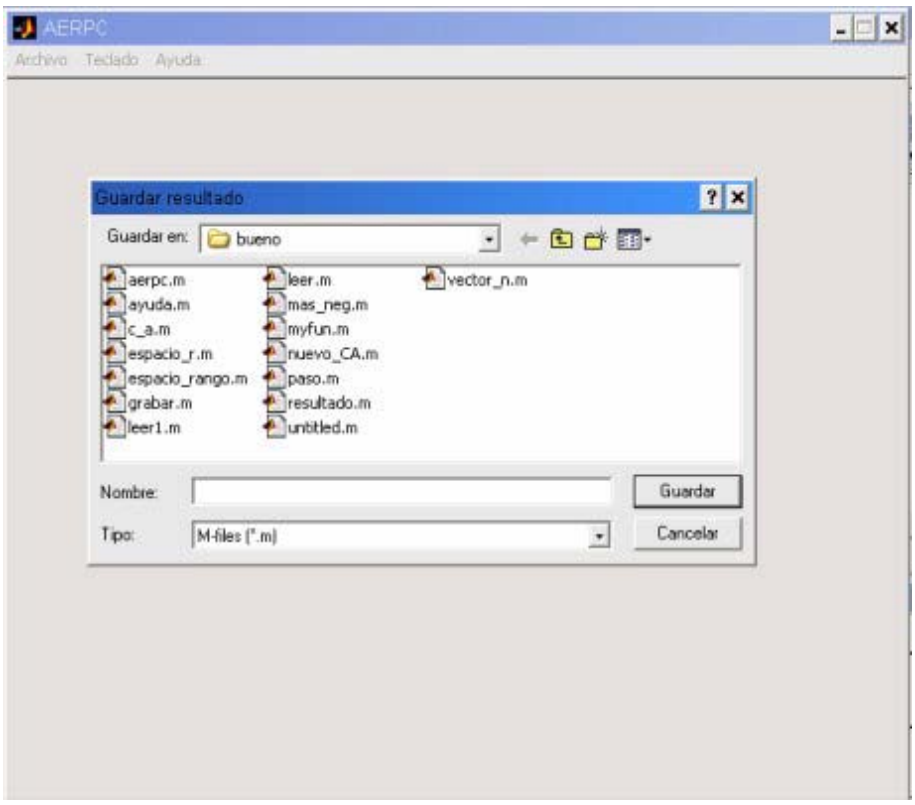


Figura B-11: Seleccionar ruta y nombre del archivo a guardar

desde archivo.

Figura B-12: Error, no hay resultados que guardar

B.4. Crear problema desde teclado.

Otra forma de cargar problemas para resolverlos es introduciendo cada una de las variables desde el teclado, la opción "Leer desde teclado" está dentro del menú teclado como se muestra en la figura B-13. Una vez seleccionada esta opción se mostrarán los cuadros de texto que le permitirán introducir cada una de las variables con el formato de matriz y vector de Matlab (figura B-14).

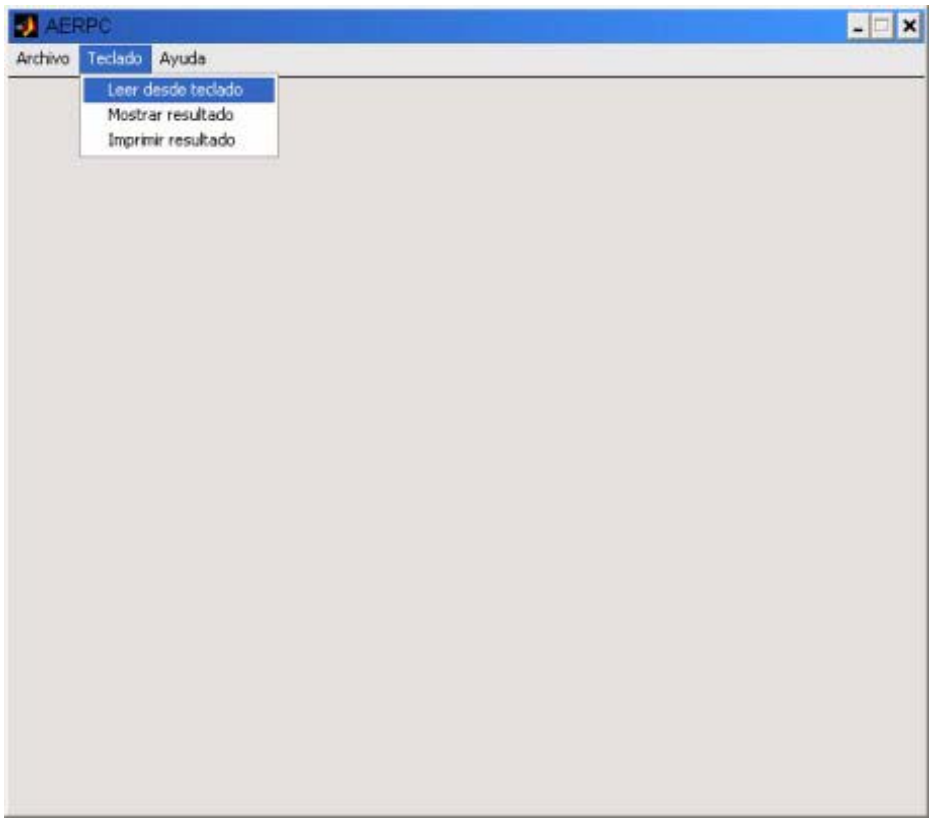


Figura B-13: Crear un problema desde teclado

Para resolver el problema, es necesario introducir todos los datos respetando las características de cada variable. Al presionar el botón "Resolver" el sistema intenta resolver el problema y si encuentra la solución óptima muestra el mensaje de la figura B-15.

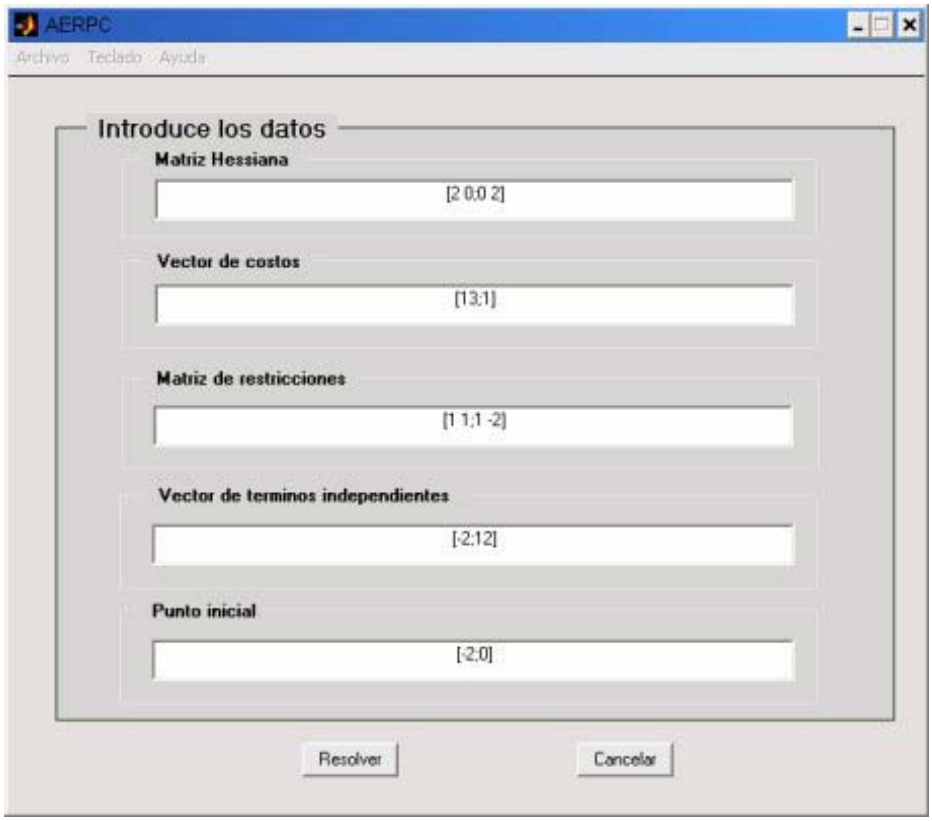


Figura B-14: Introducir datos desde teclado

Figura B-15: Problema resuelto correctamente.

Una vez resuelto el problema se puede visualizar en la pantalla o imprimir. Si llegara a faltar algún dato por introducir y se presiona el botón "Resolver" aparece el mensaje de la figura B-16. Si al intentar resolver el problema aparece el mensaje de la figura B-17, se deberá revisar cada uno de los datos, pues es posible que alguna variable no cumpla las características de los problemas a resolver (Apéndice C).

Figura B-16: Error de datos, falta información

Figura B-17: Error en los datos

Si desea cancelar la operación, el sistema le informará que ningún problema fue cargado ni resuelto, por lo que no se podrá imprimir o mostrar los resultados (figura B-18).

B.5. Mostrar resultado.

Después de resolver un problema que fue introducido por el teclado, se puede mostrar el resultado en la pantalla. La opción "Mostrar resultado" se encuentra en el menú Teclado (figura B-19).

Figura B-18: Operación cancelada

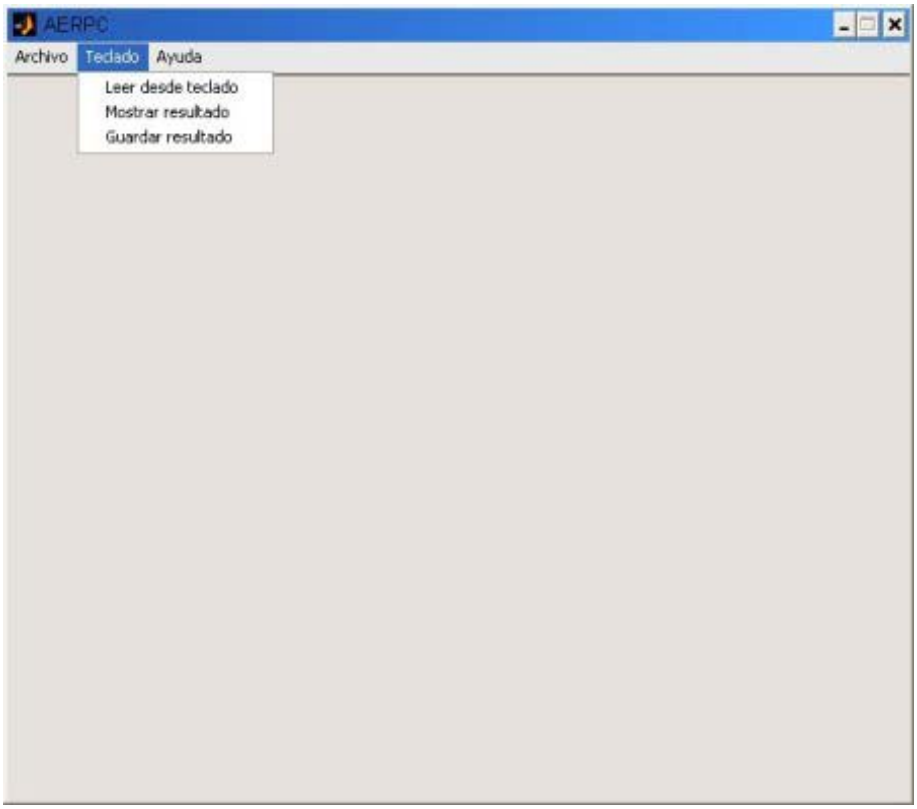


Figura B-19: Mostrar resultado

Al seleccionar esta opción se muestra en pantalla la solución óptima y los multiplicadores de Lagrange asociados a ésta, encontrados al resolver el problema introducido por teclado (figura B-20). Dentro de esta vista, se tiene el botón imprimir, que nos permite imprimir la ventana de resultados. Si se desea imprimir el resultado en formato texto, es necesario guardarlo, como se muestra en la sección guardar teclado, abrir el archivo de texto, utilizando algún procesador

de texto o el editor de Matlab y utilizar las opciones de impresión que ofrece.

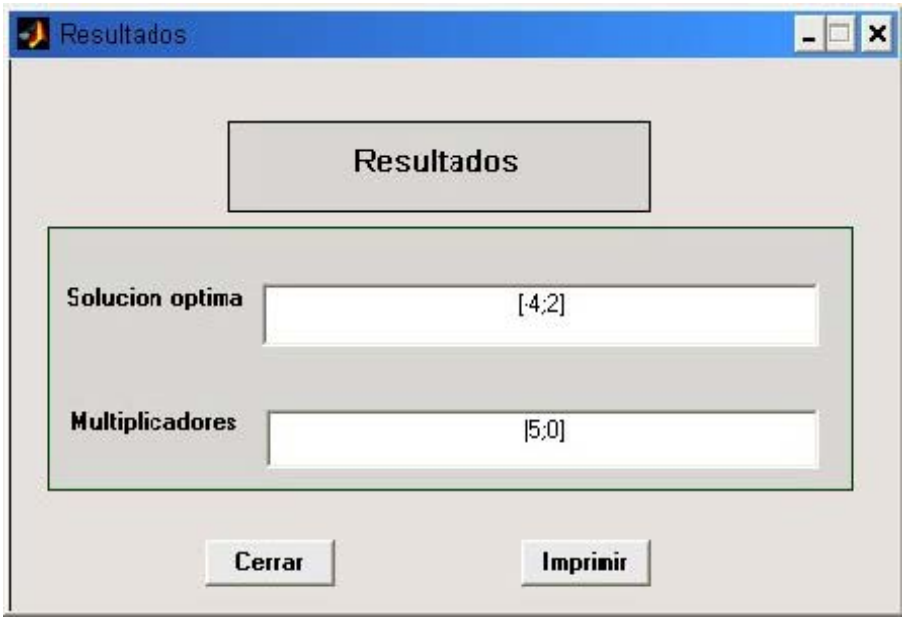


Figura B-20: Resultados

B.6. Guardar resultado.

Esta opción permite guardar el resultado obtenido después de resolver un problema introducido por teclado, la opción está disponible en el menú Teclado (figura B-21). Para más detalles de esta opción, ver "Guardar resultados"del menú Archivo.

B.7. Salir del programa.

Para finalizar la sesión de trabajo con el sistema AERPC se puede hacer de dos formas:

1. Utilizando el botón cerrar de la ventana principal del sistema (figura B-22).
2. Seleccionando la opción Salir del menú Archivo (figura B-23).

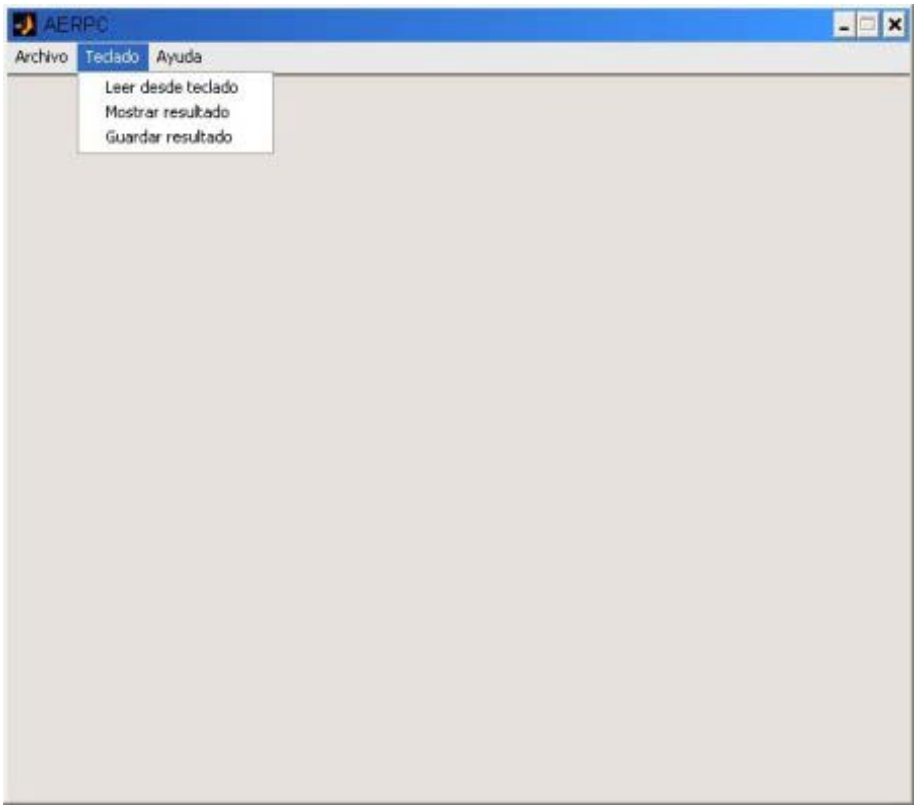


Figura B-21: Guardar resultado

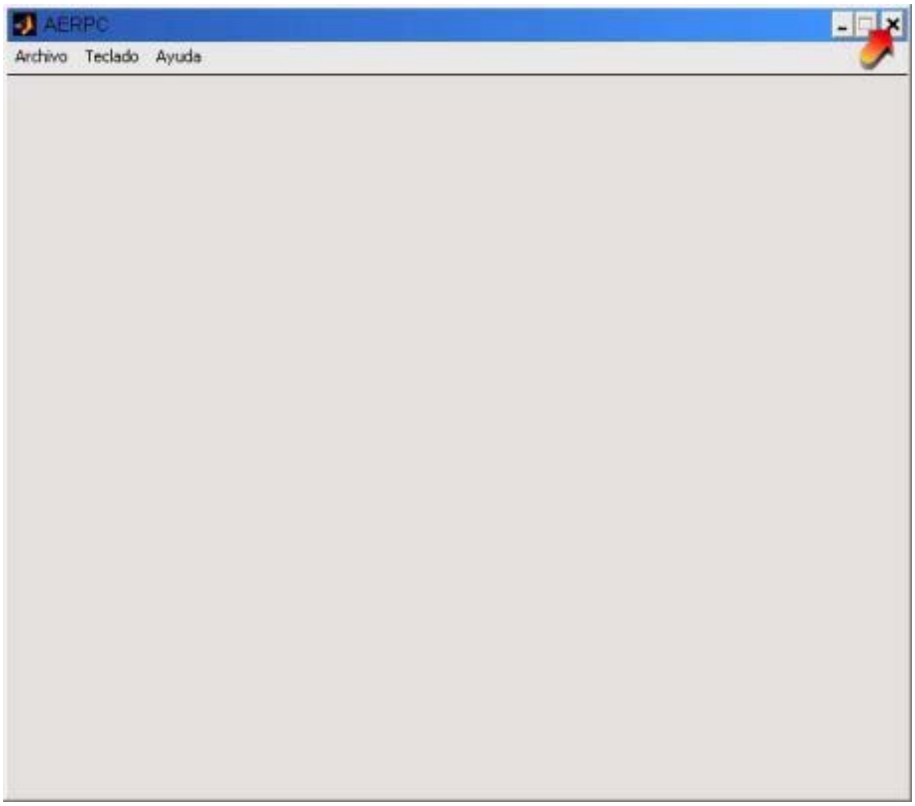


Figura B-22: Cerrar aplicación

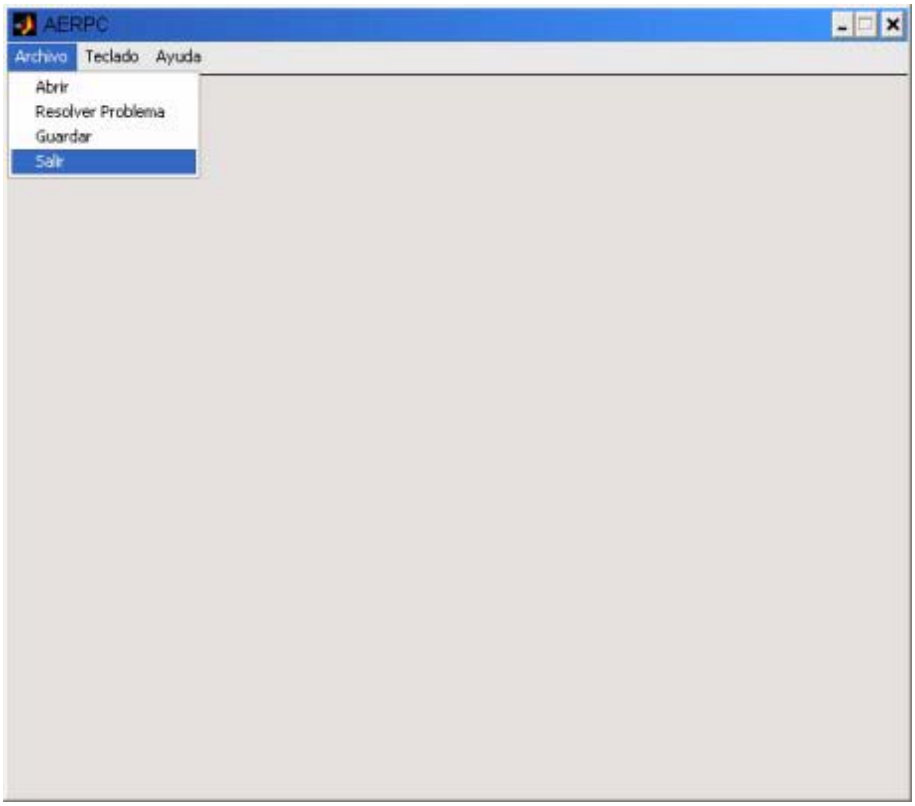


Figura B-23: Salir desde el menú

Apéndice C

Formato de archivo.

El formato que deben de respetar los problemas a ser cargados desde archivo, es el definido en el "Generador de Problemas Prueba para Programación Cuadrática" [10] y debe tener un formato de texto simple. Puede ser generado en el editor de Matlab o en algún procesador de texto, con la extensión .m:

1. La primer línea debe contener el número de variables (n).
2. La siguiente puede contener cualquier valor (ver [10] para mayor información).
3. La línea número tres debe contener el número de restricciones (m).
4. La cuarta línea contiene el número de restricciones activas en el punto óptimo.
5. La siguiente línea, la 5, contiene el número de restricciones activas simultaneamente en el punto óptimo y en el inicial.
6. La línea 6 contiene las restricciones activas en el punto inicial.
7. La línea 7 debe contener el texto "MATRIZ G".
8. A partir de la línea 8 los valores de la matriz Hessiana, anotando un número por línea y definiendo fila por fila.
9. En la siguiente línea el texto "VECTOR c".

10. A continuación los valores del vector de costos, anotando un valor por línea.

11. El texto “MATRIZ A” en la línea siguiente.

12. Después los valores de esta matriz anotando uno en cada fila y definiendo fila por fila.

13. A continuación el texto “VECTOR b”, seguida por sus valores, anotando uno en cada línea.

14. Finalmente el texto “PUNTO INICIAL” y a continuación, en cada línea, sus valores.

Ejemplo: Se enuncia un archivo que define un problema de 2 variables con 2 restricciones.

2
0
2
1
1
1
MATRIZG
2.148950616441588e+000
5.336629832559923e-001
5.336629832559925e-001
2.218052482717586e+000
VECTOR c
-8.023606446361151e+001
3.879434298496039e+001
MATRIZA
2.413242056528027e-001
3.576752739965297e-001
-1.414137503006851e+000
-2.600795074410644e+000
VECTORb
1.246198635477517e+000

-6.878002678099955e+000

PUNTOINICIAL

6.410555012507196e+000

-8.410518774420162e-001

Apéndice D

Conceptos básicos.

Definición 1 ***(Conjunto convexo).***

Sea $\Omega \in \Re^n$*, decimos que* Ω *es un conjunto convexo si* $\forall\ x_1, x_2 \in \Omega$ *se cumple que el segmento* $[x_1, x_2]$ *está contenido en* Ω*, donde*

$$[x_1, x_2] = \{\alpha x_1 + (1-\alpha)\, x_2 | \alpha \in [0,1]\}\,.$$

Ejemplo 1:

Conjuntos convexos Conjuntos no convexos

Definición 2 ***(Dirección factible).***

Dado $S \subseteq \Re^n, x_0 \in S$*, diremos que* d *es una dirección factible en* x_0 *respecto a* S *si existe* $\varepsilon_0 > 0$ *tal que* $x_0 + \varepsilon d \in S, \forall\ 0 \leq \varepsilon \leq \varepsilon_0$*. Denotaremos* $D(x_0)$ *el conjunto de todas las direcciones factibles en* x_0*.*

Definición 3 ***(Función convexa).***

Sea $\Omega \subseteq \Re^n$ *convexo, y* $f : \Omega \to \Re$*, diremos que* f *es convexa si para cualesquiera* $x_1, x_2 \in \Omega$*,* $\alpha \in (0, 1)$ *se cumple:*

$$f(\alpha x_1 + (1 - \alpha)x_2) \leq \alpha f(x_1) + (1 - \alpha)f(x_2).$$

Definición 4 ***(Función estrictamente convexa).***

Sea $\Omega \subseteq \Re^n$ *convexo, y* $f : \Omega \to \Re$*, diremos que* f *es estrictamente convexa si para cualesquiera* $x_1, x_2 \in \Omega, \alpha \in (0, 1)$ *se cumple:*

$$f(\alpha x_1 + (1 - \alpha)x_2) < \alpha f(x_1) + (1 - \alpha)f(x_2).$$

Ejemplo 2:

Sean $f, g : \Omega \to \Re$ funciones convexas y $\theta_1, \theta_2 \in \Re, \theta_1 \geq 0, \theta_2 \geq 0$, entonces $\theta_1 f + \theta_2 g$ es una función convexa.

En efecto, dados $x_1, x_2 \in \Omega, \alpha \in (0, 1)$.

$$\begin{aligned}
&(\theta_1 f + \theta_2 g)(\alpha x_1 + (1 - \alpha)x_2) = \theta_1 f(\alpha x_1 + (1 - \alpha)x_2) + \theta_2 g(\alpha x_1 + (1 - \alpha)x_2) \\
&\leq \theta_1[\alpha f(x_1) + (1 - \alpha)f(x_2)] + \theta_2 [\alpha g(x_1) + (1 - \alpha)g(x_2)] \\
&= \alpha[\theta_1 f(x_1) + \theta_2 g(x_1)] + (1 - \alpha) \left[\theta_1 f(x_2) + \theta_2 g(x_{2)}\right] \\
&= \alpha(\theta_1 f + \theta_2 g)(x_1) + (1 - \alpha)(\theta_1 f + \theta_2 g)(x_2).
\end{aligned}$$

Definición 5 ***(Vector gradiente).***

El vector gradiente es el vector que nos indica hacia dónde aumenta la función, mientras que la dirección contraria indica hacia dónde disminuye, el vector gradiente es un vector cuya i*-ésima componente es la derivada parcial de* $f(x)$ *con respecto a* x_i *y se denota por:*

$$\nabla f(x) = \left[\frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, ..., \frac{\partial f(x)}{\partial x_i}, ..., \frac{\partial f(x)}{\partial x_n} \right],$$

donde cada punto de este vector es ortogonal a las curvas de nivel.

Definición 6 ***(Matriz Hessiana).***

La matriz Hessiana $Q_{nxn}(x)$,es una matriz que está determinada por:

$$Q(x) = \frac{\partial^2 f}{\partial x_i \, \partial x_j}.$$

evaluada en los $i - ésimo$ y $j - ésimo$ elementos. La matriz Hessiana es definida positiva, si y solo si la función $f(x)$ a la que está asociada es estrictamente convexa.

Definición 7 ***(Solución factible o punto factible).***

Considerando el problema de minimizar $f(x)$ sujeto a $x \in \Omega$, un punto $x \in \Omega$ es una solución factible para este problema.

Definición 8 ***(Mínimo).***

Dado $f(x) \in \Omega$. Si existe $f(x^*) \in \Omega$ tal que:

$$x \in \Omega \qquad \Rightarrow \qquad f(x) \geq f(x^*),$$

entonces $f(x^*)$ es llamado el mínimo de $f(x) \in \Omega$, denotado por

$$f(x^*) = \min_{x \in \Omega} f(x).$$

Definición 9 ***(Mínimo local).***

Dado $\Omega \subseteq \Re^n$, y $f : \Omega \to \Re$, diremos que $x^* \in \Omega$ es un punto de mínimo local de f en Ω si para alguna vecindad V de x^* se cumple que:

$$f(y) \geq f(x^*), \forall\ y \in V \cap \Omega.$$

Definición 10 ***(Mínimo local estricto).***

Dado $\Omega \subseteq \Re^n$, y $f : \Omega \to \Re$, diremos que $x^* \in \Omega$ es un punto de mínimo local estricto de f en Ω si para alguna vecindad V de x^* se cumple que:

$$f(y) > f(x^*), \forall\ y \in V \cap \Omega.$$

Definición 11 ***(Mínimo global).***

Dado $x^* \in \Omega$, diremos que x^* es un punto de mínimo global de f en Ω, si $f(x) \geq f(x^*)\ \forall\ x \in \Omega$.

Definición 12 ***(Mínimo global estricto).***

Dado $x^* \in \Omega$, diremos que x^* es un punto de mínimo global de f en Ω, si $f(x) > f(x^*)\ \forall\ x \in \Omega$.

Definición 13 ***(Punto regular).***

Sea $x^* \in S$. Diremos que x^* es un punto regular (o que cumple las condiciones de regularidad) si los gradientes de las restricciones que se cumplen en la igualdad en x^* forman un conjunto de vectores linealmente independientes. Se consideran también puntos regulares los que no saturan ninguna restricción, es decir, los puntos interiores del conjunto factible.

Definición 14 ***(Restricción activa).***

Dado un conjunto de restricciones de desigualdad $A^T x \geq b$, se le denomina restricción activa a la i-ésima restricción a_i, que en un punto x factible se cumple en la igualdad, es decir:

$$a_i^T x = b_i.$$

D.1. Optimización sin restricciones.

Antes de comenzar con la definición de los términos utilizados para resolver problemas del tipo (1.1), es necesario conocer los términos utilizados para el caso general, lo que servirá para empezar a adentrarse y conocer más el problema a resolver.

D.1.1. Condiciones necesarias de optimalidad.

Las condiciones necesarias de optimalidad, son condiciones que deben ser cumplidas para poder considerar a un punto como un mínimo del problema y por lo tanto ser considerado como solución.[2], [3] y [4].

Teorema 1. (Condiciones necesarias de primer orden).

Tomemos a $f(x)\ \epsilon\ C^1$ en $\Re^n$ para x^*. Si x^* es un mínimo local, entonces:

$$\nabla f(x^*) = 0, \tag{1.2}$$

los puntos que cumplen con (1.2) son conocidos como puntos estacionarios.

Teorema 2. (Condiciones necesarias de segundo orden).

Dada $f(x)\ \epsilon\ C^2$ en $\Re^n$ para x^*. Si x^* es un mínimo local, entonces:

1. $\nabla f(x^*) = 0$, y
2. La matriz Hessiana $Q(x^*)$, es semidefinida positiva; esto es $y^T Q(x^*) y \geq 0$, para toda $y \in \Re^n$.

D.1.2. Condiciones suficientes de optimalidad.

Las condiciones suficientes de optimalidad son las condiciones que, si se satisfacen en un punto, garantizan que ese punto sea un mínimo. [3], [4], [5] y [6].

Teorema 3. (Condiciones suficientes de optimalidad).

Dada una función $f(x)\ \epsilon\ C^2$ en $\Re^n$ para x^*. Si

1. $\nabla f(x^*) = 0$, y
2. $Q(x^*)$ es semidefinida positiva,

entonces, x^* es un mínimo local. Si en la condición (2) $Q(x^*)$ es definida positiva, entonces x^* es un mínimo local estricto.

Bibliografía

[1] Avendaño Pérez,Carlos (2001). "Algoritmo de Espacio Nulo para Programación Cuadrática". Tesis, Universidad Tecnológica de la Mixteca

[2] García Fernández, Lina (1996). Programación Cuadrática Convexa, Método de Restricciones Activas. Benemérita Universidad Autónoma de Puebla, Facultad de Ciencias de la Computación.

[3] Luenberger, David E. (1989). Programación Lineal y No Lineal. Addison-Wesley Iberoamericana.

[4] Floudas, Christodoulos A. (1995). Nonlinear and Mixed-Integer Optimization, Fundamentals and Applications. New York Oxford - Oxford University Press.

[5] Nocedal and Wright (1999) "Numerical Optimization", Springers Series in Optimization Research.

[6] Gass, S.I. and Harris, Carl M. (1993), "Encyclopedia of Operations Research and Magament Science".

[7] Gill, PhilipE., Murray, Walter, Wright, Margaret H. (1990). Numerical Linear algebra and Optimization Volume 1. Addison - Wesley Publishing Company.

[8] Golub G. (1989) "Matrix Computations", Johns Hopkins. University Press.

[9] Camponogara, Eduardo (2003). Métodos de Optimización, Teoría y Práctica. Universidad Federal de Santa Catarina.

[10] L. Marcial, L. García y L. Sandoval (1993). "Generador Aleatorio de Problemas Prueba Para Programación Cuadrática". Reporte de Investigación No. 1862E (CONACyT). Benemérita Universidad Autónoma de Puebla.

[11] Nakamura (1997) "Análisis Numérico y Visualización Gráfica con Matlab". Prentice Hall.

[12] http://departamentos.unican.es/macc/personal/profesores/castillo/Libro/Chap8.pdf