OPTIMIZACIÓN DE FUNCIONES EN VARIAS VARIABLES MEDIANTE DESIGUALDADES LINEALES MATRICIALES

IVONNE TORRES URBANO

OMAR GUIOVANNI ZÚÑIGA CAMACHO

DEPARTAMENTO DE MATEMÁTICAS
FACULTAD DE CIENCIAS NATURALES, EXACTAS Y DE LA EDUCACIÓN
UNIVERSIDAD DEL CAUCA
POPAYÁN
2009

OPTIMIZACIÓN DE FUNCIONES EN VARIAS VARIABLES MEDIANTE DESIGUALDADES LINEALES MATRICIALES

IVONNE TORRES URBANO

OMAR GUIOVANNI ZÚÑIGA CAMACHO

Proyecto presentado como requisito para optar el título de Matemático

Director
Doctor FREDDY AMAYA

DEPARTAMENTO DE MATEMÁTICAS
FACULTAD DE CIENCIAS NATURALES, EXACTAS Y DE LA EDUCACIÓN
UNIVERSIDAD DEL CAUCA
POPAYÁN
2009

AGRADECIMIENTOS

Este proyecto es el resultado del trabajo realizado con el grupo de matemática computacional a cargo del proyecto de investigación titulado *Nuevas Alternativas para la Estimación de los Parámetros en una Gramática Incontextual Probabilística (GIP)* (proyecto 1579 inscrito en la Vicerrectoría de Investigaciones de la Universidad del Cauca). Terminarlo no hubiera sido posible sin el ánimo, la colaboración y el apoyo de muchas personas.

En primer lugar agradecemos a Dios por darnos el privilegio de llegar a esta etapa de nuestras vidas.

Queremos dar un agradecimiento especial al Doctor *Freddy Ángel Amaya*, nuestro director de trabajo de grado y director del proyecto, por su apoyo, dedicación, consejos y observaciones durante la realización del mismo.

Agradecemos también a los miembros del grupo de investigación: los docentes Mauricio Maca, Diego Correa, Luis Eduardo Montoya y Hebert Vivas; a los compañeros Edwin Murillo, Jhonny Ibarra, Astrid Alvarez y Ester Mamian; a todos mil gracias por su apoyo, colaboración y sugerencias. Hemos compartido largas horas probando, implementando, analizando y corrigiendo el proyecto.

Finalmente, un cariñoso agradecimiento a las personas más cercanas, nuestras familias y amigos, por ayudarnos con nuestra educación y la formación en casa, por darnos un apoyo incondicional para llevar a cabo todo este proceso, que inició desde nuestra infancia buscando hacernos personas de bien, y que hoy se ve reflejado en este trabajo.

RESUMEN

En este documento se presenta el trabajo titulado *Optimización de funciones en varias variables mediante Desigualdades Lineales Matriciales* desarrollado dentro del grupo de matemática computacional, en la línea de reconocimiento de formas, exigido como requisito parcial para la obtención del título de Matemático otorgado por la Universidad del Cauca. En él analizaremos la aplicación de un método de optimización que utiliza la teoría de momentos probabilísticos para optimizar polinomios en varias variables, convirtiendo el problema en una sucesión de problemas de optimización por Desigualdades Lineales Matriciales. En el trabajo además se estudia su aplicación en la estimación de los parámetros de una Gramática Incontextual Probabilística. En el primer capítulo se hará un breve repaso de los conceptos básicos de: gramáticas, gramáticas libres de contexto (GLC), gramáticas libres de contexto probabilisticas (GLP), modelos de lenguaje probabilísticos y estimación de parámetros. Veremos que la función de verosimilitud asociada a las reglas de una gramática incontextual probabilística es un polinomio en varias variables.

En el segundo capítulo se estudia un concepto útil para el desarrollo del método de optimización de polinomios en varias variables basado en la teoría de momentos; las *Desigualdades Lineales Matriciales*, de las cuales se darán algunos conceptos básicos. Este tema se da como un preámbulo de la aplicación de la teoría de momentos en la optimización de polinomios en varias variables.

En el tercer capítulo se estudiará el método de optimización de polinomios en varias variables mediante desigualdades lineales matriciales basado en la teoría de momentos presentado por Didier Henrion y Jean Lasserre en su artículo *Global Optimization with polynomials and the Problem of Moments*[9]. Aquí se mostrará cómo la teoría de momentos nos ofrece un método no convencional para calcular los valores óptimos de un polinomio en varias variables con o sin restricciones.

El capítulo 4 corresponde a la experimentación computacional del método, utilizando una herramienta diseñada por los franceses Didier Henrion y Jean Lasserre [5] para implementar el método. Se harán pruebas con diferentes problemas de optimización y con Gramáticas generadas por una herramienta de simulación y por último se analizarán los resultados obtenidos.

CONTENIDO

Capítulo 1

GRAMÁTICAS INCONTEXTUALES

1.1. Preliminares

La lingüística computacional es una rama de la informática que usa modelos matemáticos y herramientas computacionales, con el fin de mejorar los sistemas automáticos que procesan diferentes aspectos del manejo del lenguaje, tanto natural como formal. Entre los lenguajes formales utilizados tanto para procesar lenguaje natural como para otras aplicaciones estan los generados por las Gramáticas Incontextuales Probabilísticas (GIPs). La importancia de las GIPs radica en que el tipo de modelo que produce provee mayor información que otros modelos, pero en contraste el coste computacional que requieren para la estimación de los parámetros en tareas de cierta complejidad es elevado. Hasta ahora los modelos matemáticos y algoritmos propuestos y utilizados en la estimación de los parámetros de las GIPs si bien producen modelos de lenguaje con una buena capacidad expresiva, su coste computacional es elevado, por lo cual se requiere buscar métodos más eficientes para realizar la estimación de los parámetros.

En lo que resta del capítulo se hará un breve estudio sobre gramáticas, GIPs, su estimación y se planteará el problema de optimizar la función de verosimilitud asociada a las reglas de la GIP.

1.2. Cadenas, alfabetos y lenguajes

Definición 1. *Conceptos básicos sobre cadenas, alfabetos y lenguajes.*

- *Un alfabeto o vocabulario, notado como Σ, es un conjunto finito de símbolos.*
- *Una cadena (o frase) es una secuencia finita de símbolos (elementos de Σ).*
- *La longitud de una cadena x es el número de elementos que tiene, se denota $|x|$.*
- *Una cadena vacía es aquella que no posee elementos y se denota ϵ ($|\epsilon| = 0$).*
- *Por Σ^* se entiende el conjunto de todas las cadenas de Σ.*
- *Por Σ^+ se entiende el conjunto de todas las cadenas de símbolos de Σ que tienen longitud mayor o igual que 1.*
- *Un lenguaje L sobre Σ se define como un subconjunto del conjunto Σ^*.*

1.3. Gramáticas y Gramáticas Incontextuales

Definición 2. *Una gramática formal es una 4-tupla $G = (N, \Sigma, P, S)$, donde:*

N : *Conjunto finito, denominado conjunto de variables o no terminales.*

Σ : *Es el alfabeto. N y Σ son disjuntos.*

P : *Conjunto finito de reglas de producción (o de derivación): cada producción es de la forma $\boldsymbol{\alpha} \longrightarrow \boldsymbol{\gamma}$, en donde $\boldsymbol{\alpha}$ (Antecedente) y $\boldsymbol{\gamma}$ (Consecuente) son cadenas de símbolos tomados de $(N \cup \Sigma)^*$.*

S : *Símbolo inicial (Axioma) de la gramática. $S \in N$.*

Notación: La expresión $\alpha \rightarrow \gamma$, donde α y γ son cadenas de $(N \cup \Sigma)^*$, significa que la cadena α deriva en la cadena γ o que α es sustituido por γ.

Notación: La expresión $\alpha \overset{i}{\Rightarrow} \beta$ significa que α deriva en β exactamente en i pasos. La expresión $\alpha_1 \overset{a_1}{\Rightarrow} \alpha_2 \overset{a_2}{\Rightarrow} \alpha_3 \cdots \alpha_{k-1} \overset{a_{k-1}}{\Rightarrow} \alpha_k$ significa que $\alpha_1 \overset{a_1}{\Rightarrow} \alpha_2$, $\alpha_2 \overset{a_2}{\Rightarrow} \alpha_3$,..., $\alpha_{k-1} \overset{a_{k-1}}{\Rightarrow} \alpha_k$. La expresión $\alpha \overset{*}{\Rightarrow} \beta$ significa que α deriva en β, mediante alguna sucesión de derivaciones de la forma $\alpha \overset{a_1}{\Rightarrow} \alpha_1 \overset{a_2}{\Rightarrow} \alpha_2 \cdots \alpha_k \overset{a_m}{\Rightarrow} \beta$.

Definición 3. *Una derivación a izquierda de una cadena $x \in \Sigma^+$ en G, es una sucesión de reglas de derivación $dx = (q_1, q_2, \ldots, q_m)$, $m \geq 1$, tal que: $\left(S \overset{q_1}{\Rightarrow} \alpha_1 \overset{q_2}{\Rightarrow} \alpha_2 \cdots \overset{q_m}{\Rightarrow} x\right)$, donde $\alpha_i \in (N \cup \Sigma)^+$, $1 \leq i \leq m+1$ y q_i reescribe el terminal más a la izquierda de α_{i-1}*

En el siguiente ejemplo se aclara este punto:

Ejemplo 1. Sea $G = (N, \Sigma, P, S)$, donde $N = \{A, B, S\}$, $\Sigma = \{a, b\}$, S es el símbolo inicial y P consiste en:

$$\begin{aligned} S &\rightarrow aA \\ aB &\rightarrow bBAa \\ ABb &\rightarrow abB \end{aligned}$$

Note que la cadena $bbaBB$ se transforma en la cadena $bbbBAaB$ por la aplicación de la segunda regla.

En adelante utilizaremos las siguientes convenciones con respecto a las gramáticas:

- Las letras mayúsculas tales como A,B, C, D, E y S representan no terminales; S es el símbolo inicial.
- Las letras minúsculas como a, b, c, d, e y los dígitos serán los símbolos terminales.
- Las letras minúsculas u, v, w, s denotan cadenas de terminales.
- Las letras griegas minúsculas α, β, γ se usan para denotar cadenas de terminales y no terminales.

Definición 4. *Una Gramática G, para la cual el conjunto P de reglas de derivación esta constituido por reglas de la forma $A \rightarrow \alpha$ donde $A \in N$ y $\alpha \in (N \cup \Sigma)^+$, se denomina Gramática Incontextual (GI).*

Definición 5. *Una Gramática Incontextual G se encuentra en Forma Normal de Chomsky (FNC), si las reglas de derivación tienen la forma $A \rightarrow BC$ o $A \rightarrow v$ donde $A, B, C \in N$ y $v \in \Sigma$.*

Nota: En adelante cuando se hable de derivación, se entenderá como derivación a izquierda.

Definición 6. *El lenguaje generado por una gramática G es el conjunto $L(G)$, definido como:*

$$L(G) = \left\{ x \in \Sigma^+ \mid S \overset{*}{\Rightarrow} x \right\} \tag{1.1}$$

Definición 7. *Un lenguaje probabilístico sobre un alfabeto Σ es un par (L, Φ), con L un lenguaje formal y Φ medida de probabilidad sobre Σ^*, tal que $\Phi(x) > 0$ si $x \in \Sigma^*$*

Definición 8. *Una Gramática Incontextual Probabilística (GIP) G_p es una pareja (G, p) tal que G es una Gramática Incontextual y p una función $p : P \rightarrow (0, 1]$ sobre las reglas de la gramática de tal forma que:*

$$\forall A \in N, \sum_{(A \rightarrow \alpha) \in P} p(A \rightarrow \alpha) = 1 \tag{1.2}$$

1.4. Probabilidad de una cadena de una GIP

Sea G_p una GIP. Para cada $x \in L(G)$ se denomina D_x al conjunto formado por todas las derivaciones a izquierda de la cadena x. Por $N(A \to \alpha, d_x)$ se designa el número de veces que la regla $A \to \alpha$ se ha utilizado en la derivación d_x.

Definición 9. *Dada una GIP G_p, la probabilidad de una derivación d_x de la cadena $x \in \Sigma^*$ se define como:*

$$\Pr(x, d_x|G_p) = \prod_{(A\to\alpha)\in P} p(A \to \alpha)^{N(A\to\alpha,d_x)} \tag{1.3}$$

Definición 10. *Dada una GIP G_p, la probabilidad de la cadena $x \in \Sigma^*$ se define como:*

$$\Pr(x|G_p) = \sum_{d_x\in D_x} \Pr(x, d_x|G_p) \tag{1.4}$$

Definición 11. *Dada una GIP G_p, Se llama probabilidad de la mejor derivación de la cadena x a:*

$$\widehat{\Pr}(x|G_p) = \max_{d_x\in D_x} \Pr(x, d_x|G_p) \tag{1.5}$$

Definición 12. *El lenguaje generado por una gramática incontextual probabilística $G_p = (G, p)$, se define como $L(G_p) = \{x \in L(G) : \Pr(x|G_p) > 0\}$. Una GIP G_p se denomina consistente, si el lenguaje generado por G_p es estocástico, es decir:*

$$\sum_{x\in L(G_p)} \Pr(x|G_p) = 1. \tag{1.6}$$

1.5. Estimación de los Parámetros de una GIP

El problema que queda es determinar las probabilidades de las reglas de la gramática, o como suele llamarse, problema de estimación de los parámetros de una GIP.

El problema de la estimación de los parámetros de una GIP $G_p = (G, p)$ puede plantearse de la siguiente manera: sea (L, Φ) un lenguaje estocástico donde $L \subseteq L(G)$, y Φ es una función de probabilidad desconocida. Dada una muestra Ω de L, se tiene que estimar la función p de tal manera que mediante ella se pueda representar a Φ. El problema planteado queda de la forma:

$$\widehat{p} := \max_{p} f_p(\Omega) \tag{1.7}$$

donde f_p es una función a ser optimizada. De este modo, para estimar los parámetros de una GIP es necesario determinar la función a optimizar y el método de optimización que se va a utilizar. El método de optimización de uso más generalizado para la estimación de los parámetros en una GIP, esta basado en el Teorema 1 (Transformaciones Crecientes).

Teorema 1 (Transformaciones crecientes)**.** *Sea $P(\Theta)$, con $\Theta = \{\Theta_{ij}\}$, un polinomio homogéneo de grado d con coeficientes no negativos. Sea $\theta = \{\theta_{ij}\}$ un punto del conjunto*

$$D = \left\{ \theta_{ij} \mid \theta_{ij} \geq 0,\ \sum_{j=1}^{q_i} \theta_{ij} = 1,\ i = 1, \ldots, k\ j = 1, \ldots, q_i \right\}$$

para k, q_i enteros. Sea $Q(\Theta)$ una transformación en D, definida como:

$$Q(\theta_{ij}) = \frac{\theta_{ij}\left(\frac{\partial P}{\partial \Theta_{ij}}\right)_\theta}{\sum_{k=1}^{q_i} \theta_{ik}\left(\frac{\partial P}{\partial \Theta_{ik}}\right)_\theta}$$

tal que para todo i, $\sum_{k=1}^{q_i} \theta_{ik}\left(\frac{\partial P}{\partial \Theta_{ik}}\right)_\theta \neq 0$. Entonces $P(Q(\theta)) > P(\theta)$ para $Q(\theta) \neq \theta$.

La aplicación del teorema anterior permite obtener un máximo local del polinomio P en el espacio de búsqueda definido por D haciendo uso del siguiente algoritmo de estimación:

Algoritmo de estimación

Entrada	$P(\Theta)$
Salida	Θ
Método	$\Theta =$ `valores iniciales`
	repetir `calcular` $Q(\Theta)$ `utilizando` $P(\Theta)$
	$\Theta = Q(\Theta)$
	Hasta `Converger.`

Dado un polinomio que cumple las condiciones del teorema 1 y un punto inicial $\theta \in D$, el problema de la estimación se plantea como la aplicación repetida de la transformación hasta alcanzar un máximo local del polinomio definido, ó hacer la estimación hasta que la diferencia entre 2 iteraciones consecutivas esté por debajo de un umbral. Note que las probabilidades de las reglas de una GIP pueden ser puntos del dominio definido en el teorema 1 para una GIP G_p:

$$p(A_i \to \alpha_j) = \theta_{ij} \quad i = 1, \ldots, |N|\,; \quad j = 1, \ldots, |\Gamma_{A_i}| \tag{1.8}$$

donde Γ_{A_i} representa el conjunto de reglas de la gramática cuyo antecedente es A_i.

1.5.1. Función de verosimilitud

Entre las funciones objetivo a optimizar está la función de verosimiltud, (ver [2],[4])

Para el caso de las GIPs, la función de verosimilitud de una muestra Ω de $L(G)$ notada $\Pr(\Omega|G_p)$, está dada por la ecuación:

$$\Pr(\Omega|G_p) = \prod_{x\in\Omega} \Pr(x|G_p). \tag{1.9}$$

Si se utiliza la función de verosimilitud como función objetivo en (1.7), puede notarse que dicha función es un polinomio en varias variables, si se ordenan las reglas de alguna forma y a la regla i-ésima se le asigna la variable x_i, $i = 1, 2, \ldots, |P|$, lo cual se ilustra con el siguiente ejemplo:

Ejemplo 2. Sea $G = (N, \Sigma, P, S)$, donde $N = \{A, B, C, S\}$, $\Sigma = \{a, b\}$, S es el símbolo inicial y P consiste en:

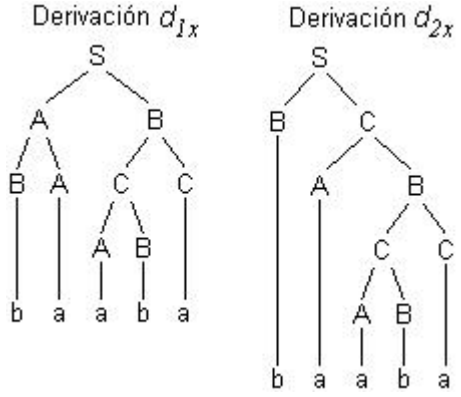
1) $S \to AB$	3) $A \to BA$	5) $B \to CC$	7) $C \to AB$
2) $S \to BC$	4) $A \to a$	6) $B \to b$	8) $C \to a$

Sea x_i la probabilidad de la regla i-ésima de la gramática, es decir:

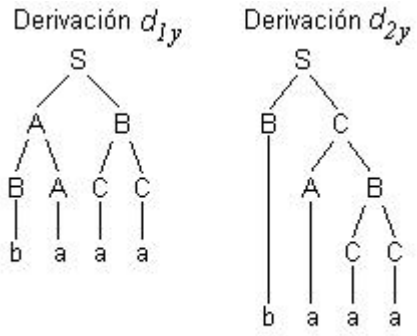
$x_1 = p(S \to AB)$	$x_3 = p(A \to BA)$	$x_5 = p(B \to CC)$	$x_7 = p(C \to AB)$
$x_2 = p(S \to BC)$	$x_4 = p(A \to a)$	$x_6 = p(B \to b)$	$x_8 = p(C \to a)$

Sea $\Omega = \{baaba, baaa\} \subset L(G)$. Sean $x = baaba$ e $y = baaa$, los árboles de derivación para los elementos de la muestra son:

$D_x = \{d_{1x}, d_{2x}\}$



$D_y = \{d_{1y}, d_{2y}\}$



El número de veces N que se usa cada regla es:

	D_x		D_y	
Regla	$N(d_{1x})$	$N(d_{2x})$	$N(d_{1y})$	$N(d_{2y})$
$S \to AB$	1	0	1	0
$S \to BC$	0	1	0	1
$A \to BA$	1	0	1	0
$A \to a$	2	2	1	1
$B \to CC$	1	1	1	1
$B \to b$	2	2	1	1
$C \to AB$	1	2	0	1
$C \to a$	1	1	2	2

De la ecuación (1.3) se tiene que:

$$\Pr(x, d_{1x}|G_p) = x_1x_3x_4^2x_5x_6^2x_7x_8 \qquad \Pr(y, d_{1y}|G_p) = x_1x_3x_4x_5x_6x_8^2$$
$$\Pr(x, d_{2x}|G_p) = x_2x_4^2x_5x_6^2x_7^2x_8 \qquad \Pr(y, d_{2y}|G_p) = x_2x_4x_5x_6x_7x_8^2$$

Luego, de la ecuación (1.4), se tiene que:

$$\Pr(x|G_p) = \Pr(x, d_{1x}|G_p) + \Pr(x, d_{2x}|G_p) = x_1x_3x_4^2x_5x_6^2x_7x_8 + x_2x_4^2x_5x_6^2x_7^2x_8$$
$$\Pr(y|G_p) = \Pr(y, d_{1y}|G_p) + \Pr(y, d_{2y}|G_p) = x_1x_3x_4x_5x_6x_8^2 + x_2x_4x_5x_6x_7x_8^2$$

Ahora, de la ecuación (1.9) tenemos que la función de verosimilitud de la muestra Ω esta dada por:

$$\begin{aligned}
\Pr(\Omega|G_p) &= \Pr(x|G_p) * \Pr(y|G_p) \\
&= (x_1x_3x_4^2x_5x_6^2x_7x_8 + x_2x_4^2x_5x_6^2x_7^2x_8) * (x_1x_3x_4x_5x_6x_8^2 + x_2x_4x_5x_6x_7x_8^2) \\
&= x_1^2x_3^2x_4^3x_5^2x_6^3x_7x_8^3 + x_1x_2x_3x_4^3x_5^2x_6^3x_7^2x_8^3 \\
&\quad + x_1x_2x_3x_4^3x_5^2x_6^3x_7^2x_8^3 + x_2^2x_4^3x_5^2x_6^3x_7^3x_8^3
\end{aligned}$$

Dado que $\Pr(\Omega|G_p)$ es un polinomio en varias variables, se tiene que el problema de estimación de los parámetros de la GIP se convierte en el problema de optimizar un polinomio en varias variables sujeto a restricciones lineales de la forma: $0 \leq x_i \leq 1$, $i = 1, 2, \ldots, |P|$ y $\sum_{x_i \in \Psi_\alpha} x_i = 1$, donde Ψ_α es el conjunto de todas las x_i que representan la probabilidad de una regla cuyo antecedente es α.

Entre los métodos de optimización de funciones de $\mathbb{R}^n$ en $\mathbb{R}$, el método basado en la teoría de los momentos está diseñado para el caso particular de los polinomios en varias variables. Este método, desarrollado por los franceses Didier Henrion y Jean

Lasserre puede ser una alternativa para la solución del problema (1.7) tomando como función objetivo f_p la función de verosimilitud. Cabe agregar que dicho método utiliza además de la teoría de los momentos, la teoría referente a las desigualdades lineales matriciales para resolver el problema.

Dado que la función de verosimilitud de una GIP es un polinomio en varias variables, y que el uso del teorema de transformaciones crecientes para encontrar el máximo de tal función produce un algoritmo que tiene un alto coste computacional, exploramos el método de optimización de polinomios en varias variables mediante el método de los momentos, como alternativa para la estimación de los parámetros de la GIP.

En el siguiente capítulo se realiza un estudio previo referente a desigualdades lineales matriciales.

Capítulo 2

DESIGUALDADES LINEALES MATRICIALES

En el presente capítulo, se tratará el tema relacionado con las *Desigualdades Lineales Matriciales* (DLM) (En ingles *Linear Matrix Inequality* LMI). Se darán primero los conceptos de matriz definida positiva y negativa, semidefinida positiva, función definida positiva y negativa, semidefinida positiva; para luego definir una DLM y a continuación se da el esquema básico para la solución de problemas que incluyen DLMs.

2.1. Preliminares

Definición 13. Sean A,B matrices $n \times n$ y 0 la matriz nula $n \times n$.

- A es definida positiva si todos sus autovalores o valores propios son positivos. Se denota $A \succ 0$.
- A es semidefinida positiva si todos sus valores propios son no negativos. Se denota $A \succeq 0$.
- A es definida negativa sí y sólo si $-A \succ 0$. Se denota $A \prec 0$.
- A es semidefinida negativa sí y sólo si $-A \succeq 0$. Se denota $A \preceq 0$.
- Decimos que $A \succ B$ sí y sólo si $A - B \succ 0$. De manera similar se define $A \succeq B$, $A \prec B$ y $A \preceq B$.

Definición 14. Sea $F : \Omega \to \mathbb{R}^{s\times s}$, donde $\Omega \subseteq \mathbb{R}^{n\times m}$. Sean 0 y 0_s los ceros de $\mathbb{R}^{n\times m}$ y $\mathbb{R}^{s\times s}$ respectivamente y $\Omega^* = \Omega - \{0\}$.

- F es definida positiva en Ω, si $F(X) \succ 0_s$ para todo $X \in \Omega^*$. Se denota $F \succ 0_s$.

- F es semidefinida positiva en Ω, si $F(X) \succeq 0_s$ para todo $X \in \Omega$. Se denota $F \succeq 0_s$.
- F es definida negativa en Ω, si $F(X) \prec 0_s$ para todo $X \in \Omega^*$. Se denota $F \prec 0_s$.
- F es semidefinida negativa en Ω, si $F(X) \preceq 0_s$ para todo $X \in \Omega$. Se denota $F \preceq 0_s$.

Definición 15. Sea $F : \mathbb{R}^{n\times m} \to \mathbb{R}^{s\times t}$. F es una función lineal en $\mathbb{R}^{n\times m}$, si $F(\alpha X_1 + \beta X_2) = \alpha F(X_1) + \beta F(X_2)$ para toda α, β constantes (reales o matrices $n \times n$) y X_1, $X_2 \in \mathbb{R}^{n\times m}$.

Definición 16. Sea $G : \mathbb{R}^{n\times m} \to \mathbb{R}^{s\times t}$. G es una función afín en $\mathbb{R}^{n\times m}$, si existe una función lineal $F : \mathbb{R}^{n\times m} \to \mathbb{R}^{s\times t}$ y una matriz constante $C \in \mathbb{R}^{s\times t}$, tal que $G(X) = F(X) + C$.

Definición 17. Sea $F : \mathbb{R}^n \to \mathbb{R}^{m\times m}$ definida como sigue:

$$F(X) := F_0 + \sum_{i=1}^{n} x_i F_i \tag{2.1}$$

Con $X = (x_1, ..., x_n) \in \mathbb{R}^n$ y $F_0, ..., F_n$ matrices $m \times m$ simétricas conocidas. (2.1) es una *Desigualdad Lineal Matricial* DLM, si F satisface una de las siguientes condiciones: $F \succ 0$, $F \succeq 0$, $F \prec 0$, $F \preceq 0$. Las x_i son las variables de optimización o de decisión (pueden ser matrices). La ecuación (2.1) se llama *forma canónica* de una DLM.

En la ecuación (2.1) intervienen $n+1$ matrices F_i, y en aplicaciones prácticas, tal forma es ineficiente por el coste computacional desde el punto de vista de almacenamiento pues se necesita almacenar muchas matrices. por tal razón se han desarrollado métodos para expresar F de una forma más simple (en el sentido de menor número de matrices involucradas), por ejemplo, como una *desigualdad de Lyapunov*[1] $A^TY + YA \prec 0$ que involucra solo una matriz $m \times m$.

Para ilustrar un caso en $\mathbb{R}^3$, sea:

$$F(X) = x_1 \begin{bmatrix} -2 & 2 \\ 2 & 0 \end{bmatrix} + x_2 \begin{bmatrix} 0 & -3 \\ -3 & 4 \end{bmatrix} + x_3 \begin{bmatrix} 0 & 0 \\ 0 & -4 \end{bmatrix} \prec 0, \tag{2.2}$$

La ecuación (2.2) se puede expresar de la forma $A^TY + YA \prec 0$ como

$$F(X) = \begin{bmatrix} -1 & 0 \\ 2 & -2 \end{bmatrix} \begin{bmatrix} x_1 & x_2 \\ x_2 & x_3 \end{bmatrix} + \begin{bmatrix} x_1 & x_2 \\ x_2 & x_3 \end{bmatrix} \begin{bmatrix} -1 & 2 \\ 0 & -2 \end{bmatrix} \prec 0, \tag{2.3}$$

[1] 1857-1918. *Aleksandr Mikhailovich Lyapunov*. Matemático ruso. Introdujo el *método de Lyapunov* en 1899. Ver documento electrónico de Carlos Mario Velez *Estabilidad de Lyapunov* (Universidad EAFIT)

Con $A = \begin{bmatrix} -1 & 2 \\ 0 & -2 \end{bmatrix}$ e $Y = \begin{bmatrix} x_1 & x_2 \\ x_2 & x_3 \end{bmatrix}$.

En el ejemplo se puede observar que hay una reducción del número de matrices involucradas, de modo que trabajar directamente con la forma canónica puede reducir la eficiencia de la DLM.

En general, en una DLM se forma una matriz de bloque donde cada bloque es una función afín de las variables de la matriz. A manera de ilustración, considere la siguiente DLM:

$$N^T \begin{pmatrix} A^T X + XA & XC^T & B \\ CX & -\gamma I & D \\ B^T & D^T & -\gamma I \end{pmatrix} N \prec 0 \tag{2.4}$$

donde A, B, C, D, N son matrices conocidas y las variables son $X = X^T \in \mathbb{R}^{n\times n}$ y $\gamma \in \mathbb{R}$ (Note que el escalar γ se puede considerar como una matriz 1×1). N es llamado *Factor Externo* y no necesita ser cuadrado (a menudo está ausente).

La matriz de bloques

$$L(X, \gamma) = \begin{pmatrix} A^T X + XA & XC^T & B \\ CX & -\gamma I & D \\ B^T & D^T & -\gamma I \end{pmatrix} \tag{2.5}$$

es llamada *Factor Interno*. (2.5) es una matriz simétrica por bloques y su estructura de bloque se caracteriza por los tamaños de sus bloques diagonales. Por simetría, (2.5) esta completamente especificado por los bloques sobre la diagonal. Cada bloque es una función afín de las variables X y γ. De esta forma, (2.5) queda definida por las matrices de bloque.

Por otro lado, a manera de ejemplo, las variables $X = (x_1, x_2, x_3, x_4) \in \mathbb{R}^4$ e $Y = (y_1, y_2, y_3) \in \mathbb{R}^3$ pueden verse como variables matriciales en $\mathbb{R}^3$, gracias a la estructura bloque - diagonal (X_{B-D}) y a la estructura simétrica de Toeplitz (X_T) respectivamente como sigue:

$$X_{B-D} = \left(\begin{array}{c|cc} x_1 & 0 & 0 \\ \hline 0 & x_2 & x_3 \\ 0 & x_3 & x_4 \end{array}\right) \qquad Y_T = \begin{pmatrix} y_1 & y_2 & y_3 \\ y_2 & y_1 & y_2 \\ y_3 & y_2 & y_1 \end{pmatrix}$$

NOTA. El conjunto $\mathcal{F} = \{X \in \mathbb{R}^n \mid F(X) \succ 0\}$ denominado conjunto solución es convexo, es decir, verifica

$$\text{Para cada } X_1,\ X_2 \in \mathcal{F},\ \lambda X_1 + (1-\lambda)X_2 \in \mathcal{F},\ \text{para todo } \lambda \in [0,1] \tag{2.6}$$

Una DLM define un problema convexo sobre la variable X, el cual se puede resolver numéricamente garantizando encontrar una solución, si existe. Un sistema basado en múltiples DLMs puede ser considerado como un conjunto de DLMs simples, lo que permite que se mantenga la condición de convexidad; más aún, La variable de decisión X en una DLM puede ser una matriz.

DLM múltiple. En muchas aplicaciones se pueden combinar DLMs de la forma:

$$\text{Si } F_i(X) \succ 0,\ i = 1, ..., k \ \text{ entonces } \begin{pmatrix} F_1(X) & 0 & \cdots & 0 \\ 0 & F_2(X) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & F_k(X) \end{pmatrix} \succ 0 \qquad (2.7)$$

donde cada $F_i(X)$ tiene la forma (2.1).

Una DLM con restricción lineal se plantea de la siguiente forma:

$$\begin{cases} F(X) \succ 0 \\ AX = b \end{cases} \qquad (2.8)$$

2.2. Problemas DLM genéricos

Los siguientes son algunos problemas que se pueden resolver mediante DLMs, dados en una región $\Omega \subset \mathbb{R}^n$:

a) VIABILIDAD. Comprobar si existe o no solución para $F(X) \succ 0$. La DLM se llama viable si existe $X \in \Omega$ que verifica ésta inecuación. En caso contrario, $F(X) \succ 0$ no es viable.

b) OPTIMIZACIÓN. Sea $F : \mathcal{S} \to \mathbb{R}$, con $\mathcal{S} = \{X \in \mathbb{R}^n \mid F(X) \succ 0\}$. El problema de optimización con restricción DLM es determinar

$$p_{opt} = \inf_{X \in \mathcal{S}} F(X) \ \text{ y/o } \ X_{opt}, \text{ tal que } F(X_{opt}) = p_{opt}. \qquad (2.9)$$

c) PROBLEMA GENERALIZADO DEL VALOR PROPIO. Consiste en calcular el mínimo escalar $\lambda \in \mathbb{R}$ verificando las restricciones:

$$\begin{cases} \lambda Id - A(X) \succ 0 \\ B(X) \succ 0 \end{cases} \qquad (2.10)$$

Siendo $A(X)$ y $B(X)$ simétricas y afines en X y X variable de optimización.

De los anteriores problemas nos interesa estudiar el de optimización y los métodos para resolverlo, para lo cual haremos uso de las herramientas que nos brinda la librería LMI Control Toolbox del programa MatLab.

2.3. Características de las DLM

Éstas son algunas ventajas que obtenemos mediante el uso de DLMs:

* Permiten convertir desigualdades no lineales en desigualdades lineales.
* Una variedad de especificaciones de diseño y restricciones pueden ser expresadas como DLMs.
* Una vez formulado en relación con DLMs, un problema se puede resolver con algoritmos de optimización convexa.

2.4. Solución de problemas DLM

Entre las herramientas para solucionar problemas DLMs esta el LMI Control Toolbox del programa MatLab, plataforma sobre la cual se hace la experimentación del problema de optimización de polinomios basado en la teoría de momentos. Existen otras plataformas que manejan dicha herramienta, como por ejemplo SciLab, software libre. El programa MatLab se basa en métodos de punto interior[6] en dicha herramienta para la solución de los problemas relacionados con desigualdades lineales matriciales.

El paquete LMI del programa Matlab resuelve inecuaciones lineales matriciales de la forma:

$$N^T\mathcal{L}(X_1,...,X_K)N \prec M^T\mathcal{R}(X_1,...,X_K)M \tag{2.11}$$

con

- $X_1,...,X_K$ variables matriciales
- N y M matrices conocidas de igual dimensión
- $\mathcal{L}(\Delta)$ y $\mathcal{R}(\Delta)$ matrices simétricas, estructuradas de la misma manera, cada bloque siendo una función afín de las variables $X_1,...,X_K$ o de sus transpuestas, como en (2.5).

En el presente capítulo se mostró el esquema de una DLM y se mencionaron algunas herramientas desarrolladas para solucionar DLMs (Para mayores detalles ver [7]). En el siguiente capítulo se estudiará el método de optimización de polinomios en varias variables basado en el problema de los momentos y que utiliza desigualdades lineales matriciales para su solución.

Capítulo 3

OPTIMIZACIÓN DE POLINOMIOS MEDIANTE LA TEORÍA DE MOMENTOS

Como se vió al final del capítulo 1, la función de verosimilitud asociada a las reglas de una GIP es un polinomio en varias variables, por lo cual el problema de estimar las probabilidades de las reglas se reduce a optimizar dicho polinomio. En este capítulo se describe el método basado en la teoría de momentos (ver [3], [9]) para la optimización de polinomios en varias variables.

A continuación se desarrolla la teoría y resultados del método de optimización global con polinomios basado en la teoría de momentos. Para ello se hará el análisis del problema de minimizar el polinomio, dado que para el problema de maximización el resultado es equivalente.

Sea $p(x) : \mathbb{R}^n \to \mathbb{R}$ un polinomio. Se define el problema $\mathbb{P}$ que consiste en encontrar el mínimo global de $p(x)$, al que llamamos p^*, notado de la siguiente forma:

$$\mathbb{P} \mapsto p^* := \min_{x \in \mathbb{R}^n} p(x) \tag{3.1}$$

Donde $p^* \in \mathbb{R}$, además encontrar $x^* \in \mathbb{R}^n$ tal que $p(x^*) = p^*$ (Para las GIPs, lo que se necesita es encontrar x^*). Se define $\mathbb{P}_K$ de la misma forma que el problema anterior pero restringido al conjunto compacto $K \subseteq \mathbb{R}^n$ definido por desigualdades polinomiales de la forma $g_i(x) \geq 0$, $i = 1, \ldots, r$:

$$\mathbb{P}_K \mapsto p_K^* := \min_{x \in K} p(x) \tag{3.2}$$

donde $p_K^* \in \mathbb{R}$, además encontrar $x_K^* \in K$ tal que $p(x_K^*) = p_K^*$.

La representación de polinomios en una variable no negativos como suma de cuadrados de polinomios es una herramienta muy útil para calcular valores óptimos, ya que permiten crear algoritmos basados en la teoría de punto interior [6] para calcular un mínimo global. sin embargo, en el caso multivariado no todo polinomio no negativo se puede escribir como suma de cuadrados de polinomios. La estrategia en este caso consiste en reducir el problema (3.2) a una sucesión equivalente de problemas cuadráticos, para que el problema de optimización se convierta en un problema convexo de desigualdades lineales matriciales.

Los problemas $\mathbb{P}$ y $\mathbb{P}_K$ definidos se pueden remplazar respectivamente por los problemas equivalentes:

$$\mathcal{P} \mapsto p^* := \min_{\mu \in \mathcal{P}(\mathbb{R}^n)} \int p(x)\mu(dx) \tag{3.3}$$

$$\mathcal{P}_K \mapsto p_K^* := \min_{\mu \in \mathcal{P}(K)} \int p(x)\mu(dx) \tag{3.4}$$

donde $\mathcal{P}(\mathbb{R}^n)$ (respectivamente $\mathcal{P}(K)$) es el conjunto de Borel de medida finita sobre $\mathbb{R}^n$ (respectivamente sobre K). En efecto; veamos que $\mathcal{P}$ es equivalente a $\mathbb{P}$:

Supongamos que $\min p(x) = p^*$ y sea x^* tal que $p(x^*) = p^*$. Como $p(x) \geq p^*$ tenemos que: $\int p(x)d\mu \geq \int p^* d\mu = p^* \int d\mu = p^*$. Así $\int p(x)d\mu \geq p^*$. Ahora, la medida de probabilidad $\mu^* := \delta_{x^*}$ (Delta de Dirac en x^*) es admisible para $\mathcal{P}$, esto es,

$$\int p(x)d\mu^* = p(x^*) = p^*.$$

Recíprocamente, supongamos que $\min \int p(x)d\mu = p^*$. Entonces $\int p(x)d\mu \geq p^*$ para toda μ, en particular para $\mu = \delta_x$. Luego $\int p(x)d\mu = p(x) \geq p^*$ para cada x.

Si p es un polinomio de grado m, el problema de minimizar p es equivalente a minimizar la combinación lineal de la colección finita de momentos y_α de orden menor o igual a m, en la medida de probabilidad μ. La teoría de momentos nos da condiciones suficientes sobre las variables y_α.

3.1. Preliminares

En adelante, llamaremos $\mathcal{A}_m$ al espacio vectorial de los polinomios $q(x) : \mathbb{R}^n \to \mathbb{R}$ de grado menor o igual que m.

Sean $B = \{1, x_1, x_2, x_3, \ldots, x_n, x_1^2, x_1x_2, x_1x_3, \ldots, x_1x_n, x_2^2, x_2x_3, \ldots, x_2x_n, x_3^2, \ldots, x_n^2, \ldots, x_1^m, x_2^m, x_3^m, \ldots, x_n^m\}$ una base de $\mathcal{A}_m$, $s(m)$ la dimensión de $\mathcal{A}_m$ y $p(x) : \mathbb{R}^n \to \mathbb{R}$ un

polinomio de grado m. Asumimos la siguiente notación para $p(x)$:

$$p(x) = \sum_{\alpha} p_\alpha x^\alpha, \text{ con } x^\alpha := x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n} \text{ y } \sum \alpha_j \leq m \tag{3.5}$$

donde $p = (p_\alpha) \in \mathbb{R}^{s(m)}$ es el vector de coeficientes de $p(x)$ en la base B. De ser necesario, un polinomio de grado m puede ser considerado como un polinomio de grado más alto, digamos r, con vector de coeficientes $p \in \mathbb{R}^{s(r)}$, donde los coeficientes de los monomios de grado más alto que m son cero.

Ejemplo 3. Sea $p(x) : \mathbb{R}^3 \to \mathbb{R}$, el polinomio definido como:

$$p(x_1, x_2, x_3) = 8 + 3x_1 + 2x_3 - 20x_1x_3 + 9x_1^2 + 4x_2^2 - 5x_3^2.$$

Así definido, podemos ver que:

- $\diamond$ $p(x)$ es un polinomio de grado 2 tal que $p(x) \in \mathcal{A}_2$.
- $\diamond$ $B = \{1, x_1, x_2, x_3, x_1^2, x_1x_2, x_1x_3, x_2^2, x_2x_3, x_3^2\}$ es una base de $\mathcal{A}_2$. $\dim B = 10$.
- $\diamond$ $\{(0,0,0), (1,0,0), (0,1,0), (0,0,1), (2,0,0), (1,1,0), (1,0,1), (0,2,0), (0,1,1), (0,0,2)\}$ es el conjunto de los α.
- $\diamond$ $p = (8, 3, 0, 2, 9, 0, -20, 4, 0, -5)$ es el vector de coeficientes.

Definición 18. Sea y el vector de dimensión $s(m)$ definido como $y := (y_\alpha)$, donde

$$y_\alpha = \int x^\alpha \mu(dx), \qquad \mu \in \mathcal{P}(\mathbb{R}^n). \tag{3.6}$$

De esta forma, y_α representa el momento (o el valor esperado) de x^α de orden $\sum \alpha_i$ respecto a la medida de probabilidad μ. Para definir la matriz de momentos $M_m(y)$, se analizará primero para el caso bidimensional. En dicho caso, $M_m(y)$ se define como la matriz de bloques de dimensión $s(m)$:

$$M_m(y) = \begin{bmatrix} M_{0,0}(y) & M_{0,1}(y) & \cdots & M_{0,m}(y) \\ M_{1,0}(y) & M_{1,1}(y) & \cdots & M_{1,m}(y) \\ \vdots & \vdots & \ddots & \vdots \\ M_{m,0}(y) & M_{m,1}(y) & \cdots & M_{m,m}(y) \end{bmatrix} \tag{3.7}$$

tal que cada matriz $M_{i,j}(y)$ con $0 \leq i, j \leq m$ tiene la forma:

$$M_{i,j}(y) = \begin{bmatrix} y_{i+j,0} & y_{i+j-1,1} & \cdots & y_{i,j} \\ y_{i+j-1,1} & y_{i+j-2,2} & \cdots & y_{i-1,j+1} \\ \vdots & \vdots & \ddots & \vdots \\ y_{j,i} & y_{j-1,i+1} & \cdots & y_{0,i+j} \end{bmatrix} \tag{3.8}$$

donde $y_{i,j}$ representa el $(i+j)-$ésimo momento de valor $\int x^i y^j \mu(d(x,y))$ en la medida de probabilidad μ.

Ejemplo 4. Para el caso $n = 2$ y $m = 2$ tenemos:

$$M_2(y) = \left[\begin{array}{c|cc|ccc} 1 & y_{1,0} & y_{0,1} & y_{2,0} & y_{1,1} & y_{0,2} \\ \hline y_{1,0} & y_{2,0} & y_{1,1} & y_{3,0} & y_{2,1} & y_{1,2} \\ y_{0,1} & y_{1,1} & y_{0,2} & y_{2,1} & y_{1,2} & y_{0,3} \\ \hline y_{2,0} & y_{3,0} & y_{2,1} & y_{4,0} & y_{3,1} & y_{2,2} \\ y_{1,1} & y_{2,1} & y_{1,2} & y_{3,1} & y_{2,2} & y_{1,3} \\ y_{0,2} & y_{1,2} & y_{0,3} & y_{2,2} & y_{1,3} & y_{0,4} \end{array}\right] \tag{3.9}$$

Ejemplo 5. Para el caso $n = 2$ y $m = 3$ tenemos:

$$M_3(y) = \left[\begin{array}{c|ccc|ccc|ccc} 1 & y_{1,0} & y_{0,1} & y_{2,0} & y_{1,1} & y_{0,2} & y_{3,0} & y_{2,1} & y_{1,2} & y_{0,3} \\ \hline y_{1,0} & y_{2,0} & y_{1,1} & y_{3,0} & y_{2,1} & y_{1,2} & y_{4,0} & y_{3,1} & y_{2,2} & y_{1,3} \\ y_{0,1} & y_{1,1} & y_{0,2} & y_{2,1} & y_{1,2} & y_{0,3} & y_{3,1} & y_{2,2} & y_{1,3} & y_{0,4} \\ \hline y_{2,0} & y_{3,0} & y_{2,1} & y_{4,0} & y_{3,1} & y_{2,2} & y_{5,0} & y_{4,1} & y_{3,2} & y_{2,3} \\ y_{1,1} & y_{2,1} & y_{1,2} & y_{3,1} & y_{2,2} & y_{1,3} & y_{4,1} & y_{3,2} & y_{2,3} & y_{1,4} \\ y_{0,2} & y_{1,2} & y_{0,3} & y_{2,2} & y_{1,3} & y_{0,4} & y_{3,2} & y_{2,3} & y_{1,4} & y_{0,5} \\ \hline y_{3,0} & y_{4,0} & y_{3,1} & y_{5,0} & y_{4,1} & y_{3,2} & y_{6,0} & y_{5,1} & y_{4,2} & y_{3,3} \\ y_{2,1} & y_{3,1} & y_{2,2} & y_{4,1} & y_{3,2} & y_{2,3} & y_{5,1} & y_{4,2} & y_{3,3} & y_{2,4} \\ y_{1,2} & y_{2,2} & y_{1,3} & y_{3,2} & y_{2,3} & y_{1,4} & y_{4,2} & y_{3,3} & y_{2,4} & y_{1,5} \\ y_{0,3} & y_{1,3} & y_{0,4} & y_{2,3} & y_{1,4} & y_{0,5} & y_{3,3} & y_{2,4} & y_{1,5} & y_{0,6} \end{array}\right] \tag{3.10}$$

Para el caso tridimensional, $M_m(y)$ es definida por bloques $\{M_{i,j,k}(y)\}$, $0 \leq i, j, k \leq m$ de manera similar al caso bidimensional, Para el caso n-dimensional $M_m(y)$ es definida por bloques $\{M_{a_1,a_2,\ldots,a_n}(y)\}$, con $0 \leq a_i \leq n$.

Sea y vector de momentos de orden $s(m)$ definido como en 3.6. Definimos la forma bilineal $\langle \cdot, \cdot \rangle_y : \mathcal{A}_m \times \mathcal{A}_m \to \mathbb{R}$ por:

$$\langle q(x), p(x) \rangle_y = \langle q, M_m(y)p \rangle = \sum_\alpha (qp)_\alpha y_\alpha = \int q(x)p(x)\mu_y(dx) \tag{3.11}$$

Esta forma bilineal también genera una forma semidefinida positiva en $\mathcal{A}_m$ como

$$\langle q(x), q(x) \rangle_y = \sum_\alpha (q^2)_\alpha y_\alpha = \int q^2(x)\mu_y(dx) \geq 0 \tag{3.12}$$

Para todo polinomio $q(x) \in \mathcal{A}_m$. La teoría de momentos identifica la sucesión y con $M_m(y) \succeq 0$ que corresponde a los momentos de la medida de probabilidad μ_y en $\mathbb{R}^n$.

Propocisión 1. *Los problemas $\mathcal{P}_K$ y $\mathbb{P}_K$ son equivalentes, esto es,*

1. $\text{ínf}\, \mathbb{P}_K = \text{ínf}\, \mathcal{P}_K$.

2. *Si x^* es un minimizador global de $\mathbb{P}_K$, entonces $\mu^* := \delta_{x^*}$ es un minimizador global de $\mathcal{P}_K$.*

3. *Si $\mathbb{P}_K$ tiene un minimizador global, entonces para cada solución óptima μ^* de $\mathcal{P}_K$, $p(x) = \min \mathbb{P}_K$, μ^* casi en todos lados.*

4. *Si x^* es el minimizador global de $\mathbb{P}_K$ y es único, entonces $\mu^* := \delta_{x^*}$ es el único minimizador global de $\mathcal{P}_K$*

Demostración:

1. Como para cada $x \in K$, $p(x) = \int p d\delta_x$, se sigue que $\inf \mathcal{P}_K \leq \inf \mathbb{P}_K$. Reciprocamente, suponiendo que $p^* := \inf \mathbb{P}_K > -\infty$. Como $p(x) \geq p^*$ para todo $x \in K$, se sigue que $\int p d\mu^* \geq p^*$ para cada medida de probabilidad μ con soporte contenido en K.[1]
2. Si x^* es un minimizador global de $\mathbb{P}_K$, entonces $\inf \mathbb{P}_K = \min \mathbb{P}_K = p(x^*)$. Luego, por *1.* $\inf \mathcal{P}_K = p(x^*)$. Además, si $\mu^* := \delta_{x^*}$, entonces se tiene que $\int p d\delta_{x^*} = p(x^*)$, por lo cual $\min \mathcal{P}_K = p(x^*)$ y por lo tanto $\mu^* := \delta_{x^*}$ es un minimizador global de $\mathcal{P}_K$.

3. De *2.*, $\mathbb{P}_K$ tiene por lo menos una solución óptima. Para una solución óptima arbitraria μ^*, se tiene $\int p d\mu^* = p^*$ con $p^* = \min \mathbb{P}_K$. Suponiendo que existe un conjunto de Borel $B \subset K$ tal que $\mu^*(B) > 0$ y $p(x) \neq p^*$ en B, es decir, $p(x) > p^*$ en B. Entonces,

$$\int p d\mu^* = \int_B p d\mu^* + \int_{K-B} p d\mu^* > p^*$$

en contradicción con $\int p d\mu^* = p^*$.

4. Esta demostración se sigue de 3.

Observe que si $p(x)$ es un polinomio de grado m, el problema planteado en (3.3) involucra solo los momentos de μ de orden m. Por tanto, ahora remplazamos μ con la sucesión finita $y := \{y_\alpha\}$ de todos los momentos de orden a lo más m, esto es:

$$y_\alpha := \int x^\alpha d\mu, \qquad \sum_{i=1}^{n} \alpha_i = k, \ k = 1, 2, \ldots, m \tag{3.13}$$

[1]Se dice que una función tiene soporte compacto si el conjunto donde no es nula conforma un conjunto cerrado y acotado, el soporte de una función $f(x)$ se define como

$$\sup f = \{x \in \mathbb{R}^n | f(x) \neq 0\}.$$

3.2. Optimización global

Sea $p(x) : \mathbb{R}^n \to \mathbb{R}$ un polinomio de grado $2m$ con vector de coeficientes $p \in \mathbb{R}^{s(m)}$. Puesto que deseamos minimizar $p(x)$, se puede asumir que el término constante desaparece, es decir, $p_0 \equiv 0$. Introduzcamos el siguiente problema de optimización convexa LMI (o Positive Semidefinite Program (PSD)):

$$\mathbb{Q} \longmapsto \begin{cases} \text{ínf}_y \sum_\alpha p_\alpha y_\alpha, \\ M_m(y) \succeq 0, \end{cases} \tag{3.14}$$

o equivalentemente,

$$\mathbb{Q} \longmapsto \begin{cases} \text{ínf}_y \sum_\alpha p_\alpha y_\alpha, \\ \sum_{\alpha \neq 0} y_\alpha B_\alpha \succeq -B_0, \end{cases} \tag{3.15}$$

donde B_0 es la matriz nula y B_α es la matriz cuya entrada α es 1 y el resto 0. El problema dual $\mathbb{Q}^*$ de $\mathbb{Q}$ es el problema convexo LMI definido por:

$$\mathbb{Q}^* \longmapsto \begin{cases} \sup_X \langle X, -B_0 \rangle \, (= -X(1,1)), \\ \langle X, B_\alpha \rangle \; = \; p_\alpha \quad \alpha \neq 0, \\ X \succeq 0, \end{cases} \tag{3.16}$$

donde X es una matriz simétrica de valores reales y $\langle A, B \rangle$ es producto interno usual dado por la traza de AB para matrices simétricas de valores reales. $(= -X(1,1))$ significa que $\sup_X \langle X, -B_0 \rangle$ es almacenado en $X(1,1)$ con signo opuesto.

Propocisión 2. *Supongamos que $\mathbb{Q}^*$ tiene una solución factible. Entonces, $\mathbb{Q}$ es soluble y se tiene que:*

$$\text{ínf}\, \mathbb{Q} = \text{máx}\, \mathbb{Q}^* \tag{3.17}$$

Demostración:

El resultado se sigue de la teoria de dualidad de programación convexa si se puede probar que existe una solución viable y de $\mathbb{Q}$ con $M_m(y) \succ 0$. Sea μ una medida de probabilidad de $\mathbb{R}^n$ con densidad f estrictamente positiva con respecto a la medida de Lebesgue[2] y con todos sus momentos finitos, esto es, μ es tal que:

$$y_\alpha = \int x^\alpha d\mu < \infty$$

Para cada combinación $\alpha_1 + \alpha_2 + \ldots + \alpha_n = r$, $r = 0, 1, 2, \ldots$. Entonces la matriz $M_m(y)$, con y definida como antes, es tal que $M_m(y) \succ 0$. Efectivamente, para cada polinomio

[2]Para más información sobre la medida de Lebesgue, ver [12] pag. 12

$q(x) : \mathbb{R}^m \to \mathbb{R}$, se tiene:

$$\begin{aligned} \langle q(x), q(x) \rangle_y = \langle q, M_m(y) q \rangle &= \int q^2(x) \mu(dx) \quad \text{(por (3.12))} \\ &= \int q^2(x) f(x) dx \\ &> 0 \quad \text{siempre que } q \neq 0 \text{ (cuando } f > 0). \end{aligned}$$

Por tanto, y es viable para $\mathbb{Q}$ y $M_m(y) \succ 0$, el resultado deseado.

Teorema 2. *Sea $p(x) : \mathbb{R}^n \to \mathbb{R}$ un polinomio de grado $2m$ definido como en (3.5) con mínimo global $p^* = \min \mathbb{P}$.*

1. *Si el polinomio no negativo $p(x) - p^*$ es suma de cuadrados de polinomios, $\mathbb{P}$ es equivalente al problema LMI convexo $\mathbb{Q}$ definido en (3.1). Esto es $\min \mathbb{Q} = \min \mathbb{P}$ y si x^* es un minimizador global de $\mathbb{P}$, entonces el vector*

$$y^* = (x_1^*, ..., x_n^*, (x_1^*)^2, x_1^* x_2^*, ..., (x_1^*)^{2m}, ..., (x_n^*)^{2m}) \tag{3.18}$$

   es un minimizador de $\mathbb{Q}$.

2. *Inversamente, Si $\mathbb{Q}^*$ tiene una solución factible, entonces $\min \mathbb{P} = \min \mathbb{Q}$ sí y sólo si $p(x) - p^*$ es suma de cuadrados.*

Demostración:

1. Sea $p(x) - p^*$ una suma de cuadrados de polinomios. Esto es:

$$p(x) - p^* = \sum_{i=1}^{r} q_i^2(x) \qquad x \in \mathbb{R}^n \tag{3.19}$$

para algunos polinomios $q_i(x) : \mathbb{R}^n \to \mathbb{R}$ con vector de coeficientes $q_i \in \mathbb{R}^{s(m)}$, $i = 1, 2, \ldots, r$. Equivalentemente:

$$p(x) - p^* = \langle X, M_m(y) \rangle . \qquad x \in \mathbb{R}^n \tag{3.20}$$

con $X = \sum_1^r q_i q_i'$ e $y = (x_1, \ldots, (x_1)^{2m}, \ldots, (x_n)^{2m})$. Pero de (3.20) y

$$p(x) - p^* = -p^* + \sum_{\alpha} p_\alpha x^\alpha$$

se sigue que

$$X(1,1) = -p^* \text{ y } \langle X, B_\alpha \rangle = p_\alpha \text{ para todo } \alpha \neq 0$$

de modo que (como $X \succeq 0$) X es viable para $\mathbb{Q}^*$ con valor $-X(1,1) = p^*$. Ahora, observe que y^* en (3.18) es viable para $\mathbb{Q}$ con valor p^* de modo que $\min \mathbb{Q} = \max \mathbb{Q}^*$ e y^* y X son soluciones óptimas de $\mathbb{Q}$ y $\mathbb{Q}^*$ respectivamente.

2. Asumimos que $\mathbb{Q}^*$ tiene una solución factible y que $\min \mathbb{P} = \min \mathbb{Q}$. Entonces, de la proposición 3, $\mathbb{Q}^*$ es soluble y no hay intervalo de dualidad, es decir, $\max \mathbb{Q}^* = \inf \mathbb{Q} = \min \mathbb{Q}$. Sea X^* una solución óptima de $\mathbb{Q}^*$, se garantiza la existencia. Escribimos

$$X^* = \sum_{i=1}^{r} \lambda_i q_i q_i'$$

con los q_i's vectores propios de X^* correspondientes a los valores propios positivos λ_i, $i = 1, \ldots, r$.

Como $\lambda^* := \max \mathbb{Q}^* = \min \mathbb{Q}$ y $\min \mathbb{Q} = \min \mathbb{P}$. Sea y^* como en (3.18) una solución óptima de $\mathbb{Q}$. A partir de la optimización tanto de X^* e y^*, se debe tener

$$\langle X^*, M_m(y^*) \rangle = 0$$

Equivalentemente,

$$0 = \sum_{i=1}^{r} \lambda_i \langle q_i, M_m(y^*) q_i \rangle = \sum_{i=1}^{r} \lambda_i q_i^2(x^*)$$

Para un $x \in \mathbb{R}^n$ arbitrario, sea

$$y := (x_1, \ldots, x_n, x_1^2, x_1 x_2, \ldots, x_1^{2m}, \ldots, x_n^{2m})$$

de la misma manera como se hizo para x^*

$$\langle X^*, M_m(y^*) \rangle = \sum_{i=1}^{r} \lambda_i q_i^2(x)$$

Por otro lado

$$\begin{aligned} \langle X^*, M_m(y^*) \rangle &= \lambda^* + \sum_{\alpha \neq 0} \langle X^*, B_\alpha \rangle \\ &= \lambda^* + \sum_{\alpha \neq 0} p_\alpha y_\alpha \\ &= \lambda^* + p(x) \end{aligned}$$

por consiguiente, como X^* es óptimo, $-X(1,1) = -\lambda^* = p^*$ y se obtiene

$$\sum_{i=1}^{r} \lambda_i q_i^2(x) = p(x) - p^*$$

El resultado deseado.

Corolario 1. *Sea $p(x) : \mathbb{R}^n \mapsto \mathbb{R}$ un polinomio de valor real de grado $2m$. Asumimos que $\mathbb{Q}^*$ tiene una solución factible. Entonces,*

$$p(x) - p^* = \sum_{i=1}^{r} q_i^2(x) - [\text{mín}\,\mathbb{P} - \text{ínf}\,\mathbb{Q}] \tag{3.21}$$

para cada polinomio $q_i(x) : \mathbb{R}^n \mapsto \mathbb{R}$ de grado a lo más m, con $i = 1, 2, .., r$. De esta forma, $\text{ínf}\,\mathbb{Q}$ es un valor cercano a p^*. Así podemos escribir $p(x) - p^*$ como una suma de cuadrados de polinomios sobre alguna constante siempre que $\mathbb{Q}^*$ tenga una solución factible.

3.2.1. Caso general

El siguiente es un resultado válido cuando $p(x) - p^*$ no necesariamente es suma de cuadrados, para lo cual se da primero la siguiente notación.

Sea $q(x) : \mathbb{R}^n \mapsto \mathbb{R}$ un polinomio de grado w con vector de coeficientes $q \in \mathbb{R}^{s(w)}$. Si la entrada i, j de $M_m(y)$ es y_β, sea $\beta(i, j)$ el subíndice β de y_β. Sea $M_m(qy)$ la matriz definida por:

$$M_m(qy)(i, j) = \sum_{\alpha} q_\alpha y_{\{\beta(i,j)+\alpha\}} \tag{3.22}$$

Ejemplo 6. Si $M_1(y) = \begin{bmatrix} 1 & y_{10} & y_{01} \\ y_{10} & y_{20} & y_{11} \\ y_{01} & y_{11} & y_{02} \end{bmatrix}$ y el polinomio $q(x) = a - x_1^2 - x_2^2$, de la ecuación (3.22) tenemos:

$$M_1(qy) = \begin{bmatrix} a - y_{20} - y_{02} & ay_{10} - y_{30} - y_{12} & ay_{01} - y_{21} - y_{03} \\ ay_{10} - y_{30} - y_{12} & ay_{20} - y_{40} - y_{22} & ay_{11} - y_{31} - y_{13} \\ ay_{01} - y_{21} - y_{03} & ay_{11} - y_{31} - y_{13} & ay_{02} - y_{22} - y_{04} \end{bmatrix}$$

Sea $\{y_\alpha\}$ un vector de tamaño $s(2m)$, vector de momentos de orden $\leq 2m$ sobre la medida de probabilidad μ_y definida en $\mathbb{R}^n$. Entonces, para cada polinomio $v(x) : \mathbb{R}^n \mapsto \mathbb{R}$, de grado menor o igual a m, con vector de coeficientes $v \in \mathbb{R}^{s(2m)}$, se define:

$$\langle v, M_m(qy)v \rangle = \int q(x)v^2(x)\mu_y(dx) \tag{3.23}$$

Por tanto, con $K_q := \{x \in \mathbb{R}^n \| q(x) \geq 0\}$, si μ_y tiene su soporte contenido en K_q, entonces se sigue de (3.23) que $M_m(y) \succeq 0$.

Si un minimizador global x^* de $p(x)$ tiene norma mayor o igual que a, para algún $a \geq 0$ ($p(x^*) = p^* = \text{mín}\,\mathbb{P}$ y $\|x^*\| \geq a$) entonces, para $x \in \mathbb{R}^n$, si $\theta(x) = a^2 - \|x\|^2$,

$p(x) - p^* \geq 0$ en $K_\alpha := \{\theta(x) \geq 0\}$. Usamos el hecho de que cada polinomio $p(x)$ estrictamente positivo sobre K_α se puede escribir como:

$$p(x) = \sum_{i=1}^{r_1} q_i^2(x) + \theta(x) \sum_{j=1}^{r_2} t_j^2(x), \tag{3.24}$$

para algunos polinomios $q_i(x)$, $t_j(x)$ con $i = 1, ..., r_1$, $j = 1, ..., r_2$. (Ver [3] pag. 119)

Para cada $N \geq m$, sea $\mathbb{Q}_a^N$ el problema LMI convexo:

$$\mathbb{Q}_a^N \mapsto \begin{cases} \text{ínf}_y \sum_\alpha p_\alpha y_\alpha \\ M_N(y) \succeq 0, \\ M_{N-1}(\theta y) \succeq 0. \end{cases} \tag{3.25}$$

Escribiendo

$$M_{N-1}(\theta y) = \sum_\alpha y_\alpha C_\alpha$$

para matrices apropiadas $\{C_\alpha\}$, el dual de $\mathbb{Q}_a^N$ es el problema LMI convexo

$$(\mathbb{Q}_a^N)^* \mapsto \begin{cases} \sup_{X,Z\succeq 0} -X(1,1) - a^2 Z(1,1), \\ \langle X, B_\alpha \rangle + \langle Z, C_\alpha \rangle = p_\alpha, \qquad \alpha \neq 0. \end{cases} \tag{3.26}$$

Teorema 3. *Sea $p(x) : \mathbb{R}^n \to \mathbb{R}$ un polinomio definido como en (3.5), con mínimo global $p^* = \min \mathbb{P}$ y tal que $||x^*|| \leq a$ para algún $a > 0$, donde x^* es minimizador. Entonces:*

1. *Cuando $N \to \infty$ se tiene* $\text{ínf}\, \mathbb{Q}_a^N \uparrow p^*$.

2. $\min \mathbb{Q}_a^N = p^*$ *sí y sólo si la ecuación (3.24) es válida. En este caso el vector*

$$y^* = (x_1^*, \ldots, x_n^*, (x_1^*)^2, x_1^* x_2^*, \ldots, (x_1^*)^{2N}, \ldots, (x_n^*)^{2N}) \tag{3.27}$$

es un minimizador de $\mathbb{Q}_a^N$. Además, $\max(\mathbb{Q}_a^N)^* = \min \mathbb{Q}_a^N$ *y para cada solución óptima (X^*, Z^*) de $(\mathbb{Q}_a^N)^*$*

$$p(x) - p^* = \sum_{i=1}^{r_1} \lambda_i q_i^2(x) + \theta(x) \sum_{j=1}^{r_2} \gamma_j t_j^2(x) \tag{3.28}$$

donde X^* y Z^* son los vectores de coeficientes de los polinomios $q_i(x)$ y $t_j(x)$ respectivamente ($X^* = (\lambda_1, \ldots, \lambda_{r_1})$ e $Z^* = (\gamma_1, \ldots, \gamma_{r_2})$).

Demostración:

1. Dado que $x^* \in K_\alpha$ y como

$$y^* := (x_1^*, \ldots, (x_1^*)^{2N}, \ldots, (x_n^*)^{2N})$$

Se sigue que $M_N(y^*) \succeq 0$ y $M_{N-1}(\theta y^*) \succeq 0$. Luego, y^* es admisible para $\mathbb{Q}_a^N$ y por tanto ínf $\mathbb{Q}_a^N \leq p^*$.

Ahora, sea $\epsilon > 0$ arbitrario. Entonces, $p(x) - (p^* - \epsilon) > 0$ y por lo tanto, existe N_0 tal que

$$p(x) - p^* + \epsilon = \sum_{i=1}^{r_1} q_i^2(x) + \theta(x) \sum_{j=1}^{r_2} t_j^2(x)$$

Para algunos polinomios $q_i(x)$, $i = 1, 2, \ldots, r_1$ de grado a lo más N_0 y $t_j(x)$, $j = 1, 2, \ldots, r_2$ de grado a lo más $N_0 - 1$. Sean $q_i \in \mathbb{R}^{s(N_0)}$ y $t_j \in \mathbb{R}^{s(N_0-1)}$ los vectores de coeficientes de los polinomios $q_i(x)$ y $t_j(x)$ respectivamente. sean

$$X := \sum_{i=1}^{r_1} q_i q_i' \quad Z := \sum_{j=1}^{r_2} t_j t_j'$$

tal que $X \succeq 0$ y $Z \succeq 0$. Se puede verificar que (X, Z) es admisible para $(\mathbb{Q}_a^{N_0})^*$ con valor $-X(1,1) - a^2 Z(1,1) = (p^* - \epsilon)$. De la dualidad débil se sigue que

$$\text{ínf}\, \mathbb{Q}_a^{N_0} \geq -(X(1,1) + a^2 Z(1,1)) = p^* - \epsilon$$

de donde

$$p^* - \epsilon \leq \text{ínf}\, \mathbb{Q}_a^{N_0} \leq p^*$$

Ahora, se prueba que no existe mucha diferencia respecto a la dualidad entre $\mathbb{Q}_a^{N_0}$ y su dual $(\mathbb{Q}_a^{N_0})^*$ cuando $N \geq N_0$. Efectivamente, sea μ una medida de probabilidad con distribución uniforme en K_α. Sea $y_\mu = \{y_\alpha\}$ con

$$y_\alpha := \int x^\alpha \mu(dx)$$

para todas las combinaciones $(\alpha_1, \ldots, \alpha_n) = r$, $r = 1, \ldots, N$. Todos los y_α's están bien definidos, entonces μ tiene su soporte contenido en el conjunto compacto K_α. De (3.11),

$$\langle q, M_N(y_\mu) q \rangle = \int q^2(x) \mu(dx) > 0 \quad \text{siempre que } q \in \mathbb{R}^{s(N)} \text{ y } q \neq 0$$

y de (3.23),

$$\langle q, M_{N-1}(\theta y_\mu) q \rangle = \int \theta(x) q^2(x) \mu(dx) > 0 \quad \text{siempre que } q \in \mathbb{R}^{s(N-1)} \text{ y } q \neq 0.$$

Se sigue que $M_N(y_\mu) \succeq 0$ y $M_{N-1}(\theta y_\mu) \succ 0$; esto es, y_μ es estrictamente admisible por $\mathbb{Q}_a^N$ y como $(\mathbb{Q}_a^N)^*$ tiene una solución admisible, de un resultado básico de optimización convexa, no existe mucha diferencia respecto a la dualidad entre $\mathbb{Q}_a^N$ y $(\mathbb{Q}_a^N)^*$. Además, $(\mathbb{Q}_a^N)^*$ es soluble, esto es,

$$\sup(\mathbb{Q}_a^N)^* = \text{máx}(\mathbb{Q}_a^N)^*$$

Del hecho de que $M_N(y) \succ 0$ implica $M_{N'}(y) \succ 0$ para cada $N \geq N'$ (tal que $M_{N'}(y) \succ 0$ es una submatriz de $M_N(y) \succ 0$ y similarmente para $M_{N-1}(\theta y)$), se sigue que ínf $\mathbb{Q}_a^N \uparrow p^*$. Por lo tanto, para cada solución y de $\mathbb{Q}_a^N$, el vector truncado adecuado y' es admisible para $\mathbb{Q}_a^{N'}$, siempre que $N' \leq N$, con el mismo valor. Por tanto,

$$\text{ínf}\, \mathbb{Q}_a^N \geq \text{ínf}\, \mathbb{Q}_a^{N'} \quad \text{siempre que} \quad N \geq N'.$$

2. $\Rightarrow$*)* y^* definido como en (3.27), es un minimizador de $\mathbb{Q}_a^N$. De *1.* se sabe que no hay dualidad posible entre $\mathbb{Q}_a^N$ y $(\mathbb{Q}_a^N)^*$ para N suficientemente grande, y $(\mathbb{Q}_a^N)^*$ es soluble. Por lo tanto, para N suficientemente grande, sea (X^*, Z^*) una solución óptima de $(\mathbb{Q}_a^N)^*$ (Se garantiza su existencia).

Como $X^* \succeq 0$, $Z^* \succeq 0$ se escribe

$$X^* = \sum_{i=1}^{r_1} \lambda_i q_i q_i' \qquad Z^* = \sum_{j=1}^{r_2} \gamma_j t_j t_j',$$

donde los q_i's (respectivamente los t_j's) son los multivectores de X^* (respectivamente, Z^*) con multivalores λ_i (respectivamente γ_j). Con

$$y = (x_1, \ldots, x_n, x_1^2, x_1x_2, \ldots, (x_1)^{2N}, \ldots, (x_n)^{2N})$$

se tiene

$$\begin{aligned} \langle X^*, M_N(y)\rangle + \langle Z^*, M_{N-1}(\theta y)\rangle &= X^*(1,1) + a^2 Z^*(1,1) \\ &\quad + \sum_{\alpha \neq 0} y_\alpha \left[\langle X^*, B_\alpha\rangle + \langle Z^*, C_\alpha\rangle\right] \\ &= X^*(1,1) + a^2 Z^*(1,1) + p(x) \\ &= p(x) - p^* \end{aligned}$$

De donde sigue la última igualdad

$$\text{mín}\, \mathbb{Q}_a^N = p^* = \text{máx}(\mathbb{Q}_a^N)^* = -X^*(1,1) - a^2 Z^*(1,1).$$

Por otro lado,

$$\langle X^*, M_N(y)\rangle = \sum_{i=1}^{r_1} \lambda_i \langle q_i, M_N(y) q_i\rangle = \sum_{i=1}^{r_1} \lambda_i q_i^2(x)$$

y

$$\langle Z^*, M_{N-1}(\theta y)\rangle = \sum_{j=1}^{r_2} \gamma_j \langle t_j, M_{N-1}(\theta y) t_j\rangle = \sum_{j=1}^{r_2} \gamma_j t_j^2(x).$$

Por lo tanto,

$$p(x) - p^* = \sum_{i=1}^{r_1} \lambda_i q_i^2(x) + \sum_{j=1}^{r_2} \gamma_j t_j^2(x)$$

lo que se quería probar.

2. $\Leftarrow$) Si se tiene (3.28) entonces se demuestra como en *1.* (pero con $\epsilon = 0$) que $\sup(\mathbb{Q}_a^N)^* \geq p^*$, así que, a decir verdad, $\max(\mathbb{Q}_a^N)^* = p^* = \min \mathbb{Q}_a^N$ para N suficientemente grande.

3.2.2. Caso restringido

Consideramos el problema (3.2) donde

- $p(x) : \mathbb{R}^n \mapsto \mathbb{R}$ es un polinomio de valor real de grado a lo más m.
- K es un conjunto compacto definido por las desigualdades de polinomios $g_i(x) \geq 0$ con $g_i(x) : \mathbb{R}^n \mapsto \mathbb{R}$ polinomio de valor real de grado a lo más w_i, $i = 1, 2, ..., r$.

Definición 19. Sea $\widetilde{w}_i := \lceil w_i/2 \rceil$ el menor entero mayor que $w_i/2$, y con $N \geq \lceil m/2 \rceil$ y $N \geq \max_i \widetilde{w}_i$. Consideremos el problema LMI convexo

$$\mathbb{Q}_K^N \mapsto \begin{cases} \inf_y \sum_\alpha p_\alpha y_\alpha \\ M_N(y) \succeq 0, \\ M_{N-\widetilde{w}_i}(g_i y) \succeq 0, \quad i = 1, ..., r. \end{cases} \tag{3.29}$$

Escribiendo $M_{N-\widetilde{w}_i}(g_i y) = \sum C_{i\alpha} y\alpha$, para matrices simétricas apropiadas $\{C_{i\alpha}\}$, el dual de $\mathbb{Q}_K^N$ es el problema LMI convexo

$$(\mathbb{Q}_K^N)^* \mapsto \begin{cases} \sup_{X, Z_i} -X(1,1) - \sum_{i=1}^r g_i(0) Z_i(1,1), \\ \langle X, B_\alpha\rangle + \sum_{i=1}^r \langle Z_i, C_{i\alpha}\rangle = p_\alpha, \qquad \alpha \neq 0, \\ X, Z_i \succeq 0, i = 1, ..., r. \end{cases} \tag{3.30}$$

Nota: El siguiente teorema es el equivalente al teorema para el caso general.

Teorema 4. *Sea $p(x) : \mathbb{R}^n \mapsto \mathbb{R}$ un polinomio de grado m y K el conjunto compacto $\{g_i(x) \geq 0, i = 1, ..., r\}$ y sea*

$$p_K^* := \min_{x \in K} p(x)$$

y tal que $\|x^*\| \leq a$ para algún $a > 0$ en algún minimizador global x^*. Entonces:

1. *si* $N \to \infty$

$$\text{ínf}\, \mathbb{Q}_K^N \uparrow p_K^*.$$

Más aún, para N *suficientemente grande, no existe diferencia respecto a la dualidad entre* Q_K^N *y su dual* $(Q_K^N)^*$ *si el interior de* K *es distinto de vacío.*

2. *Si* $p(x) - p_K^*$ *se puede representar como*

$$p(x) - p_K^* = q(x) + \sum_{i=1}^{r} g_i(x) t_i(x) \tag{3.31}$$

para algún polinomio $q(x)$ *de grado menor o igual que* $2N$*, y algunos polinomios* $t_i(x)$ *de grado menor o igual que* $2N$*,* $i = 1, ..., r$*, todos sumas de cuadrados, entonces*

$$\text{mín}\, Q_K^N = p_K^* = \text{máx}\, (Q_K^N)^*$$

y el vector y^* *definido como en (3.27) es un minimizador global de* Q_K^N*. Además, para cada solución óptima* $(X^*, Z_1^*, ..., Z_r^*)$ *de* $(Q_K^N)^*$*,*

$$p(x) - p_K^* = \sum_{i=1}^{r_0} \lambda_i q_i(x)^2 + \sum_{i=1}^{r} g_i(x) \sum_{j=1}^{r_i} \gamma_{ij} t_{ij}(x)^2 \tag{3.32}$$

donde X^* *y* Z_i^* *son los vectores de coeficientes de los polinomios* $q_i(x)$ *y* $t_{ij}(x)$ *respectivamente (*$X^* = (\lambda_1, \ldots, \lambda_{r_1})$ *y* $Z_i^* = (\gamma_{i1}, \ldots, \gamma_{ir_i})$*).*

En el siguiente ejemplo se ilustra el resultado anterior:

Ejemplo 7. Sea $p(x) : \mathbb{R}^2 \to \mathbb{R}$ definido como

$$p(x) = -a_1 x_1^2 - a_2 x_2^2,\ a_1, a_2 > 0$$

y sea K el conjunto compacto

$$K := \{(x_1, x_2) \in \mathbb{R}^2 | x_1 + x_2 \leq b_1; a x_1 + x_2 \leq b_2; x_1, x_2 \geq 0\}$$

De esta forma, por el teorema anterior tenemos que:

$$p(x) - p^* = q(x) + (b_1 - x_1 - x_2) t_1(x) + (b_2 - a x_1 - x_2) t_2(x) + x_1 t_3(x) + x_2 t_4(x)$$

para algún polinomio $q(x)$ de grado 4 y algunos polinomios $t_i(x)$, i=1,. . . ,4 todos suma de cuadrados.

A continuación se ilustra otro ejemplo:

Ejemplo 8. Sea $p(x) : \mathbb{R}^2 \to \mathbb{R}$ un polinomio definido como sigue

$$p(x) = -(x_1 - 1)^2 - (x_1 - x_2)^2 - (x_2 - 3)^2$$

bajo las siguientes restricciones:

- $1 - (x_1 - 1)^2 \leq 0$
- $1 - (x_1 - x_2)^2 \leq 0$
- $1 - (x_2 - 3)^2 \leq 0$

y sea K el conjunto compacto

$$K := \{(x_1, x_2) \in \mathbb{R}^2 | 1 - (x_1 - 1)^2 \leq 0; 1 - (x_1 - x_2)^2 \leq 0; 1 - (x_2 - 3)^2 \leq 0\}$$

Aquí el punto $(1, 2)$ es un minimizador global con valor óptimo -2. Resolviendo $\mathbb{Q}_K^1$, esto es, con $N = 1$ y $\tilde{p}(x) = p(x) + 10$ (dado que el término constante -10 debe ser eliminado), resulta un valor óptimo de 7 en lugar del valor deseado 8. En la siguiente iteración, resolviendo $\mathbb{Q}_K^2$ se obtiene un valor óptimo de 8.00017 y un minimizador global aproximado (1.0043,2.0006) (el error 0.00017 es probable debido al uso del método de punto interior).

En el presente capítulo se ha explicado el método de optimización de polinomios en varias variables mediante la teoría de momentos, haciéndose un pequeño estudio de los casos global y restringido. La siguiente etapa del proyecto corresponde al desarrollo computacional del método de optimización de polinomios visto en éste capítulo.

Capítulo 4

EXPERIMENTACIÓN

4.1. Introducción

En el presente capítulo se tratará la parte computacional del método de optimización de polinomios en varias variables mediante la teoría de momentos, vista en el capítulo anterior. Aqui se probará el software desarrollado por los franceses Didier Henrion y Jean Bernard Lasserre. Dichos autores le dieron a la herramienta el nombre de *GLOPTIPOLY* (*GLobal OPTImization of POLYnomials*), creada como una función bajo la plataforma *MATLAB* y que requiere de otra herramienta llamada *SeDuMi* (Versión 1.1), la cual es un complemento para MATLAB. Dichos autores han creado varias versiones mejoradas de la herramienta, de las cuales se analizará la versión 3.4.

4.2. Metodología

Para la parte experimental, se tomarán ejemplos de libros de cálculo vectorial, problemas típicos de la literatura de optimización y funciones de verosimilitud de algunas GIPs, para realizar pruebas con el software y comparar los resultados con los obtenidos matemáticamente, en los casos en que sea posible.

4.2.1. Herramientas

Matlab

Es la plataforma sobre la cual han sido diseñadas las demás herramientas. La versión utilizada es la 7,0

Gloptipoly

Gloptipoly (GLobal OPTImization of POLYnomials) es una herramienta para MatLab/SeDuMi diseñada para resolver el problema planteado en la ecuación (3.1), se presenta como una solución al problema de optimización de un polinomio dado en varias variables y se basa en la teoría de momentos. Gloptipoly permite modelar un problema de optimización como un problema de momentos.

Algunas características principales por las cuales se hace uso de la versión 3 de Gloptipoly son:

- ⋆ El uso de objetos de tipo polinomio y la programación orientada a objetos con clases específicas para polinomios en varias variables, las medidas, los momentos, y la correspondiente sobrecarga de los operadores.
- ⋆ Lleva a cabo sustituciones explícitas de momentos para reducir el número de variables y las restricciones.
- ⋆ Los problemas que se planteen mediante la teoría de momentos se pueden resolver numéricamente con cualquier herramienta de programación Semidefinida (SDP), por ejemplo SeDuMi y/o Yalmip.
- ⋆ Respecto a versiones anteriores de Gloptipoly, la versión 3 es más estable y soluciona muchos problemas de optimización que no podían ser resueltos por sus antecesoras.

La estructura de la versión 3.4 de Gloptipoly tiene el siguiente modelo:

- ∗ Crear las variables del polinomio. (`mpol`)
- ∗ Definir el polinomio a optimizar. (`f`)
- ∗ Definir las restricciones del polinomio.
- ∗ Definir el problema de optimización. (`msdp`)
- ∗ Resolver el problema de optimización. (`msol`)
- ∗ extraer el(los) punto(s) solución. (`double`)

Donde `mpol`, `msdp` y `msol` son funciones de Gloptipoly 3.0.

`mpol`: es una clase mediante la cual se generan los polinomios multivariados. Las variables, monomios y polinomios son definidos como objetos de la clase `mpol`. Primero se definen las variables, luego el polinomio y por último las restricciones. Por ejemplo:

```
>> mpol x1 x2 x3              % Define variables tipo objeto de clase mpol
>> f0 = x1^2 + x2^2 + x3^2    % Define polinomio tipo objeto de clase mpol

Polinomio escalar

x1^2+x2^2+x3^2

>> x1 + x2 == 0              % Restricción
Medida escalar soportada por igualdad
x1+x2 == 0
>> x3 <= 4                    % Restricción
Medida escalar soportada por desigualdad
x3 <= 4
```

Las variables del polinomio también pueden definirse mediante la instrucción `mpol x k`, donde k es el número de variables a definir, por ejemplo, la instrucción `mpol x 4` define las variables `x(1)`, `x(2)`, `x(3)`y `x(4)` como objetos de la clase `mpol`; de manera similar a como lo hace mediante la instrucción `mpol x1 x2 x3 x4`.

`msdp`: esta función genera el problema de momentos de tipo SDP (SemiDefinite Program). Gloptipoly puede manipular y resolver los Problemas Generalizados de Momentos (GPM), tal como se define a continuación:

$$\begin{aligned} \min_{d\mu}(o\,\text{máx}) \quad & \sum_k \int_{\mathbb{K}_k} f_k(x) d\mu_k(x) \\ \text{Bajo la condición} \quad & \sum_k \int_{\mathbb{K}_k} h_{jk}(x) d\mu_k(x) \geq (o =) b_j;\ j = 1, 2, \ldots \end{aligned}$$

donde las medidas $d\mu_k$ se soportan en conjuntos semialgebraicos básicos

$$\mathbb{K}_k = \{x \in \mathbb{R}^{n_k} : g_{ik}(x) \geq 0, i = 1, 2, \ldots\}$$

En las notaciones, se dan polinomios reales $g_{ik}(x)$, $h_{jk}(x)$ y una constante real b_j. Las variables de decisión en el GPM son las medidas $d\mu_k(x)$, y GloptiPoly permite optimizar más de ellos a través de sus momentos

$$y_{\alpha_k} = \int_{\mathbb{K}_k} x^{\alpha_k} d\mu_k(x),\ \alpha_k \in \mathbb{N}^{n_k}$$

donde los α_k son índices múltiples.

El modelo es el siguiente: `P = msdp(max(f))` ó `P = msdp(min(f))`, donde `f` es el polinomio a optimizar. Para el ejemplo anterior:

```
   >> P = msdp(min(f0))       % Define el problema de Momentos SDP
   GloptiPoly 3.4 del 30 de Septiembre de 2008
Define el problema de momentos SDP
  Función objetivo válida
  Número de restricciones = 0
  Número de momentos de restricciones = 0
Medida 1
  Grado = 2
  Variables = 3
  Momentos = 10
Orden de relajación = 1
Masa de medida 1 a un conjunto
Número total de momentos = 10
Realizar sustituciones de Momentos
Número de momentos después de sustituciones = 9
Generar momentos y las restricciones de soporte
Generar el problema de momentos SDP

Problema de Momentos SDP
  Medidas etiquetadas       = 1
  Ordenes de relajación     = 1
  Variables de Decisión     = 9
  Desigualdades SD          = 4x4
```

`msol`: función diseñada para buscar una solución o soluciones al problema de momentos generado mediante la función `msdp`. Su estructura es: `[status,obj] = msol(P)`, donde `status` sirve para saber si el problema tiene solución y `obj` es el valor óptimo, siempre que la función encuentre solución.

La bandera `status` puede tomar los siguientes valores:

-1 : si el problema de momentos SDP no es viable o no se puede resolver.

0 : si el problema de momentos SDP puede resolverse pero la optimalidad global no se puede detectar. Esta es necesaria para extraer los puntos donde se obtienen las medidas M.

1 : si el problema de momentos SDP puede resolverse, la optimalidad global está certificada y los puntos de apoyo se extraen de las medidas M, en cuyo caso `obj` es el óptimo global del problema de optimización de momentos original. En este caso, el optimizador global se pueden extraer con la instrucción `double(M)`.

para el ejemplo anterior:

```
>> [status.obj] = msol(P)
GloptiPoly 3.4 del 30 de Septiembre de 2008
```

```
Resolver problema de Momentos SDP
*************************************************
Llamando a SeDuMi
SeDuMi 1.1R3 by AdvOL, 2006 and Jos F. Sturm, 1998-2003.
Alg = 2: xz-corrector, Adaptive Step-Differentiation, theta = 0.250, beta = 0.500
eqs m = 9, orden n = 5, dim = 17, bloques = 2
nnz(A) = 9 + 0, nnz(ADA) = 81, nnz(L) = 45
 it :     b*y       gap    delta  rata   t/tP*  t/tD*   feas cg cg  prec
  0 :            4.05E+000 0.000
  1 :  1.07E-001 1.60E-001 0.000 0.0396 0.9900 0.9900   2.00  1  1  7.2E-002
  2 : -3.32E-006 1.15E-005 0.000 0.0001 1.0000 1.0000   1.01  1  1  4.7E-003
  3 : -3.32E-013 1.15E-012 0.000 0.0000 1.0000 1.0000   1.00  1  1  4.7E-010

iter segundo digito       c*x               b*y
  3      0.3   7.3  4.5078508895e-013 -3.3191889344e-013
|Ax-b| =  3.0e-013, [Ay-c]_+ =  0.0E+000, |x|= 1.7e+000, |y|= 1.9e-013

Tiempo Detallado (seg)
   Ant          IPM          Post
1.875E-001    2.969E-001    1.094E-001
Norma máxima: ||b||=1, ||c|| = 1,
Cholesky |agregados|=0, |saltos| = 0, ||L.L|| = 1.
Tiempo de CPU = 0.59375 Segundos
*************************************************
Comprobando viabilidad (eps = 1.0000e-003):
  SDP viable
Comprobando norma euclidiana de la solución (max = 1.0000e+006):
  Norma = 1.9163e-013
Comprobando los momentos de primer orden (abs tol = 1.0000e-003):
  Solución 1
    SDP objetivo = 3.3192e-013
    La solución alcanza el mismo objetivo
Optimalidad global certificada numéricamente

status =

    obj: 1                    %Signifiva que el problema fue solucionado
```

`double`: Mediante esta instrucción se puede obtener el o los puntos en los cuales se alcanza el valor óptimo del polinomio. El operador `double` convierte una medida o sus variables en un número de punto flotante. Para el ejemplo anterior:

```
>> x = double([x1 x2 x3])

x =

  1.0e-018 *

         0         0    0.5204
```

Para su correcto funcionamiento, esta herramienta necesita una herramienta de programación Semidefinida (SDP solver), para realizar los cálculos correspondientes al problema dual definido como en el capítulo anterior; para lo cual dispone de 2 herramientas: SeDuMi y Yalmip.

SeDuMi

SeDuMi es un complemento para Matlab, que permite resolver problemas de optimización con restricciones lineales, cuadráticas y semidefinidas. Es posible tener un valor de datos complejos y variables en SeDuMi. Es usado por herramientas que necesitan una solución a un problema de Programación Semidefinida (SDP) y para nuestro caso, por Gloptipoly. el llamado a SeDuMi desde MatLab se hace con la instrucción:

$$[\text{x,y,info}] = \text{sedumi(A,b,c,K,PARS);}$$

donde la mayoría de los parámetros pueden omitirse. Con este llamado SeDuMi resuelve los siguientes problemas de optimización principal-dual:

$$\begin{array}{cc} \min c^T x & \max b^T y \\ Ax = b & A^T y + s = c \\ x \in K & y \in K^* \end{array}$$

donde $A \in \mathbb{R}^{m \times n}$, $x, s, c \in \mathbb{R}^n$, $y, b \in \mathbb{R}^m$, K es un cono y K^* es su cono dual. (Para más información sobre SeDuMi, ver [8]).

Yalmip

Yalmip es un lenguaje de modelización para definir y resolver problemas de optimización avanzadas. Es implementado como una caja de herramientas para MATLAB. Es una herramienta de Programación Semidefinida (SDP) al igual que SeDuMi, y puede ser utilizadada por Gloptipoly a la par con SeDuMi.

El lenguaje de YALMIP es coherente con la sintaxis estándar de MATLAB, por lo que resulta de fácil uso para cualquier persona familiarizada con MATLAB. otro beneficio de YALMIP es que otra herramienta implementa una gran cantidad de trucos de modelado, llevando al usuario a concentrarse en la reunión de un modelo de alto nivel, mientras que Yalmip se encarga del bajo nivel para obtener numéricamente modelos de manera eficiente y racional como sea posible. (Para más información sobre Yalmip, visitar la página web: *control.ee.ethz.ch/∼joloef/yalmip.php*)

4.3. Experimentación

A continuación se tienen los resultados de la experimentación con el algoritmo realizada en 3 partes.

↪ Ejercicios de libros de cálculo vectorial.

↪ Problemas típicos de optimización.

↪ Función de verosimilitud de una GIP.

En un computador con procesador *AMD Athlon 64 2800+* de 1.8 GHz, memoria RAM de 1 GB y sistema operativo *Windows XP Profesional SP3*, los resultados obtenidos de la experimentación con el algoritmo se dan a continuación:

4.3.1. Ejercicios de cálculo vectorial

Ejemplo 1:

Se pide calcular el valor mínimo, si existe, del polinomio $p : \mathbb{R}^2 \to \mathbb{R}$ definido como:

$$p(x_1, x_2) = 3 + 3x_1^2 + x_2^2 + 2x_1^2x_2^2.$$

solución: El polinomio a optimizar es de grado 4 y 2 variables. En primer lugar, se crea el polinomio mediante las siguientes instrucciones:

```
>>mpol x1 x2;
>>f1 = 3 + 3*x1^2+x2^2 + 2*x1^2*x2^2;
```

A continuación se define el problema de optimización:

```
>> P = msdp(min(f1))
GloptiPoly 3.4 del 30 de Septiembre de 2008
Define el problema de momentos SDP
  Función objetivo válida
  Número de restricciones = 0
  Número de momentos de restricciones = 0
Medida 1
  Grado = 4
  Variables = 2
  Momentos = 15
Orden de relajación = 2
Masa de medida 1 a un conjunto
Número total de momentos = 15
Realizar sustituciones de Momentos
Número de momentos después de sustituciones = 14
Generar momentos y las restricciones de soporte
Generar el problema de momentos SDP
```

```
Problema de Momentos SDP
  Medidas etiquetadas       = 1
  Ordenes de relajación     = 2
  Variables de Decisión     = 14
  Desigualdades SD          = 6x6
```

Resultan 15 momentos. El siguiente paso es encontrar la solución con `msol`:

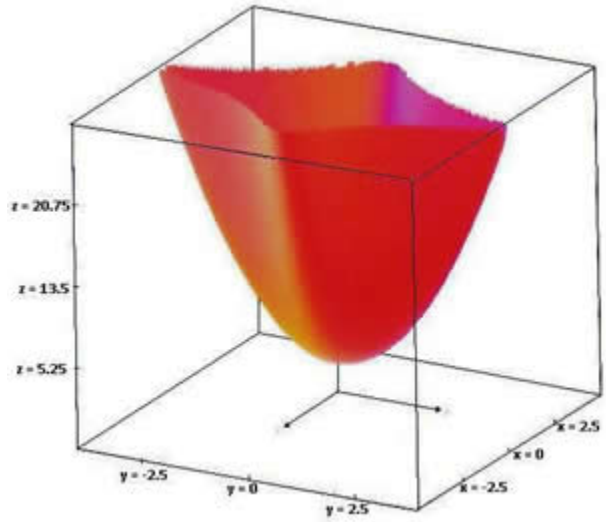
```
>> [status.obj] = msol(P)
GloptiPoly 3.4 del 30 de Septiembre de 2008
Define el problema de momentos SDP
  Función objetivo válida
  Número de restricciones = 0
  Número de momentos de restricciones = 0
Medida 1
  Grado = 4
  Variables = 2
  Momentos = 15
Orden de relajación = 2
Masa de medida 1 a un conjunto
Número total de momentos = 15
Realizar sustituciones de Momentos
Número de momentos después de sustituciones = 14
Generar momentos y las restricciones de soporte
Generar el problema de momentos SDP

Problema de Momentos SDP
  Medidas etiquetadas       = 1
  Ordenes de relajación     = 2
  Variables de Decisión     = 14
  Desigualdades SD          = 6x6

>> [status.obj] = msol(P)
GloptiPoly 3.4 del 30 de Septiembre de 2008
Resolver problema de Momentos SDP
*************************************************
Llamando a SeDuMi
SeDuMi 1.1R3 by AdvOL, 2006 and Jos F. Sturm, 1998-2003.
Alg = 2: xz-corrector, Adaptive Step-Differentiation, theta = 0.250, beta = 0.500
eqs m = 14, orden n = 7, dim = 37, bloques = 2
nnz(A) = 20 + 0, nnz(ADA) = 196, nnz(L) = 105
 it :     b*y       gap    delta  rata   t/tP*  t/tD*   feas cg cg  prec
  0 :            4.57E+000 0.000
  1 : -4.63E-001 9.45E-001 0.000 0.2067 0.9000 0.9000   1.59  1  1  1.7E+000
  2 : -1.31E-001 1.97E-001 0.000 0.2088 0.9000 0.9000   1.31  1  1  1.1E+000
  3 : -1.23E-003 4.46E-003 0.000 0.0226 0.9900 0.9900   1.06  1  1  8.3E-001
  4 : -6.88E-010 1.55E-008 0.000 0.0000 1.0000 1.0000   1.00  1  1  8.3E-006
  5 :  3.15E-016 4.62E-015 0.000 0.0000 1.0000 1.0000   1.00  1  1  2.1E-012
```

```
iter segundo digito       c*x               b*y
  5      0.3   9.6  4.7235294684e-015  3.1536555918e-016
|Ax-b| =  5.5e-015, [Ay-c]_+ =  6.0E-016, |x|= 3.7e+000, |y|= 1.7e+000

Tiempo Detallado (seg)
   Ant           IPM           Post
2.031E-001    3.281E-001    9.375E-002
Norma máxima: ||b||=3, ||c|| = 1,
Cholesky |agregados|=0, |saltos| = 0, ||L.L|| = 1.
Tiempo de CPU = 0.625 Segundos
*****************************************************
Comprobando viabilidad (eps = 1.0000e-003):
  Marginalmente viable SDP: residual = -5.3889e-016
Comprobando norma euclidiana de la solución (max = 1.0000e+006):
  Norma = 1.6906e+000
Comprobando los momentos de primer orden (abs tol = 1.0000e-003):
  Solución 1
    SDP objetivo = 3.0000e+000
    La solución alcanza el mismo objetivo
Optimalidad global certificada numéricamente

status =

    obj: 1
```

El valor de `status` 1 significa que la herramienta encontró por lo menos una solución, y para visualizarlas se hace bajo la instrucción:

```
>> x = double([x1 x2])

x =

  1.0e-015 *

    0.0018   -0.1009
```

De donde se puede visualizar que el valor óptimo obtenido mediante la herramienta se tiene en el punto $(0.018*10^{-15}, -0.1009*10^{-15})$. La siguiente es la gráfica del polinomio, obtenida mediante la herramienta *Matemáticas de Microsoft*, cambiando las variables x_1, x_2 por x, y respectivamente:



Usando el cálculo diferencial en varias variables, se tiene que:

$$\nabla p(x_1, x_2) = (6x_1 + 4x_1x_2^2\ ,\ 2x_2 + 4x_1^2x_2)$$
$$\nabla p(x_1x_2) = 0 \ \text{ sí y sólo si } \ x_1 = 0 \ \text{ y } \ x_2 = 0$$

De esta forma, $p(0,0) = 3$, con lo cual se puede ver que $p(0,0)$ coincide con el valor obtenido en `out.obj` y el valor obtenido en `out.sol{:}` de $(0{,}0011, -0{,}9067)$ se aproxima al punto $(0,0)$ con una precisión de 10^{-16}.

Ejemplo 2:

Dado el polinomio $p : \mathbb{R}^2 \to \mathbb{R}$ definido como

$$p(x_1, x_2) = 4x_1^2 + x_1x_2 - 4x_2^2 - 2{,}1x_1^4 + 4x_2^4 + \frac{1}{3}x_1^6$$

se pide calcular su valor mínimo.

El polinomio a optimizar es de grado 6 y 2 variables. Definiendo el polinomio:

```
>> mpol x1 x2
>> f3 = 4*x1^2+x1*x2-4*x2^2-2.1*x1^4+4*x2^4+x1^6/3;
```

Definiendo el problema de optimización:

```
>> P = msdp(min(f3))
GloptiPoly 3.4 del 30 de Septiembre de 2008
Define el problema de momentos SDP
  Función objetivo válida
```

```
  Número de restricciones = 0
  Número de momentos de restricciones = 0
Medida 1
  Grado = 6
  Variables = 2
  Momentos = 28
Orden de relajación = 3
Masa de medida 1 a un conjunto
Número total de momentos = 28
Realizar sustituciones de Momentos
Número de momentos después de sustituciones = 27
Generar momentos y las restricciones de soporte
Generar el problema de momentos SDP

Problema de Momentos SDP
  Medidas etiquetadas       = 1
  Ordenes de relajación     = 3
  Variables de Decisión     = 27
  Desigualdades SD          = 10x10
```

Resultan 28 momentos. El siguiente paso es encontrar la solución con `msol`:

```
>> [status.obj] = msol(P)
GloptiPoly 3.4 del 30 de Septiembre de 2008
Resolver problema de Momentos SDP
*****************************************************
Llamando a SeDuMi
SeDuMi 1.1R3 by AdvOL, 2006 and Jos F. Sturm, 1998-2003.
Alg = 2: xz-corrector, Adaptive Step-Differentiation, theta = 0.250, beta = 0.500
eqs m = 27, orden n = 11, dim = 101, bloques = 2
nnz(A) = 54 + 0, nnz(ADA) = 729, nnz(L) = 378
 it :     b*y       gap    delta  rata   t/tP*  t/tD*   feas cg cg  prec
  0 :            3.86E+000 0.000
  1 :  9.71E-001 8.93E-001 0.000 0.2314 0.9000 0.9000   0.73  1  1  1.9E+000
  2 :  9.44E-001 8.35E-002 0.000 0.0935 0.9900 0.9900   1.62  1  1  1.9E-001
  3 :  9.99E-001 2.02E-002 0.000 0.2421 0.9000 0.9000   1.51  1  1  3.2E-002
  4 :  1.03E+000 1.98E-003 0.000 0.0977 0.9900 0.9900   1.38  1  1  2.4E-003
  5 :  1.03E+000 4.26E-004 0.000 0.2155 0.9000 0.9000   1.00  1  1  5.0E-004
  6 :  1.03E+000 6.77E-005 0.000 0.1590 0.9000 0.9111   1.05  1  1  8.6E-005
  7 :  1.03E+000 1.40E-005 0.000 0.2059 0.9000 0.9120   1.10  1  1  1.7E-005
  8 :  1.03E+000 2.44E-006 0.000 0.1753 0.9000 0.9053   1.12  1  1  2.8E-006
  9 :  1.03E+000 4.24E-007 0.000 0.1734 0.9000 0.9054   1.13  1  1  4.9E-007
 10 :  1.03E+000 6.93E-008 0.000 0.1635 0.9000 0.9051   1.10  1  1  8.2E-008
 11 :  1.03E+000 1.08E-008 0.000 0.1559 0.9000 0.9043   1.05  2  1  1.4E-008
 12 :  1.03E+000 9.10E-010 0.335 0.0842 0.9900 0.9900   1.02  3  3  1.1E-009

iter segundo digito       c*x               b*y
 12      0.2   9.3  1.0316284530e+000  1.0316284525e+000
|Ax-b| =  4.2e-009, [Ay-c]_+ =  2.1E-011, |x|= 6.5e+000, |y|= 6.6e-001
```

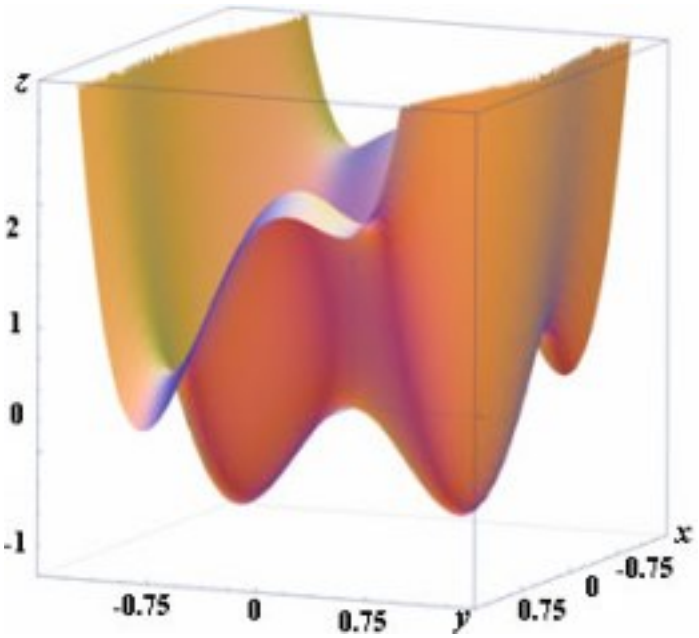
```
Tiempo Detallado (seg)
   Ant           IPM           Post
1.563E-002    1.719E-001    1.563E-002
Norma máxima: ||b||=4, ||c|| = 1,
Cholesky |agregados|=0, |saltos| = 0, ||L.L|| = 655.84.
Tiempo de CPU = 0.20313 Segundos
*****************************************************
Comprobando viabilidad (eps = 1.0000e-003):
  Marginalmente viable SDP: residual = -2.1289e-011
Comprobando norma euclidiana de la solución (max = 1.0000e+006):
  Norma = 6.5981e-001
Comprobando los momentos de primer orden (abs tol = 1.0000e-003):
  Solución 1
    SDP objetivo = -1.0316e+000
    La solución alcanza diferentes objetivos = -1.2008e-015
Comprobandolas filas de la matriz de momentos (rel gap svd = 1.0000e-003):
  Medida 1
    Rango de cambio = 1
    La matriz de Momentos de orden  1 tiene tamaño  3 y rango  2
    La matriz de Momentos de orden  2 tiene tamaño  6 y rango  2
  El rango de condiciones puede garantizar la optimalidad global
Tratando de extraer soluciones (rel tol detección de la base = 1.0000e-006):
  Medida 1
    Máximo error relativo = 3.0603e-008
    2 soluciones extraídas
Comprobando extracción de soluciones (abs tol = 1.0000e-003):
  Solución 1
    SDP objetivo = -1.0316e+000
    La solución alcanza el mismo objetivo
  Solución 2
    SDP objetivo = -1.0316e+000
    La solución alcanza el mismo objetivo
2 soluciones óptimas globales extraídas
Optimalidad global certificada numéricamente

status =

    obj: 1
```

Así, el valor mínimo del polinomio encontrado por *Gloptipoly 3* es $-1{,}0316$ y lo alcanza en 2 puntos dados mediante la instrucción:

```
>> x = double([x1 x2])

x(:,:,1) =

    0.0898   -0.7127


x(:,:,2) =
```

```
   -0.0898    0.7127
```

de donde se tiene que el valor mínimo es alcanzado en los puntos (0.0898,-0.7127) y (-0.0898,0.7127). La gráfica del polinomio teniendo en cuenta es la siguiente:



donde se pueden ver los puntos mínimos, además del valor mínimo.

Ejemplo 3

Dado el polinomio $p : \mathbb{R}^4 \to \mathbb{R}$ definido como

$$p(x_1, x_2, x_3, x_4) = 1 - (x_1 - 3)^8 - (x_2 - 4)^8 - 6(x_3 - 5)^8 - 4(x_4 - 6)^8$$

se pide calcular su valor máximo.
El polinomio a optimizar es de grado 8 y 4 variables. Definiendo el polinomio:

```
>> mpol x1 x2 x3 x4
>> f4 = 1 - (x1-3)^8 - (x2-4)^8 - 6*(x3-5)^8 - 4*(x4-6)^8;
```

Definiendo el problema de optimización:

```
>> P = msdp(max(f4))
GloptiPoly 3.4 del 30 de Septiembre de 2008
Define el problema de momentos SDP
  Función objetivo válida
  Número de restricciones = 0
  Número de momentos de restricciones = 0
```

```
Medida 1
  Grado = 8
  Variables = 4
  Momentos = 495
Orden de relajación = 4
Masa de medida 1 a un conjunto
Número total de momentos = 495
Realizar sustituciones de Momentos
Número de momentos después de sustituciones = 494
Generar momentos y las restricciones de soporte
Generar el problema de momentos SDP

Problema de Momentos SDP
  Medidas etiquetadas       = 1
  Ordenes de relajación     = 4
  Variables de Decisión     = 494
  Desigualdades SD          = 70x70
```

Resultan 495 momentos. El siguiente paso es encontrar la solución con `msol`:

```
>> [status.obj] = msol(P)
GloptiPoly 3.4 del 30 de Septiembre de 2008
Resolver problema de Momentos SDP
************************************************
Llamando a SeDuMi
SeDuMi 1.1R3 by AdvOL, 2006 and Jos F. Sturm, 1998-2003.
Alg = 2: xz-corrector, Adaptive Step-Differentiation, theta = 0.250, beta = 0.500
eqs m = 494, orden n = 71, dim = 4901, bloques = 2
nnz(A) = 2484 + 0, nnz(ADA) = 244036, nnz(L) = 122265
 it :     b*y       gap    delta  rata   t/tP*  t/tD*   feas cg cg  prec
  0 :            5.68E+005 0.000
  1 :  2.31E+006 2.23E+005 0.000 0.3936 0.9000 0.9000   0.82  1  1  2.1E+000
  2 :  3.91E+006 7.60E+004 0.000 0.3399 0.9000 0.9000   1.63  1  1  5.4E-001
  3 :  5.25E+006 2.70E+004 0.000 0.3550 0.9000 0.9000   1.59  1  1  1.5E-001
  4 :  6.10E+006 9.97E+003 0.000 0.3699 0.9000 0.9000   1.05  1  1  5.4E-002
  5 :  6.58E+006 4.37E+003 0.000 0.4377 0.9000 0.9000   0.92  1  1  2.6E-002
  6 :  6.85E+006 2.33E+003 0.000 0.5341 0.9000 0.9000   0.65  1  1  1.8E-002
  7 :  7.15E+006 1.10E+003 0.000 0.4723 0.9000 0.9000   0.77  1  1  9.3E-003
  8 :  7.34E+006 5.43E+002 0.000 0.4934 0.9000 0.9000   0.41  1  1  5.4E-003
  9 :  7.60E+006 2.31E+002 0.000 0.4259 0.9000 0.9000   0.64  1  1  2.7E-003
 10 :  7.77E+006 1.02E+002 0.000 0.4420 0.9000 0.9000   0.37  1  1  1.9E-003
 11 :  7.97E+006 4.25E+001 0.000 0.4155 0.9000 0.9000   0.55  1  1  9.9E-004
 12 :  8.10E+006 1.89E+001 0.000 0.4441 0.9000 0.9000   0.31  1  1  7.0E-004
 13 :  8.27E+006 7.66E+000 0.000 0.4059 0.9000 0.9000   0.52  1  1  3.6E-004
 14 :  8.38E+006 3.33E+000 0.000 0.4347 0.9000 0.9000   0.30  1  1  2.5E-004
 15 :  8.51E+006 1.35E+000 0.000 0.4060 0.9000 0.9000   0.51  1  1  1.3E-004
 16 :  8.59E+006 6.00E-001 0.000 0.4435 0.9000 0.9000   0.31  1  1  9.2E-005
 17 :  8.69E+006 2.44E-001 0.000 0.4075 0.9000 0.9000   0.52  1  1  4.8E-005
 18 :  8.75E+006 1.10E-001 0.000 0.4511 0.9000 0.9000   0.33  1  1  3.3E-005
 19 :  8.83E+006 4.53E-002 0.000 0.4106 0.9000 0.9000   0.54  1  1  1.7E-005
```

```
 20 :  8.88E+006 2.10E-002 0.000 0.4637 0.9000 0.9000   0.36  1  1  1.2E-005
 21 :  8.93E+006 1.93E-016 0.000 0.0000 0.9000 0.7848   0.57  1  1  6.2E-006
 22 :  8.96E+006 9.26E-017 0.000 0.4793 0.7069 0.9000   0.39  1  1  4.4E-006
 23 :  9.01E+006 3.31E-017 0.000 0.3578 0.9000 0.9127   0.61  1  1  2.0E-006
 24 :  9.03E+006 1.63E-017 0.000 0.4910 0.9127 0.9000   0.55  1  1  1.3E-006
 25 :  9.05E+006 9.45E-018 0.000 0.5810 0.6646 0.9000   0.57  1  1  9.6E-007
 26 :  9.07E+006 4.39E-018 0.000 0.4649 0.9000 0.9027   0.70  1  1  5.4E-007
 27 :  9.08E+006 2.61E-018 0.000 0.5949 0.6612 0.9000   0.64  1  1  4.0E-007
 28 :  9.09E+006 1.32E-018 0.000 0.5044 0.9000 0.9007   0.78  2  2  2.3E-007
 29 :  9.10E+006 8.38E-019 0.000 0.6352 0.6037 0.9000   0.66  1  1  1.8E-007
 30 :  9.11E+006 4.24E-019 0.000 0.5063 0.9000 0.9024   0.79  2  2  1.0E-007
 31 :  9.11E+006 2.70E-019 0.000 0.6372 0.6374 0.9000   0.67  2  1  7.9E-008
 32 :  9.12E+006 1.42E-019 0.000 0.5250 0.8877 0.9000   0.79  2  2  4.8E-008
 33 :  9.12E+006 9.15E-020 0.000 0.6453 0.9000 0.9000   0.68  2  2  3.6E-008
 34 :  9.12E+006 4.83E-020 0.000 0.5277 0.9000 0.9000   0.76  2  2  2.2E-008
 35 :  9.12E+006 3.12E-020 0.000 0.6452 0.9000 0.9000   0.67  2  2  1.7E-008
 36 :  9.13E+006 1.67E-020 0.000 0.5354 0.9000 0.9000   0.75  2  2  1.0E-008
 37 :  9.13E+006 1.07E-020 0.000 0.6436 0.9000 0.9000   0.68  2  2  7.7E-009

iter segundo digito       c*x               b*y
 37     20.4   Inf  9.1279899790e+006  9.1280744197e+006
|Ax-b| =  1.5e-001, [Ay-c]_+ =  0.0E+000, |x|= 1.2e+007, |y|= 1.4e+005

Tiempo Detallado (seg)
   Ant           IPM           Post
2.031E-001    2.044E+001    3.125E-002
Norma máxima: ||b||=8957952, ||c|| = 1,
Cholesky |agregados|=0, |saltos| = 0, ||L.L|| = 7.21632.
Tiempo de CPU = 20.6719 Segundos
******************************************************
Comprobando viabilidad (eps = 1.0000e-003):
  SDP viable
Comprobando norma euclidiana de la solución (max = 1.0000e+006):
  Norma = 1.3510e+005
Comprobando los momentos de primer orden (abs tol = 1.0000e-003):
  Solución 1
    SDP objetivo = -6.2356e+003
    La solución alcanza diferentes objetivos = -2.5289e+003
Comprobando las filas de la matriz de momentos (rel gap svd = 1.0000e-003):
  Medida 1
    Rango de cambio = 1
    La matriz de Momentos de orden  1 tiene tamaño  5 y rango  5
    La matriz de Momentos de orden  2 tiene tamaño 15 y rango 15
    La matriz de Momentos de orden  3 tiene tamaño 35 y rango 35
    La matriz de Momentos de orden  4 tiene tamaño 70 y rango 70
  El rango de condiciones no puede garantizar la optimalidad global
Tratando de extraer soluciones (rel tol detección de la base = 1.0000e-006):
  Medida 1
    Base incompleta - sin solución extraída
```

```
    La optimalidad global no se puede garantizar
Ninguna solución óptima global pudo ser extraída
Optimalidad global no se puede garantizar

status =

    obj: 0
```

En este caso la herramienta no puede extraer una solución a pesar de que matemáticamente si la tiene (el valor máximo es 1 y lo alcanza en el punto $(3, 4, 5, 6)$). Sin embargo, haciendo una traslación de dicho punto al origen de coordenadas, la función se convierte en $p_1(x_1, x_2, x_3, x_4) = 1 - x_1^8 - x_2^8 - 6x_3^8 - 4x_4^8$, para la cual si puede extrae el valor óptimo y el punto donde lo alcanza (Valor óptimo 1 y lo alcanza en (0.4241,-0.3784,0.0872,-0.1582) con aproximación de 10^{-8}). Una de las posibles causas es que la herramienta utiliza un criterio de aproximación cuando los valores óptimos son alcanzados en puntos cuya norma es menor que 1.

Ejemplo 4

Dado el polinomio $p : \mathbb{R}^4 \to \mathbb{R}$ definido como

$$p(x_1, x_2, x_3, x_4) = 1 - x_1^{16} - x_2^{16} - 6x_3^{16} - 4x_4^{16}$$

se pide calcular su valor máximo.
El polinomio a optimizar es de grado 16 y 4 variables. Definiendo el polinomio:

```
>> mpol x1 x2 x3 x4
>> f5 = 1 - x1^16 - x2^16 - 6*x3^16 - 4*x4^16;
```

Definiendo el problema de optimización:

```
>> P = msdp(max(f5))
GloptiPoly 3.4 del 30 de Septiembre de 2008
Define el problema de momentos SDP
  Función objetivo válida
  Número de restricciones = 0
  Número de momentos de restricciones = 0
Medida 1
  Grado = 16
  Variables = 4
  Momentos = 4845
Orden de relajación = 8
Masa de medida 1 a un conjunto
Número total de momentos = 4845
Realizar sustituciones de Momentos
Número de momentos después de sustituciones = 4844
Generar momentos y las restricciones de soporte
Generar el problema de momentos SDP
```

```
Problema de Momentos SDP
  Medidas etiquetadas       = 1
  Ordenes de relajación     = 8
  Variables de Decisión     = 4844
  Desigualdades SD          = 495x495
```

Resultan 4.845 momentos. El siguiente paso es encontrar la solución con `msol`:

```
>> [status.obj] = msol(P)
GloptiPoly 3.4 del 30 de Septiembre de 2008
Resolver problema de Momentos SDP
*************************************************
Llamando a SeDuMi
SeDuMi 1.1R3 by AdvOL, 2006 and Jos F. Sturm, 1998-2003.
Alg = 2: xz-corrector, Adaptive Step-Differentiation, theta = 0.250, beta = 0.500
eqs m = 4844, orden n = 496, dim = 245026, bloques = 2
nnz(A) = 122759 + 0, nnz(ADA) = 23464336, nnz(L) = 11734590
 it :     b*y       gap    delta  rata   t/tP*  t/tD*   feas cg cg  prec
  0 :            9.00E-002 0.000
  1 : -2.51E+000 4.63E-002 0.000 0.5142 0.9000 0.9000   1.73  1  1  2.4E+000
  2 : -6.45E-001 2.11E-002 0.000 0.4552 0.9000 0.9000   3.51  1  1  5.9E-001
  3 : -1.24E-001 9.64E-003 0.000 0.4575 0.9000 0.9000   3.29  1  1  2.3E-001
  4 : -2.17E-002 3.01E-003 0.000 0.3119 0.9000 0.9000   2.25  1  1  1.2E-001
  5 : -3.38E-003 7.78E-004 0.000 0.2589 0.9000 0.9000   1.75  1  1  8.6E-002
  6 : -7.15E-004 2.49E-004 0.000 0.3203 0.9000 0.9000   1.47  1  1  5.8E-002
  7 : -2.04E-004 8.58E-005 0.000 0.3443 0.9000 0.9000   1.33  1  1  2.0E-002
  8 : -5.20E-005 2.89E-005 0.000 0.3367 0.9000 0.9000   1.25  1  1  5.6E-003
  9 : -1.96E-005 1.17E-005 0.000 0.4064 0.9000 0.9000   1.20  1  1  1.9E-003
 10 : -5.05E-006 4.06E-006 0.000 0.3461 0.9000 0.9000   1.17  1  1  5.8E-004
 11 : -1.91E-006 1.59E-006 0.000 0.3902 0.9000 0.9000   1.15  1  1  2.0E-004
 12 : -4.81E-007 4.90E-007 0.000 0.3088 0.9000 0.9000   1.13  1  1  5.4E-005
 13 : -1.61E-007 1.69E-007 0.000 0.3451 0.9000 0.9000   1.12  1  1  1.7E-005
 14 : -3.93E-008 4.69E-008 0.000 0.2776 0.9000 0.9000   1.11  1  1  4.1E-006
 15 : -1.01E-008 1.28E-008 0.000 0.2722 0.9000 0.9000   1.11  1  1  1.0E-006
 16 : -2.61E-009 3.55E-009 0.000 0.2782 0.9000 0.9000   1.10  1  2  2.5E-007
 17 : -7.72E-010 1.05E-009 0.000 0.2964 0.9000 0.9000   1.09  2  2  6.8E-008
 18 : -6.02E-010 2.61E-010 0.000 0.2482 0.9000 0.2719   1.08  4  4  2.6E-008
 19 :  1.54E-010 4.68E-011 0.000 0.1792 0.9204 0.9000   1.05 10  9  5.1E-009

iter segundo digito       c*x               b*y
 19   9694.4   6.3  4.7626769226e-010  1.5375437737e-010
|Ax-b| =  6.6e-010, [Ay-c]_+ =  8.1E-011, |x|= 7.6e+000, |y|= 6.4e-002

Tiempo Detallado (seg)
   Ant          IPM          Post
8.906E+000    9.694E+003    1.406E+000
Norma máxima: ||b||=6, ||c|| = 1,
Cholesky |agregados|=8, |saltos| = 0, ||L.L|| = 500000.
Tiempo de CPU = 9704.6719 Segundos
```

```
*****************************************************
Comprobando viabilidad (eps = 1.0000e-003):
  Marginalmente viable SDP: residual = -8.1127e-011
Comprobando norma euclidiana de la solución (max = 1.0000e+006):
  Norma = 6.3976e-002
Comprobando los momentos de primer orden (abs tol = 1.0000e-003):
  Solución 1
    SDP objetivo = 1.0000e+000
    La solución alcanza el mismo objetivo
Optimalidad global certificada numéricamente

status =

    obj: 1
```

Así, el valor mínimo del polinomio encontrado por *Gloptipoly 3* es 1 y lo alcanza en el punto dado mediante la instrucción:

```
>> x = double([x1 x2 x3 x4])

x =

  1.0e-007 *

   -0.0918   -0.0716   -0.0124    0.1442
```

Lo cual indica que el valor máximo encontrado se encuentra en el punto (-0.0918$*10^{-7}$,-0.0716$*10^{-7}$,-0.0124$*10^{-7}$,0.1442$*10^{-7}$). Ahora, mediante la teoría del cálculo diferencial en varias variables, se obtiene el valor y el punto donde el polinomio alcanza el mínimo buscando los puntos críticos con el gradiente.

$$\nabla p(x_1, x_2, x_3, x_4) = (-16x_1^{15}, -16x_2^{15}, -96x_3^{15}, -64x_4^{15})$$

$\nabla p(x_1, x_2, x_3, x_4) = (0, 0, 0, 0)$ implica que $x_1 = x_2 = x_3 = x_4 = 0$. De ahi el punto crítico es $(0, 0, 0, 0)$, en el cual se alcanza el máximo.

$$(x_1, x_2, x_3, x_4) = (0, 0, 0, 0)$$

Puede verse que los resultados obtenidos matemáticamente son bastante cercanos a los obtenidos mediante Gloptipoly, con lo cual se puede decir que el programa funciona correctamente para el caso. No es posible observar gráficamente el polinomio, dado que la dimensión del mismo es 5. Además puede verse que la herramienta SeDuMi tardó 9500 Segundos (aproximadamente 2 horas y 40 minutos), lo cual indica que el coste computacional es elevado.

4.3.2. Problemas típicos de optimización

A continuación se hará la experimentación con ejercicios típicos de optimización, para lo cual se han tomado algunos ejercicios del libro *More test examples for nonlinear programming codes*[11]. Los resultados obtenidos por Gloptipoly se pueden comparar con los citados por el autor del libro.

Ejemplo 1

Del libro citado, se prueba el algoritmo para el problema 384 pag. 203.

$$\begin{aligned} f(x) &= -486x_1 - 640x_2 - 758x_3 - 776x_4 - 477x_5 - 707x_6 - 175x_7 - 619x_8 \\ &\quad -627x_9 - 614x_{10} - 475x_{11} - 377x_{12} - 524x_{13} - 468x_{14} - 529x_{15} \end{aligned}$$

bajo restricciones de la forma:

$$g_i(x) = b_i - \sum_{j=1}^{15} a_{ij} x_j^2 \geq 0,\ i = 1, 2, \ldots, 10.$$

donde $[a_{ij}]$ y (b_i) estan definidos como sigue:

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	b_j
a_{1j}	100	100	10	5	10	0	0	25	0	10	55	5	45	20	0	385
a_{2j}	90	100	10	35	20	5	0	35	55	25	20	0	40	25	10	470
a_{3j}	70	50	0	55	25	100	40	50	0	30	60	10	30	0	40	560
a_{4j}	50	0	0	65	35	100	35	60	0	15	0	75	35	30	65	565
a_{5j}	50	10	70	60	45	45	0	35	65	5	75	100	75	10	0	645
a_{6j}	40	0	50	95	50	35	10	60	0	45	15	20	0	5	5	430
a_{7j}	30	60	30	90	0	30	5	25	0	70	20	25	70	15	15	485
a_{8j}	20	30	40	25	40	25	15	10	80	20	30	30	5	65	20	455
a_{9j}	10	70	10	55	25	65	0	30	0	0	25	0	15	50	55	390
a_{10j}	5	10	100	5	20	5	10	35	95	70	20	10	35	10	30	460

El polinomio a optimizar es de grado 1 y 15 variables. Definiendo el polinomio y sus restricciones:

```
>> mpol x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x11 x12 x13 x14 x15
>> tp1 = -486*x1 - 640*x2 - 758*x3 - 776*x4 - 477*x5 - 707*x6 - 175*x7 - 619*x8 - 627*x9 ...
   - 614*x10 - 475*x11 - 377*x12 - 524*x13 - 468*x14 - 529*x15;
>> 385 - (100*x1^2 + 100*x2^2 + 10*x3^2 + 5*x4^2 + 10*x5^2 + 25*x8^2 + 10*x10^2 ...
   + 55*x11^2 + 5*x12^2 + 45*x13^2 + 20*x14^2) >= 0;
>> 470 - (90*x1^2 + 100*x2^2 + 10*x3^2 + 35*x4^2 + 20*x5^2 + 5*x6^2 + 35*x8^2 ...
   + 55*x9^2 + 25*x10^2 + 20*x11^2 + 40*x13^2 + 25*x14^2 + 10*x15^2) >= 0;
```

```
>> 560 - (70*x1^2 + 50*x2^2 + 55*x4^2 + 25*x5^2 + 100*x6^2 + 40*x7^2 + 50*x8^2 ...
   + 30*x10^2 + 60*x11^2 + 10*x12^2 + 30*x13^2 + 40*x15^2) >= 0;
>> 565 - (50*x1^2 + 65*x4^2 + 35*x5^2 + 100*x6^2 + 35*x7^2 + 60*x8^2 ...
   + 15*x10^2 + 75*x12^2 + 35*x13^2 + 30*x14^2 + 65*x15^2) >= 0;
>> 645 - (50*x1^2 + 10*x2^2 + 70*x3^2 + 60*x4^2 + 45*x5^2 + 45*x6^2 + 35*x8^2 + 65*x9^2 ...
   + 5*x10^2 + 75*x11^2 + 100*x12^2 + 75*x13^2 + 10*x14^2) >= 0;
>> 430 - (40*x1^2 + 50*x3^2 + 95*x4^2 + 50*x5^2 + 35*x6^2 + 10*x7^2 + 60*x8^2 ...
   + 45*x10^2 + 15*x11^2 + 20*x12^2 + 5*x14^2 + 5*x15^2) >= 0;
>> 485 - (30*x1^2 + 60*x2^2 + 30*x3^2 + 90*x4^2 + 30*x6^2 + 5*x7^2 + 25*x8^2 ...
   + 70*x10^2 + 20*x11^2 + 25*x12^2 + 70*x13^2 + 15*x14^2 + 15*x15^2) >= 0;
>> 455 - (20*x1^2 + 30*x2^2 + 40*x3^2 + 25*x4^2 + 40*x5^2 + 25*x6^2 + 15*x7^2 + 10*x8^2 ...
   + 80*x9^2 + 20*x10^2 + 30*x11^2 + 30*x12^2 + 5*x13^2 + 65*x14^2 + 20*x15^2) >= 0;
>> 390 - (10*x1^2 + 70*x2^2 + 10*x3^2 + 55*x4^2 + 25*x5^2 + 65*x6^2 + 30*x8^2 ...
   + 25*x11^2 + 15*x13^2 + 50*x14^2 + 55*x15^2) >= 0;
>> 460 - (5*x1^2 + 10*x2^2 + 100*x3^2 + 5*x4^2 + 20*x5^2 + 5*x6^2 + 10*x7^2 + 35*x8^2 ...
   + 95*x9^2 + 70*x10^2 + 20*x11^2 + 10*x12^2 + 35*x13^2 + 10*x14^2 + 30*x15^2) >= 0;
```

Definiendo el problema de optimización:

```
>> P = msdp(min(tp1))
GloptiPoly 3.4 del 30 de Septiembre de 2008
Define el problema de momentos SDP
  Función objetivo válida
  Número de restricciones de soporte = 0
  Número de restricciones momentos = 0
Medida 1
  Grado = 1
  Variables = 15
  Momentos = 136
Orden de relajación = 1
Masa de medida 1 a un conjunto
Número total de momentos = 136
Realizar sustituciones de Momentos
Número de momentos después de sustituciones = 135
Generar momentos y las restricciones
Generar el problema de momentos SDP

Problema de Momentos SDP
  Medida etiquetada         = 1
  Orden de relajación       = 1
  Variables de Decisión     = 135
  Desigualdades SD         = 16x16
```

Resultan 136 momentos. Encontrando la solución con `msol`:

```
>> [status.obj] = msol(P)
GloptiPoly 3.4 del 30 de Septiembre de 2008
Resolver problema de Momentos SDP
************************************************
Llamando a SeDuMi
```

```
SeDuMi 1.1R3 by AdvOL, 2006 and Jos F. Sturm, 1998-2003.
Alg = 2: xz-corrector, Adaptive Step-Differentiation, theta = 0.250, beta = 0.500
eqs m = 135, orden n = 17, dim = 257, bloques = 2
nnz(A) = 135 + 0, nnz(ADA) = 18225, nnz(L) = 9180
 it :     b*y       gap    delta  rata   t/tP*  t/tD*   feas cg cg  prec
  0 :            2.06E+002 0.000
  1 :  5.02E+003 1.96E+001 0.000 0.0950 0.9900 0.9900  -0.43  1  1  1.3E+000
  2 :  1.60E+004 3.57E+000 0.000 0.1821 0.9000 0.9000  -0.52  1  1  1.1E+000
  3 :  2.77E+004 8.00E-001 0.000 0.2240 0.9000 0.9000  -0.31  1  1  5.4E-001
  4 :  3.64E+004 2.43E-001 0.000 0.3040 0.9000 0.9000   0.00  1  1  2.8E-001
  5 :  6.30E+004 5.39E-002 0.000 0.2219 0.9000 0.9000  -0.09  1  1  1.7E-001
  6 :  9.76E+004 1.39E-002 0.000 0.2579 0.9000 0.9000  -0.09  1  1  1.0E-001
  7 :  1.74E+005 2.95E-003 0.000 0.2118 0.9000 0.9000  -0.21  1  1  6.7E-002
  8 :  2.53E+005 8.34E-004 0.000 0.2831 0.9000 0.9000  -0.12  1  1  4.0E-002
  9 :  4.53E+005 1.73E-004 0.000 0.2078 0.9000 0.9000  -0.25  1  1  2.6E-002
 10 :  6.60E+005 4.87E-005 0.000 0.2812 0.9000 0.9000  -0.14  1  1  1.6E-002
 11 :  1.17E+006 1.01E-005 0.000 0.2072 0.9000 0.9000  -0.25  1  1  1.0E-002
 12 :  1.72E+006 2.79E-006 0.000 0.2768 0.9000 0.9000  -0.16  1  1  6.0E-003
 13 :  3.03E+006 5.90E-007 0.000 0.2113 0.9000 0.9000  -0.25  1  1  3.9E-003
 14 :  4.43E+006 1.65E-007 0.000 0.2802 0.9000 0.9000  -0.15  1  1  2.4E-003
 15 :  7.75E+006 3.56E-008 0.000 0.2150 0.9000 0.9000  -0.25  1  1  1.5E-003
 16 :  1.12E+007 1.01E-008 0.000 0.2843 0.9000 0.9000  -0.14  1  1  9.2E-004
 17 :  1.97E+007 2.18E-009 0.000 0.2158 0.9000 0.9000  -0.24  1  1  6.1E-004
 18 :  2.83E+007 6.25E-010 0.000 0.2866 0.9000 0.9000  -0.13  1  1  3.6E-004
 19 :  4.99E+007 1.33E-010 0.000 0.2136 0.9000 0.9000  -0.24  1  1  2.4E-004
 20 :  7.19E+007 3.81E-011 0.000 0.2854 0.9000 0.9000  -0.13  1  1  1.4E-004
 21 :  1.27E+008 2.61E-017 0.000 0.0000 0.0000 0.9000  -0.24  1  1  1.2E-004

el Principal no es factible, corrija  la dirección encontrada del dual.
iter segundo  |Ax|    [Ay]_+     |x|       |y|
 21      0.8  1.1e-005  7.2e-009  2.5e-001  4.4e+001

Tiempo Detallado (seg)
   Ant          IPM          Post
2.344E-001    7.656E-001    1.094E-001
Norma máxima: ||b||=776, ||c|| = 1,
Cholesky |agregados|=0, |saltos| = 0, ||L.L|| = 1.
Tiempo de CPU = 1.1094 Segundos
************************************************
Solución aparente principal no viable
El problema SDP puede ser ilimitado
Trate de hacer cumplir las restricciones adicionales vinculadas
Comprobando viabilidad (eps = 1.0000e-003):
  SDP viable
Comprobando norma euclidiana de la solución (max = 1.0000e+006):
  Norma = 4.3699e+001
Comprobando los momentos de primer orden (abs tol = 1.0000e-003):
  Solución 1
    SDP objetivo = -1.0000e+000
```

```
    La solución alcanza el mismo objetivo
Optimalidad global certificada numéricamente

status =

    obj: 1
```

Así, el valor mínimo del polinomio encontrado por *Gloptipoly 3.4* es -1 y lo alcanza en el punto dado mediante la instrucción:

```
>> x = double([x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x11 x12 x13 x14 x15])

x =

  1.0e-003 *

  Columns 1 through 7

    0.0997    0.1313    0.1555    0.1592    0.0978    0.1450    0.0359

  Columns 8 through 14

    0.1270    0.1286    0.1259    0.0974    0.0773    0.1075    0.0960

  Column 15

    0.1085
```

Ejemplo 2

Del libro citado, se prueba el algoritmo para el problema 394 pag. 212.

$$f(x) = \sum_{i=1}^{20} i(x_i^2 + x_i^4)$$

bajo la restricción de la forma:

$$g(x) = \sum_{i=1}^{20} x_i^2 = 1.$$

El polinomio a optimizar es de grado 4 y 20 variables. Definiendo el polinomio y sus restricciones:

```
>> mpol x 20
>> tp2 = 0;
>> for i=1:20
      tp2 = tp2 + i*(x(i)^2 + x(i)^4);
   end;
```

```
>> g2 = 0;
>> for i=1:20
      g2 = g2 + x(i)^2;
   end;
>> g2 == 1;
```

Definiendo el problema de optimización:

```
>> P = msdp(min(tp2))
GloptiPoly 3.4 del 30 de Septiembre de 2008
Define el problema de momentos SDP
  Función objetivo válida
  Número de restricciones de soporte = 0
  Número de restricciones momentos = 0
Medida 1
  Grado = 4
  Variables = 20
  Momentos = 10626
Orden de relajación = 2
Masa de medida 1 a un conjunto
Número total de momentos = 10626
Realizar sustituciones de Momentos
Número de momentos después de sustituciones = 10625
Generar momentos y las restricciones
Generar el problema de momentos SDP

Problema de Momentos SDP
  Medida etiquetada         = 1
  Orden de relajación       = 2
  Variables de Decisión     = 10625
  Desigualdades SD         = 231x231
```

Resultan 10.626 momentos. Encontrando la solución con `msol`:

```
>> [status.obj] = msol(P)
GloptiPoly 3.4 del 30 de Septiembre de 2008
Resolver problema de Momentos SDP
************************************************
Llamando a SeDuMi
SeDuMi 1.1R3 by AdvOL, 2006 and Jos F. Sturm, 1998-2003.
Alg = 2: xz-corrector, Adaptive Step-Differentiation, theta = 0.250, beta = 0.500
??? Error using ==> sparse
Out of memory. Type HELP MEMORY for your options.

Error in ==> getsymbada at 46
    SYMBADA=sparse(ones(size(At,2),size(At,2)));

Error in ==> sedumi at 337
ADA = getsymbada(A,Ablkjc,DAt,K.sblkstart);
```

```
Error in ==> msdp.msol at 67
 [x,y,info] = sedumi(P.A,P.b,P.c,P.K,pars);
```

De esta forma, la herramienta gloptipoly no puede resolver el problema de optimización por desborde de memoria en la ejecución de SeDuMi. Más exactamente, el problema de desborde lo presenta la función `ones` de MatLab (*para más información de* `ones`, *ver MatLab Help*), la cual no se puede definir con una dimensión tan grande. Además, se puede ver que el espacio vectorial de los polinomios con 20 variables y grado menor o igual a 4 tiene dimensión 10.626, lo cual nos da una idea de que aproximadamente para una dimensión de dicho espacio mayor a 10.000, la herramienta posiblemente no es aplicable.

4.3.3. Función de verosimilitud de una GIP

A continuación se experimentará sobre la función de verosimilitud correspondiente a la gramática dada. Las muestras utilizadas han sido generadas a partir de gramáticas típicas y la función de verosimilitud obtenida mediante una herramienta de simulación.

Ejemplo 1

Dada la gramática $G = (N, \Sigma, P, A_1)$ definida como:

- $\Sigma = \{a, b\}$
- $N = \{A_1, A_2, A_3\}$
- $P = \{A_1 \rightarrow A_2A_3,\ A_2 \rightarrow A_3A_1,\ A_2 \rightarrow b,\ A_3 \rightarrow A_1A_2,\ A_3 \rightarrow a\}$

El corpus generado mediante la herramienta de simulación de gramáticas, es el siguiente:
Muestra (Corpus):

a a b a b a b b a b
b b b b a b a a b a a b b a a a b a b a a b
a b a b a a b a
b a b b b b b a a b a b b a
b b a a b a
b b b b a b b b
b b a a a b a b a b
b a b b a a
a b b a b a
b b a a a a b a b a b a

b b a a b b b a b a a b a a b a a b
b b b a b b

Del ejemplo se tienen los siguientes parámetros:

- El número de reglas corresponde el número de variables, para este caso, el número de variables es 5
- Cada variable corresponde respectivamente a la probabilidad de cada regla; es decir:

$$x_1 = p(A_1 \rightarrow A_2 A_3)$$
$$x_2 = p(A_2 \rightarrow A_3 A_1)$$
$$x_3 = p(A_2 \rightarrow b)$$
$$x_4 = p(A_3 \rightarrow A_1 A_2)$$
$$x_5 = p(A_3 \rightarrow a)$$

- La muestra (Corpus) tiene 12 elementos.

A partir de la muestra y utilizando herramientas de software[1] se obtiene la función de verosimilitud para el caso:

$$p(x) = 60x_1^{63}x_2^{23}x_3^{68}x_4^{28}x_5^{58} \qquad \text{polinomio de grado 240.}$$

El algoritmo de optimización Gloptipoly versión 3.4, entrega el siguiente resultado:

```
>> g0=60*x1^63*x2^23*x3^68*x4^28*x5^58

Polinomio Esscalar

60x1^63x2^23x3^68x4^28x5^58

>> x1==1
Scalar measure support equality
x1 == 1
>> x2+x3==1
Scalar measure support equality
x2+x3 == 1
>> x4+x5==1
Scalar measure support equality
x4+x5 == 1
>> P=msdp(max(g0));
GloptiPoly 3.4 del 30 de Septiembre de 2008
```

[1] CYK2 diseñada por el grupo de investigación para la construcción y estimación de los parámetros de una GIP

```
Define el problema de momentos SDP
  Función objetivo válida
  Número de restricciones de soporte = 0
  Número de restricciones momentos = 0
Medida 1
??? Out of memory. Type HELP MEMORY for your options.

Error in ==> D:\Omar\Trabajo de Grado\gloptipoly3\genpow.m
On line 22  ==>    v(r+1:r+rd,1) = repmat(k,rd,1);

Error in ==> D:\Omar\Trabajo de Grado\gloptipoly3\genind.m
On line 49  ==>  pow = genpow(nvar+1,d);

Error in ==> D:\Omar\Trabajo de Grado\gloptipoly3\genmom.m
On line 39  ==> genind(nvar,2*ord);

Error in ==> D:\Omar\Trabajo de Grado\gloptipoly3\@msdp\msdp.m
On line 324  ==>   nvm = genmom(mm,pindvar,ord(m));
```

El desborde de memoria que se presenta al generar el vector de potencias se debe a que la dimensión del espacio vectorial de los polinomios de 5 variables y grado menor o igual que 240 supera ampliamente el valor del máximo entero que maneja la plataforma ($240^{5^2} \approx 3{,}2 * 10^{59}$ aprox.), y por tanto no puede generar las potencias, además la capacidad de memoria se ve limitada y aunque se aumente, el programa se sigue viendo limitado por la capacidad de memoria necesaria, esto teniendo en cuenta que el algoritmo separa toda esa memoria para los momentos, aunque no la utilice.

Ejemplo 2

A partir de la gramática del ejemplo anterior con muestra:

Muestra (Corpus):

a b a a
a b a b a b
b b b a a b a b
b b a b

La muestra (Corpus) tiene 4 elementos. Mediante el algoritmo de simulación, se obtiene la función de verosimilitud para el caso:

$$p(x) = x_1^{11} x_2^3 x_3^{12} x_4^4 x_5^{10} \qquad \text{polinomio de grado 40.}$$

El algoritmo de optimización Gloptipoly versión 3.4, entrega el siguiente resultado:

```
>> mpol x1 x2 x3 x4 x5
>> g0 = x1^11*x2^3*x3^12*x4^4*x5^10;
>> x1==1;
>> x2+x3==1;
>> x4+x5==1;
>> x1<=1;
>> x1>=0;
>> x2<=1;
>> x2>=0;
>> x3<=1;
>> x3>=0;
>> x4<=1;
>> x4>=0;
>> x5<=1;
>> x5>=0;
>> P=msdp(max(g0));
GloptiPoly 3.4 del 30 de Septiembre de 2008
Define el problema de momentos SDP
  Función objetivo válida
  Número de restricciones de soporte = 0
  Número de restricciones momentos = 0
Medida 1
??? Error using ==> zeros
Product of dimensions is greater than maximum integer.

Error in ==> genind at 56
 bas = zeros(dmm,dmm,nvar);

Error in ==> genmom at 39
genind(nvar,2*ord);

Error in ==> msdp.msdp at 324
  nvm = genmom(mm,pindvar,ord(m));
```

El error usando `zeros` se da al inicializar la matriz de índices. Si bien la dimensión del espacio vectorial de los polinomios de 5 variables y grado menor o igual que 40 no supera el valor del máximo entero que maneja la plataforma, al inicializar la matriz de momentos con ceros (es lo que hace la función `zeros`), la dimensión se eleva a la potencia 5, por lo cual la función `zeros` debe manejar ese número de elementos y por tanto dicho número (aprox. $4,3 * 10^{31}$) supera también el valor del máximo entero que maneja la plataforma.

Ejemplo 3

A partir de la gramática del ejemplo anterior con muestra:
Muestra (Corpus):

a b a a
a b a b a b
b b a b

La muestra (Corpus) tiene 3 elementos. Mediante el algoritmo de simulación, se obtiene la función de verosimilitud para el caso:

$$p(x) = x_1^7 x_2^2 x_3^7 x_4^2 x_5^7 \qquad \text{polinomio de grado 25.}$$

El algoritmo de optimización Gloptipoly versión 3.4, entrega el siguiente resultado:

```
>>mpol x1 x2 x3 x4 x5;
>>fv1 = x1^7*x2^2*x3^7*x4^2*x5^7;
>> x1==1;
>> x2+x3==1;
>> x4+x5==1;
>> x1<=1;
>> x1>=0;
>> x2<=1;
>> x2>=0;
>> x3<=1;
>> x3>=0;
>> x4<=1;
>> x4>=0;
>> x5<=1;
>> x5>=0;
>> P=msdp(max(fv1));
GloptiPoly 3.4 del 30 de Septiembre de 2008
Define el problema de momentos SDP
  Función objetivo válida
  Número de restricciones de soporte = 0
  Número de restricciones momentos = 0
Medida 1
??? Error using ==> unknown
Out of memory. Type HELP MEMORY for your options.

Error in ==> genind at 56
 bas = zeros(dmm,dmm,nvar);

Error in ==> genmom at 39
genind(nvar,2*ord); % genera índice de tablas

Error in ==> msdp.msdp at 324
  nvm = genmom(mm,pindvar,ord(m));
```

El error, presentado como de origen desconocido se produce al inicializar la matriz de índices. Al igual que en el ejemplo anterior, esto se debe a que la dimensión del espacio vectorial de los polinomios de 5 variables y grado menor o igual que 25 es 142506, pero

la dimensión de la matriz de momentos es $142506^5 \approx 5{,}87 * 10^{25}$ para lo cual la cantidad de memoria a separar supera el valor del máximo entero que maneja la plataforma. Una posible solución es aumentando el tamaño de la memoria virtual, pero después de varias pruebas el resultado es el mismo.

Capítulo 5

CONCLUSIONES

- El método de optimización de polinomios en varias variables mediante la teoría de Momentos es un método novedoso que puede ser aplicable en la estimación de los parámetros de una Gramática Incontextual Probabilistica, dado que la función de verosimilitud asociada a las reglas de la GIP es un polinomio multivariado.

- Experimentalmente, la herramienta utilizada para la optimización de polinomios en varias variables encuentra muy buenas aproximaciones tanto en valores óptimos como en puntos de optimización. Existen casos en que no puede encontrar directamente los óptimos, pero se logra mediante restricciones del dominio de manera conveniente o mediante traslaciones (*ver ejemplo 4 de la sección 4.3.1*).

- El coste computacional del algoritmo se puede determinar haciendo un análisis de las funciones involucradas, dado que por su extensión y complejidad no es posible calcular un valor exacto sino que se busca determinar si su orden es mayor al polinómico (Ver numeral siguiente).

- Para optimizar un polinomio $p(x) : \mathbb{R}^n \mapsto \mathbb{R}$ de grado m, tanto teórica como experimentalmente el algoritmo necesita generar una base del espacio vectorial de los polinomios $q(x) : \mathbb{R}^n \mapsto \mathbb{R}$ de grado menor o igual que m, la dimensión se puede determinar mediante la expresión:

$$D_n(m) = \sum_{i=0}^{m} D_{n-1}(i)$$

  donde $D_k(j)$ es la dimensión del espacio vectorial de los polinomios $q(x) : \mathbb{R}^k \mapsto \mathbb{R}$ de grado menor o igual que j, $D_1(j) = j + 1$ y $D_k(0) = 1$; para todo $1 \leq k \leq n$, $0 \leq j \leq m$. De esta forma, el coste computacional para construir la base es de orden $O(m^n)$.

- Una base del espacio vectorial de los polinomios de variable en $\mathbb{R}^n$ y grado menor o igual a m es el conjunto B definido como en la sección 3.1. Entonces, si la dimensión de B es d ($d = dim(B)$), la matriz de momentos $M_m(y)$ definida como en (3.7) es una matriz simétrica de tamaño d^n. Como d es de orden $O(m^n)$, se sigue que el tamaño de la matriz $M_m(y)$ es de orden $O((m^n)^n) = O(m^{n^2})$. Por tanto, un algoritmo de construcción de la matriz de momentos es de orden $O(m^{n^2})$.

- De la experimentación con gramáticas, se puede ver que a mayor número de elementos en la muestra, mayor será el grado del polinomio obtenido (función de verosimilitud). Por lo tanto, el polinomio a optimizar depende tanto del número de reglas como del tamaño de la muestra, y entre mayor complejidad tenga la gramática, mayor serán tanto el número de variables como el grado del polinomio.

- Una Gramática Incontextual Probabilística (GIP) en general posee un elevado número de reglas (más de 100 en una GIP elemental). Luego, la función de verosimilitud tendrá tantas variables como reglas; por otro lado, el tamaño de la muestra (corpus) debe ser grande(más de 100 elementos), esto implica que el grado del polinomio es bastante grande; razón por la cual al implementar el problema mediante Gloptipoly, se genera un desborde de memoria. Por ejemplo, para la GIP definida en el ejemplo 1 de la sección (4.3.3), se tienen 5 reglas y una muestra de 12 elementos. La función de verosimilitud obtenida mediante la herramienta de simulación para Gramáticas es un polinomio (más exactamente un monomio) de 5 variables y grado 240, por lo cual la dimensión del espacio vectorial de los polinomios de variable en $\mathbb{R}^5$ y grado menor o igual a 240 es aproximadamente $240^5 = 7{,}962624 \times 10^{11}$ y la dimensión de la matriz de momentos de orden 240 es aproximadamente $240^{5^2} = 240^{25} = 3{,}20096586 \times 10^{59}$, valores demasiado altos e imposibles de manejar con esta herramienta.

- La herramienta Gloptipoly no es aplicable en la estimación de los parámetros de una GIP, dado que por su elevado coste computacional el algoritmo genera un desborde de memoria aún bajo gramáticas con un número mínimo de reglas y elementos en la muestra, a pesar de obtener muy buenas aproximaciones en polinomios para los cuales la base del espacio vectorial que genera el polinomio no tenga dimensión superior a 10.000 aproximadamente.

- Se han encontrado polinomios para los cuales el punto o los puntos donde ocurren los óptimos no están en el origen y la herramienta ha encontrado el óptimo pero no los puntos donde lo alcanza.

- En un estudio futuro es posible modificar la herramienta aplicando otros métodos de programación y almacenamiento, de tal manera que se puedan tratar polinomios en un espacio de mayor dimensión, o en su defecto, implementar el algoritmo en otra plataforma de mayor capacidad.

Bibliografía

[1] Boyd, S., Ghaoui, L. E., Feron, E., y Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, Studies in Applied Mathematics, 15, SIAM, Philadelphia, Pennsylvania. V. (1994).

[2] Bolfarine, H. and Carneiro M., *Introducción a la Inferencia Estadística*, Primera Edición 2005.

[3] C. Berg, *The multidimensional moment problem and semi-groups*, in Moments in Mathematics, H.J. Landau, ed. AMS, Providence. (1980).

[4] Devore, J. L., *Probabilidad y Estadística para Ingeniería y Ciencias*, Quinta Edición 2005.

[5] Didier Henrion, Jean Bernard Lasserre, *GloptiPoly: Global Optimization over Polynomials with Matlab and SeDuMi*, Version 2.3.0 de Marzo 11 de 2005.

[6] Edgar M. Carreño, Eliana M. Toro, Antonio Escobar, *Optimización de Sistemas Lineales usando Métodos de Punto Interior*, Scientia et Technica Año X, No 24, Mayo 2004. UTP. ISSN 0122-1701.

[7] Gahinet, P, Nemirovski, A, Laub, A. J, and Chilali, *LMI Control Toolbox For Use with MATLAB User's Guide*, The MathWorks, Natick, Massachusetts. M. (1995).

[8] Imre Pólik, *Addendum to the SeDuMi user Guide Versión 1.1*, Basado en la guía original de SeDuMi y el conternido de la ayuda de SeDuMi. (2005).

[9] Jean L. Lasserre, *Global Optimization with Polynomials and the Problem of Moments*, Society for Industrial and Applied Mathematics, vol. 11 No. 3 (2001), pág 796-817.

[10] Joan Andreu Sánchez, *Estimación de Gramáticas Incontextuales Probabilísticas y su aplicación en modelización del lenguaje*, Tésis doctoral Universidad politécnica de Venecia (1999).

[11] Klaus Schittkowski, *More Test Examples for nonlinear Programming Codes*, Lecture Notes in Economics and Mathematical Systems. Ed. Springer-Vertaq. No. 282.

[12] Myriam Muñoz de Özak, Liliana Blanco Castañeda, *Introducción a la Teoría Avanzada de la Probabilidad*, Universidad Nacional de Colombia, Sede Bogotá. Primera Edición. (2002).

[13] Proyecto GIP, documento proyecto *Nuevas alternativas para la estimación de los parámetros en una gramática incontextual probabilística*, Departamento de Matemáticas, Universidad del Cauca. (2004).

[14] Seminario de Investigacion, *Notas de exposiciones en el seminario*, 2004-2006.

[15] Y. Nesterov, *Squared functional systems and optimization problems*, In High Performance optimization, H. Frenk, K. Roos, T. Terlaky, and S. Zhang, eds., Kluwer, Dordrecht, 2000.