Facultad de Matemáticas

Trabajo de fin de Grado

Doble grado en Física y Matemáticas

Optimización de polinomios no negativos

Autor: Julia Vázquez Escobar

Tutor: Manuel Jesús Gago Vargas

Junio de 2019

Resumen

La optimización es un campo fundamental en el desarrollo tecnológico actual. Sus herramientas permiten realizar avances en robótica, economía financiera o regresión estadística. Presentamos una nueva perspectiva con la que tratar los problemas de optimización sobre polinomios no negativos, que consisten en minimizar o maximizar imponiendo restricciones equivalentes a que uno o varios polinomios sean no negativos. La formulación *sos* (sum of squares) ha supuesto un gran avance para este tipo de optimización en los últimos años ya que,como veremos, se puede solventar usando programación semidefinida. En este trabajo, definimos este tipo de programación y estudiamos las cualidades de os polinomios que son suma de cuadrados, así como su relación con los polinomios no negativos. Sin embargo, para problemas con un gran número de variables, la formulación *sos* se ve limitada y no es capaz de proporcionar una solución. Es por ello que introducimos la optimización *dsos* y *sdsos* como alternativa. Se basa en el uso de polinomios de las clases homónimas. Éstos cumplen estar contenidos en la clase *sos* por lo que perdemos, en principio, región factible y, por tanto, fiabilidad en la solución. Esta pérdida de precisión se compensa ya que veremos que optimizar sobre polinomios *dsos* y *sdsos* se puede realizar usando programación lineal y cónica de segundo orden, respectivamente. Esto supone una mejora en eficiencia ya que, si usamos métodos de punto interior, la programación lineal y cónica de segundo orden son menos complejas de resolver que la programación semidefinida. Hablamos de una técnica que nos permite reducir el tiempo de computación significativamente. Además, en algunos casos en los que la programación *sos* no puede dar una solución, la programación *dsos* y *sdsos* sí nos proporciona una.

Abstract

Optimization is a fundamental field in current technological development. Its tools allow advances in robotics, financial economics or statistical regression. We present a new perspective with which to deal with optimization problems over non-negative polynomials. These problems consist of minimizing or maximizing while imposing restrictions equivalent to one or several polynomials being non-negative. *Sos* (sum of squares) optimization has made important progress in this area because, as we will see, it can be solved using semidefinite programming. In this paper, we introduce this type of programming and study the characteristics of *sos* polynomials, as well as their relationship with non-negative polynomials. However, for problems with a large number of variables, *sos* formulation is limited and is not able to provide a solution. That is why we present the *dsos* and *sdsos* optimization as an alternative. It is based on the use of polynomials of homonymous classes. These are contained in the *sos* polynomials set so we lose, in principle, feasible region and, therefore, reliability in the solution. This loss of precision is compensated since, as we will see, optimizing on *dsos* and *sdsos* polynomials can be done using linear and second order conic programming , respectively. This means that we are improving the efficiency since linear and second order conic programming are less complex to solve than semidefinite programming, if we use interior point methods. We are talking about a technique that allows us to reduce computing time significantly. Also, in some cases in which *sos* programming can not give a solution, *dsos* and *sdsos* programming does provides one.

Índice general

Introducción

El objetivo de este trabajo es presentar una alternativa a las técnicas usadas hasta ahora para resolver problemas de optimización sobre polinomios no negativos. Nos basamos en el artículo [AM19] que estudia la optimización *dsos* y *sdsos*.

La optimización sobre polinomios no negativos consiste en minimizar o maximizar una cantidad imponiendo restricciones que se pueden expresar como la condición de que uno o varios polinomios sean no negativos. En el capítulo 1 presentamos distintas formulaciones de las que haremos uso. Introducimos la programación lineal, cónica de segundo orden, semidefinida y cónica como generalización de las anteriores. Además se estudia cómo, usando métodos de punto interior, resolver problemas de programación lineal y cuadráticos de segundo orden es más eficiente que resolver problemas de programación semidefinida.

El objetivo es, por tanto, obtener un método que verifique si un polinomio es no negativo. En el capítulo 2 se explora esta posibilidad. Veremos que para algunos casos la estructura del polinomio a tratar nos permite obtener una caracterización, pero en general esto no ocurre.

Hallar un método de verificación que sirva en un amplio rango de situaciones se vuelve complicado. Es por ello que hablamos de los polinomios que son sumas de cuadrados. En los últimos años el campo de la optimización se ha visto influido por el advenimiento de la optimización *sos*. Esta técnica consiste en sustituir la condición de que un polinomio sea no negativo, por la restricción

de que cumpla ser suma de cuadrados. Un polinomio de estas características es siempre no negativo, aunque la implicación no se cumpla en general. Esta situación supone que al hacer el intercambio reduzcamos el conjunto de soluciones factibles en principio, sin embargo, probaremos que optimizar sobre polinomios *sos* se puede solventar mediante programación semidefinida. Esto supone que contamos con un método general que, en principio, puede solventar nuestro problema.

Como alternativa a esta aproximación, presentamos en el capítulo 3 dos nuevas clases de polinomios, *dsos* y *sdsos*. Se cumple que $dsos, sdsos \subseteq sos$. Veremos que esta condición es exclusiva por lo que si imponemos que un polinomio pertenezca a estas dos nuevas clases en vez de a *sos* estaremos siendo más restrictivos. Sin embargo probaremos que optimizar sobre polinomios *dsos* y *sdsos* es equivalente a resolver un problema de programación lineal y uno cónico de segundo orden respectivamente. Esto presenta una ventaja en cuanto a eficiencia si empleamos un método de punto interior como resolutor.

Para polinomios con una gran cantidad de variables esta ganancia en eficiencia compensa la posible reducción del conjunto de soluciones factibles. Esto se debe a que en casos de este tipo la formulación *sos* tarda mucho tiempo en proporcionar una solución o ni siquiera es capaz de dar una.

Finalmente, en el capítulo 4 se ofrecen algunas aplicaciones de esta nueva técnica. Veremos como al calcular el mínimo de un polinomio las formulaciones *dsos* y *sdsos* proporcionan soluciones más rápidas que la *sos*, y para más de 30 variables dan solución cuando la formulación *sos* no puede por limitaciones de memoria RAM. Otra importante aplicación es en el ámbito de teoría de control. Mediante estas técnicas veremos cómo calcular funciones de Lyapunov que nos garanticen estabilidad global para un sistema dinámico o, en su defecto, calcular regiones de atracción asociadas a funciones de Lyapunov dadas.

Usando resultados de álgebra del siglo XIX, cuando los matemáticos se preguntaban si la no negatividad en los polinomios equivalía a ser suma de cuadrados, se han logrado hacer avances en optimización que influyen enorme-

mente en el ámbito de la robótica. Gracias al desarrollo conseguido hace más de cien años podemos ahora mejorar el funcionamiento de robots, así como drones y coches automáticos. Un importante avance en disciplinas actuales que se consigue con herramientas matemáticas de hace dos siglos. Se deja el enlace a un artículo [1] que trata este tema.

[1] https://www.quantamagazine.org/a-classical-math-problem-gets-pulled-into-the-modern-world-20180523/

Capítulo 1

Problemas de optimización

En esta sección introducimos distintas formulaciones que forman parte de la programación convexa. Con estas herramientas podemos plantear gran parte de los problemas de optimización de hoy en día. Hablaremos sobre programación lineal, cónica de segundo orden y semidefinida. Estos problemas se pueden generalizar al caso de optimización sobre conos, como veremos más tarde, y guardan ciertas similitudes y relaciones de inclusión entre sí.

1.1. Programación Lineal

Se ha estudiado, en el curso de Programación Matemática del grado, la programación lineal. Los contenidos de esta sección se basan en los apuntes de la asignatura.

La optimización sobre restricciones de tipo lineal puede parecer limitada, pero existen muchos problemas que se pueden formular con condiciones de este tipo. Los algoritmos de solución para estos planteamientos están muy estudiados, por lo que disponemos de mayor capacidad resolutiva que para otras cuestiones. Tanto estudios teóricos (sistemas de ecuaciones, método de Gauss...) como algoritmos diseñados para ordenadores (SIMPLEX,...) han sido profundamente desarrollados y perfeccionados. Estudiamos este caso, ya que si exigimos ciertas características a nuestro polinomio, convertimos la comproba-

ción de restricciones en un sistema lineal. Es intuitivo pensar que la resolución del problema transformado será mas eficaz y rápida a medida que aumentemos la complejidad de los datos. Por estos motivos nos detenemos a explicar con detalle esta rama de la optimización.

Antes de tratar formalmente un problema de programación lineal vamos a definir unos conceptos que tendremos que usar. La noción de poliedro, y cómo se puede construir nos va a resultar útil a la hora de generar un algoritmo que nos permita hallar máximos o mínimos de una función lineal sobre dicho conjunto.

Definición 1.1.1 *Un conjunto $S \subset \mathbb{R}^n$ se denomina poliedro si existen una matriz $A \in \mathbb{R}^{m\times n}$ y un vector $b \in \mathbb{R}^m$ tal que $S = \{x \in \mathbb{R}^n \mid Ax \leq b\}$. Un poliedro acotado se llama politopo.*

Se aclara la notación $Ax \leq b$, donde se entiende que Ax es menor o igual que b componente a componente.

Ejemplo 1.1.1 *Consideremos el sistema de ecuaciones siguiente*

$$Ax \leq b \quad con \quad A = \begin{pmatrix} 5 & -1 \\ 2 & 1 \\ 1 & -2 \end{pmatrix} \quad y \quad b = \begin{pmatrix} 10 \\ 5 \\ 6 \end{pmatrix}$$

siendo $x \in \mathbb{R}^2$. Obtenemos el poliedro de la imagen 1.1

Definición 1.1.2 *Sea $S \subset \mathbb{R}^n$. Decimos que $x_0 \in S$ es punto extremo de S si no existen $y, z \in S$, con $y \neq z$ tales que $x_0 = (1-\lambda)y + \lambda z$, $\lambda \in (0,1)$.*

Observamos que un poliedro cumple ser un conjunto convexo. Efectivamente si dos puntos x, y cumplen $Ax \leq b$ y $Ay \leq b$, otro de la forma $z = \lambda x + (1-\lambda)y$ con $\lambda \in (0,1)$ cumplirá $Az = A(\lambda x + (1-\lambda)y) = \lambda Ax + (1-\lambda)Ay \leq \lambda b + (1-\lambda)b = b$. Además un poliedro tiene, a lo sumo, $\binom{m}{n}$ puntos extremos.

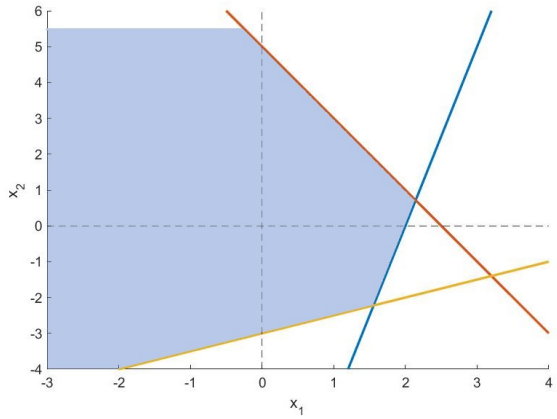

Figura 1.1

Ejemplo 1.1.2 *Veamos un ejemplo con el máximo número de puntos extremos. Si tenemos el sistema*

$$Ax = \begin{pmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} = b$$

Resulta en el poliedro de la imagen 1.1.2. Efectivamente $4 \leq \binom{4}{2}$.

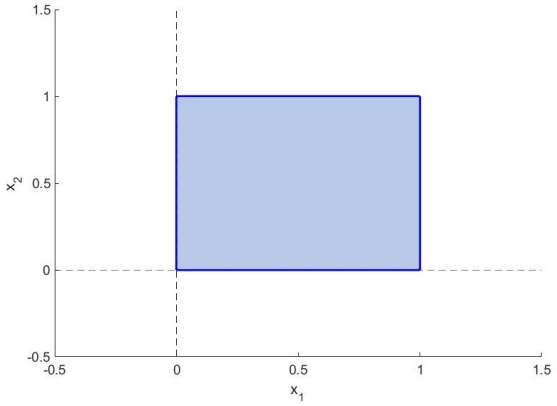

Figura 1.1.2

Para hacernos una idea de la estructura de nuestro conjunto vamos a enunciar el siguiente teorema. Para más información sobre la representación de poliedros y, en concreto este resultado, nos referimos a [MSB77, sección 2.6].

Teorema 1.1.1 ***Representación de poliedro .*** *Sea $S \subset \mathbb{R}^n$ un poliedro. Existen conjuntos $\{v^1, ..., v^p\} \subset S$ y $\{d^1, ..., d^q\} \subset \mathbb{R}^n$ tales que:*

$$S = \{x \in \mathbb{R}^n : x = \sum_{i=1}^{p} \lambda_i v^i + \sum_{j=1}^{q} \mu_j d^j, \lambda_i \geq 0 \ \forall i, \mu_j \geq 0 \ \forall j, \ \sum_{i=1}^{p} \lambda_i = 1\}$$

Definamos un *cono* como un conjunto $P \subset \mathbb{R}^n$ que cumple $\{\lambda x : \lambda \geq 0\} \subset P, \ \forall x \in P$ y tomemos el cono generado por $\{d^1, ..., d^q\}$ como:

$$cono(\{d^1, ..., d^q\}) = \{d \in \mathbb{R}^n : d = \sum_{j=1}^{q} d^j \mu_j, \mu_j \geq 0 \ \forall j\}$$

Con estas definiciones podemos interpretar el teorema anterior diciendo que nuestro poliedro se puede expresar como la suma de dos conos generados.

$$S = cono(\{d^1, ..., d^q\}) + cono(\{v^1, ..., v^p\})$$

Observamos que, en general, no podemos tomar el conjunto $cono(\{v^1, ..., v^p\})$ como el conjunto de puntos extremos de S. En efecto, si consideramos $S = \mathbb{R}$, es un poliedro sin puntos extremos.

Ahora sí podemos formular un *problema de programación lineal*. Es un problema de optimización que se expresa de la siguiente forma, siendo S un poliedro.

$$\begin{array}{ll} \min\limits_{x} & c^T x \\ \text{s.a.} & x \in S \end{array} \tag{1.1}$$

Puede que se nos presente un problema lineal que no esté formulado exactamente igual que éste, pero si hacemos uso de las variables de holgura y de partición, siempre llegamos a la forma indicada anteriormente. Vamos a caracterizar ahora las soluciones de nuestro problema, para hacernos a la idea de cómo podemos abordarlo.

Definición 1.1.3 *Una solución básica del poliedro* $\{x \in \mathbb{R}^n \mid Ax \leq b, x \geq 0\}$ *es cualquier* $x \in \mathbb{R}^n$ *tal que existen un subconjunto de los índices* $I \subset \{1, ..., n\}$ *con* $|I| = m$ *cumpliendo que* $\{a_i : a_i \in A, i \in I\}$ *es linealmente independiente y* $\sum_{i \in I} a_i x_i = b, \quad x_j = 0 \ \ \forall j \notin I$. *Decimos que es además factible si se cumple* $x \geq 0$.

Proposición 1.1.1 *Sea* $A \in \mathbb{R}^{m \times n}$, *con rango* $m \leq n$, *y* $b \in \mathbb{R}^m$. *Consideramos el poliedro* $S = \{x \in \mathbb{R}^n \mid Ax = b\}$. *Dado* $x \in S$, *son equivalentes:*

1. x *es solución básica factible.*

2. x *es punto extremo de* S.

Esta proposición se ha demostrado en el curso de Programación matemática, para más información [MSB77, p.90-92].

Pongamos un ejemplo, si tenemos el poliedro $S = \{x_1 = -1, \ x_1 + x_2 = -1, \ x_1 - x_2 = 1\}$ tiene como puntos extremos $(-1, 0)$ y $(0, -1)$, lo que los convierte en candidatos a solución factible de un posible problema de optimización que se nos presente.

Existen además resultados de existencia de solución básica factible, y un método denominado método SÍMPLEX que nos garantiza encontrar dicha solución de nuestro problema de optimización [DOW55].

Si al problema descrito hasta ahora lo denominamos *primal*, definamos a partir de él lo que sería el problema *dual* que nos va a resultar muy útil en el tratamiento de éste tipo de problemas.

$$\begin{array}{ll} \min\limits_{x} & c^T x \\ \text{s.a.} & Ax = b \\ & x \geq 0 \end{array} \qquad (1.2) \qquad\qquad \begin{array}{ll} \max\limits_{x} & b^T y \\ \text{s.a.} & y^T A \leq c^T \\ & y \in \mathbb{R}^m \end{array} \qquad (1.3)$$

Nos fijamos de que se trata de otro problema de programación lineal. Algunas de las propiedades que relacionan ambos problemas son las siguientes.

Nota 1.1.1 *Para cualquier solución factible x del problema primal, y y del problema dual, se cumple que*

$$c^T x - b^T y = x^T c - (Ax)^T y = x^T (c - A^T y) \geq 0$$

donde usamos las condiciones de factibilidad de ambos problemas para concluir la desigualdad. Podemos, por tanto, obtener una cota inferior para el problema primal a partir del dual y una superior para el dual a partir del primal. Esto se denomina dualidad débil.

Formulemos ahora el resultado de dualidad fuerte.

Teorema 1.1.2 ***Dualidad fuerte****. Si en el problema primal se alcanza solución óptima en x^* entonces el dual alcanza su solución óptima en y^* con $c^T x^* = b^T y^*$.*

PRUEBA: Lo probamos por construcción. Sea x^* la solución básica óptima primal asociada a una base B, esta base es la matriz formada con las columnas de A, determinadas por el conjunto I 1.1.3. Las columnas restantes forman la matriz N. Denotamos como c_B^T al vector de los elementos de c indexados según I asociado a x^*.

Determinamos $y^* = c_B^T B^{-1}$. Probemos que es solución óptima del problema dual. Para ello hay que demostrar que $b^T y^* \leq b^T y \ \forall \ y$ solución factible del

problema dual y que y^* es factible, es decir, $y^{*T}A \geq c^T$.

Primeramente

$$b^T y^* = y^{*T} b = c_B^T B^{-1} b = c^T x^* \leq b^T y$$

para cualquier y factible en el dual. Recordemos $x^* = \begin{bmatrix} B^{-1}b \\ 0 \end{bmatrix}$. La desigualdad se da por la propiedad de dualidad débil.

Veamos que la solución propuesta es factible

$$y^{*T}A = c_B^T B^{-1} A = c_B^T B^{-1}[B : N] = [c_B^T : c_B^T B^{-1} N] \geq c^T$$

ya que $c_B^T \geq c_B^T$ y $c_B^T B^{-1} N \geq c_N^T$. Tenemos, por tanto que $y^{*T}A \geq c^T$.

Comprobemos que x^* y y^* cumplen las condiciones de holgura. Se tienen las tres condiciones siguientes

$$\begin{gathered} x^* \quad \text{factible primal} \quad \Rightarrow x_i^* \geq 0 \quad \forall i \\ y^* \quad \text{factible dual} \quad \Rightarrow y^{*T} a_i - c_i \geq 0 \quad \forall i \\ c^T x^* = b^T y^* \end{gathered}$$

por lo que

$$0 = y^{*T} b^T - c^T x^* = y^{*T} A x^* - c^t x^* = \sum_i x_i^* (y^{*T} a_i - c_i) \geq 0$$

Cada uno de los sumando es mayor o igual que 0, por lo que todos los sumandos deben ser nulos. Se cumple la condición de complementaridad de holgura $x_i^* y^{*T} a_i - c_i = 0 \quad \forall i$.

□

Veamos un ejemplo. Queremos resolver el problema siguiente,

$$\begin{array}{ll} \min\limits_{x,y} & -x-y \\ \text{s.a.} & Ax \leq b \end{array}$$

$$\text{con } A = \begin{pmatrix} 1 & -1 \\ 0{,}3 & 1 \\ 10 & -1 \end{pmatrix} \text{ y } b = \begin{pmatrix} 3 \\ 5 \\ 30 \end{pmatrix}$$

El poliedro resultante es

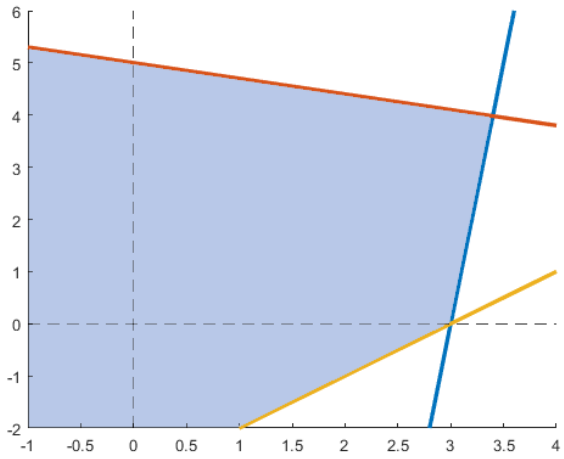

Figura 1.1

Es fácil obtener la solución visualmente. Será el vértice superior que aparece en la imagen. Si resolvemos el problema, igualmente obtenemos que la solución óptima es $(3{,}398, 3{,}981)$ con valor óptimo $-7{,}379$. Al resolver el dual se obtiene que la solución óptima es $-28{,}38$. Esto se debe a que si formulamos el problema en la forma general $Ax = b$ resulta que no existe solución factible.

1.2. Programación cónica de segundo orden

Definamos otro tipo de programación convexa que tendremos que usar. La programación cónica de segundo orden (SOCP, correspondientes a las siglas en inglés). Consiste en minimizar una función linear restringiendo nuestras variables a la intersección de una variedad linear afín con el producto cartesianos de conos de Lorentz, también denominados conos de segundo orden. Un cono de Lorentz viene definido por

$$\mathcal{L}^n = \{\underline{x} = (x_0, \underline{x_1}) \in \mathbb{R}^n : x_0 \geq ||\underline{x_1}||\} \tag{1.4}$$

usando la norma euclídea.

Efectivamente es un cono ya que si multiplicamos un elemento que pertenezca al conjunto por un escalar positivo λ obtenemos $x_0 \geq ||\underline{x_1}|| \Rightarrow \lambda x_0 \geq \lambda||\underline{x_1}||$. Además es cerrado ya que las restricciones que se imponen no son estrictas.

Formularemos un problema de programación de este tipo (SOCP)

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{s.a.} \quad & Ax = b \\ & x \in \mathcal{L}^n \end{aligned} \tag{1.5}$$

siendo $\mathcal{L}^n$ un cono de Lorentz. Debe cumplirse $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ y $b \in \mathbb{R}^m$.

Es interesante observar que si escogemos imponer $x_i \geq 0$ tenemos un problema de programación lineal, pero esto no son más que conos de Lorentz de dimensión 1, por lo que los problemas de programación lineal son casos particulares de los de programación cónica de segundo orden.

Dado que un cono de Lorentz en dos dimensiones sería $\mathcal{L}^n = \{(x_0, x_1) \in \mathbb{R}^2 : x_0 \geq |x_1|\}$, que es una rotación del primer cuadrante, es claro que un problema SOCP en el que todos los conos de Lorentz sean de 1 o 2 dimensiones se puede transformar en uno de programación lineal.

Un ejemplo en el que podemos aplicar este tipo de formulación es el problema de minimización de la suma de las normas. Podemos expresar el problema de minimizar $\sum_{i=1}^{m} ||\underline{v_i}||$ como

$$\begin{aligned} \min \quad & \sum_{i=1}^{r} v_{i0} \\ \text{s.a.} \quad & \underline{v_i} \in \mathcal{L}^n \ \ \forall i = 1..m \end{aligned} \tag{1.6}$$

1.3. Programación Semidefinida

Esta formulación resulta relevante en el ámbito de la optimización ya que, como veremos, la programación semidefinida nos permite saber si podemos expresar un polinomio como suma de cuadrados (lo que implica directamente su no negatividad). Estudiamos en esta sección cómo se plantean este tipo de problemas y las cualidades que poseen. Usamos como referencia [Par13] donde se profundiza más en este tipo de programación.

Para definir un problema de programación semidefinida vamos a presentar antes algunos conceptos.

Definición 1.3.1 *Una desigualdad matricial linear es de la forma*

$$A_0 + \sum_{i=1}^{m} A_i x_i \succeq 0$$

con $A_i \in \mathcal{S}^n$ matrices simétricas $\forall i = 0, ..., m$. La notación $M \succeq 0$ ($\succ$) indica que la matriz M tiene que ser semidefinida (definida) positiva.

Definición 1.3.2 *Un conjunto $S \subset \mathbb{R}^m$ es un espectaedro si se puede expresarse de la siguiente manera:*

$$S = \left\{ (x_1, ..., x_m) \in \mathbb{R}^m : A_0 + \sum_{i=1}^{m} A_i x_i \succeq 0 \right\}$$

siendo $A_i \in \mathcal{S}^n$ matrices simétricas $\forall i = 0, ..., m$.

La desigualdad matricial que define al espectaedro equivale a imponer infinitas condiciones del tipo $v^T \left(A_0 + \sum_{i=1}^{m} A_i x_i\right) v \geq 0$, una por cada $v \in \mathbb{R}^n$ posible. Cada una de esta desigualdades representa un conjunto cerrado convexo, por lo que la intersección resultante de aplicar infinitas condiciones de este tipo es un conjunto cerrado convexo. Podemos interpretar geométricamente un espectaedro como la intersección de un cono semidefinido positivo y un subespacio afín, el generado por $A_1, \ldots, A_m$, trasladado a A_0.

Señalamos que un poliedro es un caso específico de espectaedro; efectivamente, si escogemos las matrices A_i de manera que sean diagonales, obtenemos un poliedro.

Veamos un ejemplo de este tipo de conjuntos. Sea $A \in \mathbb{R}^2$ dado por los pares $(x, y) \in \mathbb{R}^2$ tales que:

$$\begin{bmatrix} x+1 & 0 & y \\ 0 & 2 & -x-1 \\ y & -x-1 & 2 \end{bmatrix} \succeq 0$$

En la imagen se representa el espectaedro. Corresponde a la zona sombreada de azul claro.

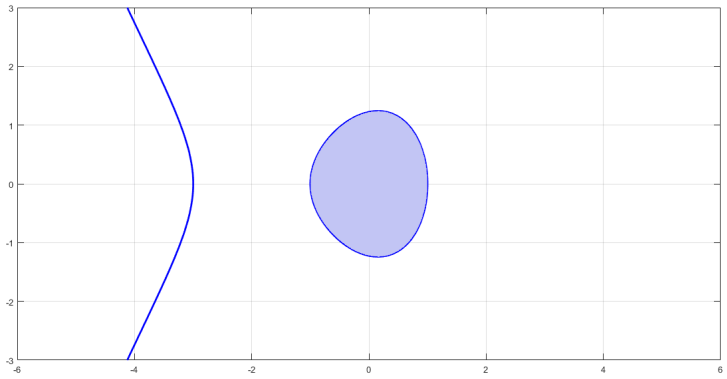

Imagen 2.1

Hemos visto que podemos ver un espectaedro como un conjunto cerrado convexo en $\mathbb{R}^m$. Igualmente podemos expresarlo como $\{A_0 + \sum_{i=1}^m A_i x_i | x \in \mathbb{R}^m\} \cap \mathcal{S}_+^n$ en el espacio de matrices simétricas, y también tenemos que es un conjunto convexo. Si se cumple que las matrices A_i son linealmente independientes los dos conjuntos convexos (en $\mathbb{R}^m$ y en $\mathcal{S}_+^n$) son afinmente equivalentes.

Espectaedros proyectados. Definimos los *espectaedros proyectados* que son proyecciones lineales de dichos conjuntos.

Definición 1.3.3 *Decimos que un conjunto $S \subset \mathbb{R}^m$ es un espectaedro proyectado si tiene la forma*

$$\left\{(x_1, \ldots, x_m) \in \mathbb{R}^m : \exists\, (y_1, \ldots, y_p) \in \mathbb{R}^p, A_0 + \sum_{i=1}^m A_i x_i + \sum_{j=1}^p B_j y_j \succeq 0\right\}$$

donde $A_0, A_1, \ldots, A_m, B_1, \ldots, B_p$ son matrices simétricas dadas.

En general un espectaedro proyectado no tiene por qué ser un espectaedro. Esta característica se contrapone al caso de los poliedros, ya que la proyección lineal de estos conjuntos siempre son poliedros.

Ejemplo 1.3.1 *Tomamos el espectaedro definido por*

$$S = \{(x, y) \in \mathbb{R}^2 : \begin{bmatrix} x & 1 \\ 1 & y \end{bmatrix} \succeq 0\} \quad \Rightarrow \quad x \geq 0, \quad xy \geq 1 \tag{1.7}$$

El conjunto correspondiente es la zona sombreada de la imagen

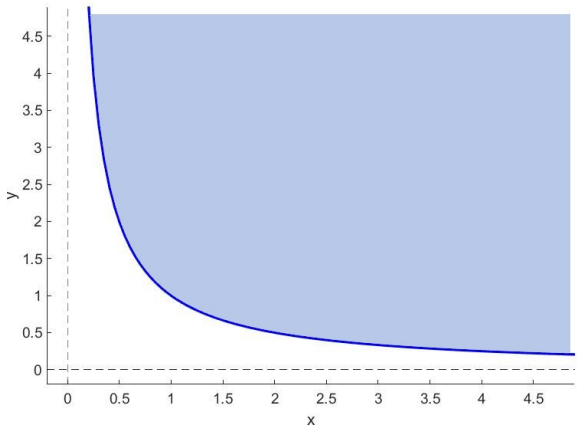

Figura 1.3.1

Al proyectar el resultado sobre el eje x queda $S_x = \{x \in \mathbb{R} : x > 0\}$ que es un conjunto no cerrado, por lo que no puede ser un espectaedro.

Si un problema se nos presenta en forma de espectaedro proyectado, podremos añadir variables de holgura para expresarlo como un espectaedro, y resolver así el problema usando programación semidefinida. Por tanto este tipo de conjuntos son de gran interés en el ámbito de la optimización.

Con estos conceptos ya podemos formular el problema general de programación semidefinida.

1.3.1. Formulación SDP primal

Los programas semidefinidos son problemas de optimización lineal en conjuntos que son espectaedros. Se expresan como:

$$\begin{aligned} \min_{X} \quad & \langle C, X \rangle \\ \text{s.a.} \quad & \langle A_i, X \rangle = b_i, \quad i = 1, ..., m \\ & X \succeq 0 \end{aligned} \tag{1.8}$$

siendo $C, A_i \in \mathcal{S}^n$ datos dados y $X \in \mathcal{S}^n$ la variable a optimizar. Tenemos en cuenta que $\langle X, Y \rangle = Tr(X^T Y) = \sum_{ij} X_{ij} Y_{ij}$. La variable sobre la que optimizamos es la matriz $X \in \mathcal{S}^n$.

El conjunto de soluciones factibles es la intersección del cono semidefinido positivo $\mathcal{S}^n_+$ y un subespacio afín, ambos conjuntos convexos. Por tanto, el conjunto de matrices X que satisfacen las restricciones es un espectaedro. Al contrario que en programación lineal el conjunto de soluciones factible no es un poliedro. Veamos un ejemplo de este tipo de problemas.

Ejemplo 1.3.2 *Consideramos el problema*

$$\begin{array}{ll} min & 2x_{11} + 2x_{12} \\ s.a. & x_{11} + x_{22} = 1 \\ & \begin{bmatrix} x_{11} & x_{12} \\ x_{12} & x_{22} \end{bmatrix} \succeq 0 \end{array}$$

Es un problema del tipo SDP-P claramente con m=1 y

$$C = \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix}, \quad A_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad b_1 = 1$$

Las condiciones impuestas sobre X se pueden reducir a que se cumpla $x_{11}(1 - x_{11}) \geq x_{12}^2$. Por tanto, el conjunto de soluciones factibles es un disco cerrado, no un poliedro. Se obtiene que la solución óptima es

$$X^\star = \begin{bmatrix} \frac{2-\sqrt{2}}{4} & -\frac{1}{2\sqrt{2}} \\ -\frac{1}{2\sqrt{2}} & \frac{2+\sqrt{2}}{4} \end{bmatrix}$$

con valor óptimo $1 - \sqrt{2}$ que no es racional. Vemos en la siguiente imagen una representación del conjunto factible.

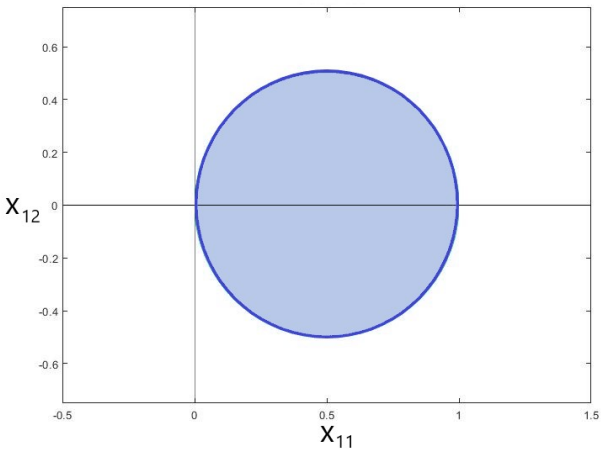

Figura 1.3.1

Comprobamos por tanto, que aún siendo expresado el problema con datos racionales, se puede obtener una solución irracional. Las soluciones son, aún así, algebraicas.

1.3.2. Dualidad

Definamos el problema dual asociado al primal ya definido 1.8. Sean $b = (b_1, \ldots, b_m)$, y $y = (y_1, \ldots, y_m)$ variables de decisión dual. Formulamos el SDP-D como se muestra:

$$\begin{aligned} \max_{y} \quad & b^T y \\ \text{s.a.} \quad & \sum_{i=1}^{m} A_i y_i \preceq C \end{aligned} \tag{1.9}$$

La teoría dual, que estudia el comportamiento de este problema y su relación con el primal, es relevante desde el punto de vista teórico y aplicado.

Análogamente al caso de programación lineal, se usan las propiedades del problema dual para acotar la solución del primal y viceversa. Sean X e y dos soluciones factibles cualesquiera del problema primal y dual respectivamente. Se cumple lo siguiente:

$$\langle C, X\rangle - b^T y = \langle C, X\rangle - \sum_{i=1}^{m} y_i \langle A_i, X\rangle = \left\langle C - \sum_{i=1}^{m} A_i y_i, X \right\rangle \geq 0 \quad (1.10)$$

Se debe a que el producto matricial definido en 1.8 de dos matrices semidefinidas positivas ($C - \sum_{i=1}^{m} A_i y_i$ y X) cumple ser no negativo. Tenemos, por tanto, gracias a 1.10 que el valor objetivo del problema primal, evaluado en cualquier X factible, es siempre mayor o igual que el valor objetivo del problema dual evaluado en cualquier y factible. Esto se conoce como *dualidad débil*. Hemos conseguido por tanto una cota superior del máximo del problema dual y una cota inferior del mínimo del problema primal.

Se cumple la propiedad de que $\langle X, Y\rangle = 0$ si y solamente si $XY = YX = 0$. Efectivamente, supongamos que $AB = 0$. Entonces, $\langle A, B\rangle = \mathrm{Tr}(AB) = 0$. Contrariamente supongamos que $\langle A, B\rangle = 0$, podemos expresar $B = \sum_{i=1}^{k} R_i$, donde R_i son matrices semidefinidas positivas de rango 1. Entonces se cumple que $\langle A, B\rangle = \sum_{i=1}^{k} \langle A, R_i\rangle = 0$. Dado que el cono $\mathcal{S}_+^n$ es autodual, sabemos que $\langle A, R_i\rangle \geq 0$ y por tanto $\langle A, R_i\rangle = 0$ para todo i. Las matrices R_i tienen rango 1 por lo que existe algún vector $v_i \in \mathbb{R}^n$ tal que $R_i = v_i v_i^T$. Por tanto reescribimos el producto $\langle A, R_i\rangle = v_i^T A v_i = 0$, y dado que $A \succeq 0$ se tiene que v_i pertenece al núcleo de A. Consecuentemente $AR_i = Av_i v_i^T = 0$ para todo i y $AB = 0$.

Veamos un resultado que caracteriza la optimalidad de los problemas 1.8 y 1.9.

Lema 1.3.1 ***Condiciones de optimalidad para SDP*** *Sean X e y soluciones factibles de problema primal y dual respectivamente. Si cumplen la siguiente condición*

$$\left(C - \sum_{i=1}^{m} A_i y_i\right) X = 0 \tag{1.11}$$

entonces alcanzan el mismo valor objetivo, $\langle C, X\rangle = b^T y$ y cumplen ser las soluciones óptimas al problema primal y dual del SDP.

En general, la implicación contraria no se cumple. Se requieren más hipótesis.

Si consideramos el ejemplo 1.3.2, el dual asociado es el siguiente

$$\begin{array}{ll} \max & y \\ \text{s.a.} & \begin{bmatrix} 2-y & 1 \\ 1 & -y \end{bmatrix} \succeq 0 \end{array} \tag{1.12}$$

La solución óptima es $y^\star = 1 - \sqrt{2}$, con valor de coste $1 - \sqrt{2}$. En este caso los costes de las soluciones óptimas coinciden. Se cumple además que:

$$\left(C - \sum_{i=1}^{m} A_i y_i^\star\right) X^\star = \begin{bmatrix} 1+\sqrt{2} & 1 \\ 1 & \sqrt{2}-1 \end{bmatrix} \begin{bmatrix} \frac{2-\sqrt{2}}{4} & -\frac{1}{2\sqrt{2}} \\ -\frac{1}{2\sqrt{2}} & \frac{2+\sqrt{2}}{4} \end{bmatrix} = 0 \tag{1.13}$$

En contraste con la programación lineal, la dualidad fuerte no se cumple en general, existen problemas en los que tanto la formulación primal como la dual son factibles, pero sus valores óptimos difieren, existe un *gap* entre ellos.

Ejemplo 1.3.3 *Consideremos el siguiente problema primal con su dual asociado*

$$\begin{array}{ll} \min & x_{11} \\ s.a. & 2x_{12} = 1 \\ & \begin{bmatrix} x_{11} & x_{12} \\ x_{12} & x_{22} \end{bmatrix} \succeq 0 \end{array} \qquad \begin{array}{ll} \max & y \\ s.a. & \begin{bmatrix} 0 & y \\ y & 0 \end{bmatrix} \preceq \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \end{array}$$

En el caso del dual, la única solución factible es $y = 0$, por lo que el valor óptimo es 0. En el problema primal no puede ocurrir $x_{11} = 0$ ya que en ese caso la solución no sería factible. Por tanto, existe una diferencia entres los valores óptimos de ambas formulaciones.

Existen condiciones bajo las cuales se cumple la dualidad fuerte para programación semidefinida. Un ejemplo de ellas son las *condiciones de Slater* por las cuales tanto el problema primal como el dual deben ser estrictamente factibles, es decir, para el primal se debe cumplir $X \succ 0$ y para el dual $C - \sum_i A_i y_i \succ 0$. Dichas matrices deben ser definidas positivas, no semidefinidas positivas. Por ejemplo, en el caso 1.3.3 existe una diferencia entre valores óptimos necesariamente, pero el dual no es estrictamente factible ya que $\begin{bmatrix} 0 & y \\ y & 0 \end{bmatrix} \prec \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \Leftrightarrow y^2 < 0$ lo cual es imposible. Se violan las condiciones de Slater.

1.4. Programación cónica

Podemos intuir ciertas similitudes entre las formulaciones de problemas de optimización lineales 1.2 , cónicos de segundo orden **??** y semidefinidos 1.8 y efectivamente existe una formulación que engloba estos tres casos. Se trata de la *programación cónica*. Antes de formular un problema general de este tipo vamos a definir ciertos conceptos necesarios.

En los tres casos anteriores estamos tratando de hallar la solución en conjuntos que cumplen ser la intersección de un cono propio y un subespacio afín. En el caso lineal se trata del subespacio afín $Ax = b$ y el cono $x \geq 0$. Pero,

¿qué es un *cono propio*? Veamos una descripción formal.

Sean S y T dos espacios vectoriales reales y una función lineal $\mathcal{A} : S \to T$. Sean S^* y T^* los espacios duales asociados a S y T, es decir, los espacios vectoriales de funciones lineales reales. Definimos el producto entre un espacio y su dual como $\langle \cdot, \cdot \rangle : S^* \times S :\to \mathbb{R}$ determinado por $\langle f, x \rangle = f(x)$. El adjunto de la función $\mathcal{A}$ sería $\mathcal{A}^* : T^* \to S^*$ definido por

$$\langle \mathcal{A}^* y, x \rangle_S = \langle y, \mathcal{A}x \rangle_T \qquad \forall x \in S, y \in T^* \tag{1.14}$$

Decimos que un cono $\mathcal{K} \subset S$ tiene punta si $\mathcal{K} \cap (-\mathcal{K}) = \{0\}$ y es sólido si tiene dimensión máxima, es decir, $\dim \mathcal{K} = \dim S$.

Definición 1.4.1 *Un cono propio es un cono convexo, cerrado, con punta y sólido.*

Programación cónica estándar. Dado un mapa lineal $\mathcal{A} : S \to T$ y un cono propio $\mathcal{K} \subset S$, definimos los problemas de optimización primal y dual cónicos como

$$\begin{array}{lll} \min & \langle c, x \rangle_S & \\ \text{s.a.} & \mathcal{A}x = b & (1.15) \\ & x \in \mathcal{K} & \end{array} \qquad \begin{array}{lll} \max & \langle y, b \rangle_T & \\ \text{s.a.} & c - \mathcal{A}^* y \in \mathcal{K}^* & (1.16) \end{array}$$

donde $b \in T, c \in S^*$. Se prueba la dualidad fácilmente, procediendo de manera análoga al caso lineal y al semidefinido.

Los caso particulares que hemos tratado son el de programación lineal donde el cono $\mathcal{K}$ sería $\mathbb{R}^n_+$, programación cónica de segundo orden $\mathcal{K} = \mathcal{L}^n_+$ y programación semidefinida $\mathcal{K} = \mathcal{S}^n_+$.

1.5. Coste computacional

Queremos realizar una comparativa entre los problemas de tipo LP, SOCP y SDP y el coste de resolverlos usando métodos de punto interior. Se cumple que un problema LP se puede expresar como uno SOCP y, a su vez, uno SOCP como uno SDP, por tanto, tenemos las inclusiones siguientes:

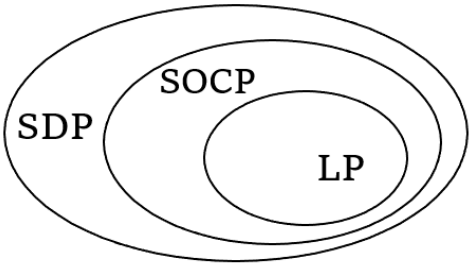

Figura 1.5

Se espera que si comparamos por ejemplo la complejidad de resolución de un problema LP con uno SOCP se tenga que para el peor de los casos LP, y el más simple de los SOCP, el LP sea igual o más sencillo de resolver (lo mismo para la comparativa LP con SDP y SOCP con SDP). Damos los detalles de la complejidad general de cada problema, para una profundización en el tema nos referimos a [BTN01, cap.6].

$$\text{Problema LP} = \begin{cases} \text{Formulamos el problema:} \\ min\{c^T x \ : \ a_i^T x \leq b_i, \ \ ||x||_\infty \leq R\} \\ \\ Data : [n; m; c; a_i; b_i; R] \\ Size = dim(Data) = (m+1)(n+1)+2 \\ \\ \text{Su complejidad viene dada por la fórmula:} \\ \mathcal{O}(1)(m+n)^{3/2} n^2 Digits(\epsilon) \end{cases}$$

$$\text{Problema SOCP} = \begin{cases} min\{c^T x \ : \ ||A_i x + b_i|| \leq c_i^T x + d_i, \ \ ||x||_2 \leq R\} \ \ [b_i \in \mathbb{R}^{k_i}] \\ Data : [n; m; k_i; c; A_i; b_i; c_i; d_i; R] \\ Size = dim(Data) = (m + \sum_1^m k_i)(n+1) + m + n + 3 \\ \mathcal{O}(1)(m+n)^{1/2} n(n^2 + m + \sum_1^m k_i^2) Digits(\epsilon) \end{cases}$$

$$\text{Problema SDP} = \begin{cases} min\{c^T x \ : \ A_0 + \sum_1^m A_j x_j \succeq 0, \ \ ||x||_2 \leq R\} \\ Data : [n; m; k_i; c; A_0; A_j; R] \\ Size = dim(Data) = (\sum_1^m k_i(k_i+1)/2)(n+1) + m + n + 3 \\ \mathcal{O}(1)(\sum_1^m k_i + n)^{1/2} n(n^2 + n\sum_1^m k_i^2 + \sum_1^m k_i^3) Digits(\epsilon) \end{cases}$$

A la hora de comparar las complejidades tomamos $m = n$ ya que nos da una cota superior de tamaño y $k_i = 1$ ya que es el mejor de los casos para los problemas de tipo SOCP y SDP (queremos ver que se resuelven con menor eficacia), las complejidades simplifcadas quedan $\mathcal{O}(1)n^{7/2}Digits(\epsilon)$, $\mathcal{O}(1)n^{3/2}(n^2 + 2n)Digits(\epsilon) = \mathcal{O}(1)n^{5/2}(n+2)Digits(\epsilon)$ y $\mathcal{O}(1)n^{1/2}n(n^2 + n)Digits(\epsilon) = \mathcal{O}(1)n^{5/2}(n+1)Digits(\epsilon)$ respectivamente. Resultan todos del orden de $n^{7/2}$ sin contar el término $Digits(\epsilon)$ que también es el mismo si las $k_i = 1$ y se reducen los casos SDP y SOCP al LP.

Sin embargo , en el momento en que $k_i > 1$ las complejidades de los problemas del tipo SOCP y SDP se hacen estrictamente mayores que la del caso LP.

Para el caso de un problema del tipo LP, se pueden resolver problemas usando métodos de punto interior que tengan decenas o cientos de variables y datos de restricción. Si además, los datos están estructurados favorablemente para la resolución, el tamaño de las variables y restricciones puede ascender hasta varios millones y que siga siendo razonable resolverlo usando métodos de punto interior. Esto suele ocurrir para la mayoría de problemas de tipo LP, por la forma en la que están formulados. Por tanto, en la realidad, ese método es aceptablemente potente para problemas de tipo LP (hay que tener cuidado ya que si los datos no tienen la forma óptima la complejidad será mayor de la descrita).

En el caso de problemas del tipo SOCP y SDP el método de punto interior no puede llegar tan lejos. Para los del tipo SDP los códigos que funcionan son para problemas con 1000-2000 variables y para los del tipo SOCP no hay mucha información.

Los principales motivos de esta complejidad añadida para los del tipo SDP son:

1. Este tipo de problemas han sido históricamente menos estudiados que los LP, por lo que es razonable que no se hayan alcanzado tantos avances.

2. Los datos que estructuran estos problemas son simplemente más grandes y esto afecta a la complejidad de resolución directamente. Para usar el método de punto interior en SDP hay que ir, como poco, almacenando ciertas matrices cuyas entradas son los datos, y hay que emplear el método de Cholesky para resolver sistemas de ecuaciones.

Hemos visto como la complejidad aumenta irremediablemente si tenemos que resolver un SOCP o un SDP, en vez de un LP, usando métodos de punto interior. Un hipotético cambio de formulación que nos transformara un problema del tipo SDP o SOCP en uno LP sería, por tanto, muy útil a la hora de enfrentarnos a la resolución del problema.

Capítulo 2

Optimización sobre polinomios SOS

En esta sección tratamos la relación entre los polinomios no negativos y los que son suma de cuadrados. Además vemos cómo podemos resolver computacionalmente problemas de optimización sobre polinomios suma de cuadrados usando programación semidefinida.

Para ello vamos a ver definiciones y resultados que nos ayuden a caracterizar estos tipos de polinomios, primero en el caso de una variable y luego en el caso general de varias variables.

2.1. Polinomios no negativos

Consideremos un polinomio en n variables con coeficientes reales. Decimos que es *no negativo* si cumple

$$p(x_1, ..., x_n) \geq 0 \quad \text{para todo} \quad (x_1, ..., x_n) \in \mathbb{R}^n$$

Es razonable preguntarse si dado un polinomio $p(x)$, ¿cómo podremos determinar si es no negativo o no?, y en caso de contar con un método general de respuesta, ¿es eficiente? También cabría plantear si todos los polinomios no negativos siguen una estructura general ya que, en ese caso, sería más fácil identificarlos.

2.1.1. Polinomios en una variable

Trataremos estas cuestiones en el caso general, pero para empezar consideramos el caso de una variable ($n = 1$). Sea el polinomio $p(x)$

$$p(x) = p_d x^d + p_{d-1} x^{d-1} + ... + p_1 x + p_0$$

Cumpliendo $p_i \in \mathbb{R} \ \ \forall i = 1, ..., d$. Suponemos que el coeficiente líder no es nulo, por lo que siempre podremos normalizar el polinomio ($p_d = 1$), en este caso decimos que el polinomio es *mónico*.

Definición 2.1.1 *Las raíces de un polinomio son los valores de x para los cuales el polinomio se anula.*

El teorema fundamental del álgebra nos asegura que podremos expresar nuestro polinomio en una variable como

$$p(x) = \prod_{i=1}^{d} (x - x_i)$$

Siendo x_i las raíces de nuestro polinomio. Dichas raíces serán complejas en general, y también pueden contar con multiplicidades.

Como primer criterio debemos imponer que el grado de $p(x)$ sea par para que sea no negativo. Es claro ya que si el grado es impar, en uno de los límites $x \to \infty$ o $x \to -\infty$ nuestro polinomio se hará negativo.

Para casos más concretos se puede encontrar una caracterización de polinomio no negativo.

Ejemplo 2.1.1 *Tomando un polinomio cuadrático, $p(x) = x^2 + p_1 x + p_0$, ¿qué condiciones debe cumplir para que sea no negativo?*

Dado que estamos ante una función convexa que alcanza su mínimo, bastará imponer que dicho mínimo sea mayor o igual que 0. Si resolvemos de manera general, el problema de minimización, se obtiene que $x_{opt} = -p_1/2$ con valor óptimo $p_0 - p_1^2/4$. Podremos entonces caracterizar los polinomios cuadráticos no negativos de la siguiente manera

$$\{p(x) : p(x) \geq 0 \ \ \forall x \in \mathbb{R}\} = \{p(x) : 4p_0 - p_1^2 \geq 0\}$$

Este procedimiento nos ha servido para el caso de un polinomio de grado 2, pero en el caso general, para una sola variable todavía, este camino puede complicarse. Otra vía que explorar es el comportamiento de las raíces del polinomio. Si $p(x)$ cumple tener el coeficiente líder positivo y que sea no negativo surgen dos posibilidades. Sólo puede ocurrir que, o bien el polinomio no tiene raíces reales, o bien tiene raíces reales con multiplicidad par (si fuera impar el polinomio tomaría en algún momento valores negativos). Sin embargo, esta condición no nos proporciona una caracterización satisfactoria ya que las raíces de un polinomio no se pueden , en general, obtener como función elemental de los coeficientes de dicho polinomio.

Para ciertas configuraciones podemos encontrar un criterio aplicable al caso específico que estamos tratando, como el siguiente caso.

Ejemplo 2.1.2 *Veamos como caracterizar la positividad de un polinomio del tipo $x^4 + ax + b$. Lo derivamos e igualamos a cero para hallar los puntos en los que puede cambiar de monotonía.*

$$p'(x) = 4x^3 + a = 0 \Rightarrow x = \sqrt[3]{\frac{-a}{4}} = r_0$$

Por tanto, cambia de ser decreciente a creciente ya que la segunda derivada siempre es mayor que cero para cualquier $a \neq 0$. Esto nos indica que tendrá dos raíces reales si $p(r_0) < 0$, una raíz real múltiple si $p(r_0) = 0$ o ninguna raíz real si $p(r_0) > 0$. Vemos en la siguiente gráfica el número de raíces reales según los valores de a y b.

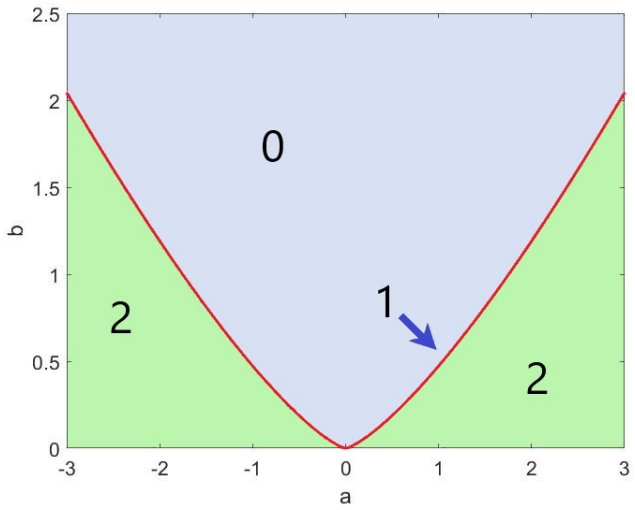

Figura 2.1.1

La línea roja indica los valores para los cuales el polinomio sólo tiene una raíz que es r_0, por lo que será no negativo. La zona azul indica los valores del par (a,b) para los cuales el polinomio es estrictamente positivo. En las zonas verdes el polinomio no cumple ninguna de estas características.

Queremos buscar, sin embargo, un método general, que podamos aplicar a un polinomio sin ningún tipo de imposición inicial. Existen varios métodos para saber si un polinomio de una variable es no negativo que no involucran la obtención de las raíces. Describimos ahora uno conocido como el método de *Hermite* o de *forma de traza*. Consideramos un polinomio en una variable mónico. Definimos su matriz de Hermite como la siguiente matriz $d \times d$ simétrica:

$$H_1(p) = \begin{pmatrix} s_0 & s_1 & \cdots & s_{d-1} \\ s_1 & s_2 & \cdots & s_d \\ \vdots & \vdots & \ddots & \vdots \\ s_{d-1} & s_d & \cdots & s_{2d-2} \end{pmatrix}, \qquad s_k = \sum_{j=1}^{d} x_j^k$$

Cumpliendo que x_j son las raíces del polinomio. Los valores s_k se pueden obtener directamente de las *identidades de Newton* operando con los coeficientes del polinomio, sin necesidad de hallar sus raíces.

$$s_0 = d \qquad s_k = \sum_{j=1}^{k} (-1)^{j-1} p_j s_{k-j} \tag{2.1}$$

A continuación presentamos unos resultados que nos permiten contar el número de raíces mediante el estudio de las propiedades de la matriz de Hermite.

Teorema 2.1.1 *El rango de la matriz de Hermite asociada a un polinomio $p(x)$ es igual al número de raíces complejas distintas. Además, el número de autovalores positivos menos el de autovalores negativos de la matriz, es decir, su signatura, nos indica la cantidad de raíces reales distintas del polinomio.*

Demostremos este resultado usando las siguientes definiciones e implicaciones.

Definición 2.1.2 *Sea A una matriz simétrica. La inercia de dicha matriz notada como $\mathcal{I}(A)$, es el triplete de enteros (n_-, n_0, n_+) que nos indica el número de autovalores positivos n_+, nulos n_0 y negativos n_-.*

Usamos el siguiente resultado *ley de inercia de Sylvester* [BPR06, p.120].

Teorema 2.1.2 *Sea V un $\mathbb{R}-$espacio vectorial de dimensión d y $Q : V \to \mathbb{K}$ una forma cuadrática.*

1. *Existen $L_1, ..., L_r$ formas lineales $L_i : V \to \mathbb{K}$ linealmente independientes y escalares no nulos $c_i \in \mathbb{K}$ tales que $Q = \sum_{i=1}^{r} c_i L_i^2$.*
2. *$rg(Q) = r$.*
3. *Para $\mathbb{K} = \mathbb{R}$, la terna formada por (n_-, n_0, n_+) está bien definida, donde n_+ (n_-) es el número de coeficientes $c_i > 0$ ($c_i < 0$) y $n_0 = d - r$.*

También precisamos de los siguientes lemas. La prueba del primero se encuentra en [Mey00, p.185].

Lema 2.1.1 *Dados $\alpha_1, ..., \alpha_m \in \mathbb{C}$ distintos dos a dos, construimos la matriz de Vardermonde, que cumple tener rango n si $n \leq m$.*

$$V_{m\times n} = \begin{pmatrix} 1 & \alpha_1 & \cdots & \alpha_1^{n-1} \\ 1 & \alpha_2 & \cdots & \alpha_2^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha_m & \cdots & \alpha_m^{n-1} \end{pmatrix}$$

Definimos la forma $L(x_i) : \mathbb{R}[x]_d \to \mathbb{C}$ caracterizada por $L(x_i)[f] = f(x_i)$ siendo $x_1, ..., x_r$ las distintas raíces de $p(x)$. $\mathbb{R}[x]_d$ es el espacio de polinomios de grados menor o igual que d.

Lema 2.1.2 *Las formas $L(x_1), ..., L(x_r)$ son independientes.*

PRUEBA: Si tomamos la base $1, x, x^2, ..., x^{d-1}$, la matriz de $L(x_i)$ es el vector columna $(1, x_i, x_i^2, ..., x_i^{d-1})^t$. Todas estas columnas son independientes, pues forman la traspuesta de una de una matriz de Vandermonde. □

Proposición 2.1.1 *El rango de $H_1(p)$ es igual a r.*

PRUEBA: Esto se debe a que para cada $f \in V$ se cumple que

$$H_1(p)[f] = \sum_{i=1}^{r} \mu(x_i) f(x_i)^2 = \sum_{i=1}^{r} \mu(x_i) L(x_i)[f]^2,$$

donde $\mu(x_i)$ es la multiplicidad de la raíz en p. Tenemos así una expresión de $H_1(p)$ como suma de cuadrados, todos ellos independientes, por lo que su rango tiene que ser r. □

Proposición 2.1.2 *La signatura de $H_1(p)$ es igual al número de raíces reales de p(x).*

PRUEBA: Sean

$$\{x \in \mathbb{R} | p(x) = 0\} = \{y_1, ..., y_s\}$$
$$\{x \in \mathbb{C} - \mathbb{R} | p(x) = 0\} = \{z_1, \bar{z}_1 ..., z_t, \bar{z}_t\}$$

La forma cuadrática $Q(p)$ verifica

$$Q(p)[f] = \sum_{i=1}^{s} \mu(x_i) L(y_i)[f]^2 + \sum_{j=1}^{t} \mu(z_j)(L(z_j)[f]^2 + L(\bar{z}_j)[f]^2)$$

Observemos que las formas $L(y_i), L(z_j)$ y $L(\bar{z}_j)$ son linealmente independientes. La matriz de cada una de ellas respecto de la base $1, x, x^2, ..., x^{d-1}$ es un vector columna de una matriz de Vandermonde, con todas las raíces distintas. Además, debemos expresar la forma cuadrática $Q(p)[f]$ como suma de cuadrados de formas lineales independientes sobre $\mathbb{R}$. Escribimos

$$\mu(z_j) = a(z_j)^2, z_j^i = s_i(z_j) + i t_i(z_j)$$

donde $s_i(z_j), t_i(z_j)$ son, respectivamente, la parte real e imaginaria de z_j^i. Definimos

$$L_1(z_j)[f] = a(z_j) \sum_{i=0}^{d-1} s_i(z_j) f_i$$
$$L_2(z_j)[f] = a(z_j) \sum_{i=0}^{d-1} t_i(z_j) f_i$$

que son formas lineales sobre $\mathbb{R}$. Se verifica que

$$\mu(z_j)(L(z_j)[f]^2 + L(\bar{z}_j)[f]^2) = 2L_1(z_j)[f]^2 - 2L_2(z_j)[f]^2.$$

De manera similar a lo visto anteriormente, las formas $L(y_i), L_1(z_j), L_2(z_j), i = 1, \ldots, s, j = 1, \ldots, t$, son linealmente independientes. Por tanto, tenemos una expresión de $Q(f)$ como suma de cuadrados, con $s+t$ términos con coeficiente positivo y t términos con coeficiente negativo. Por tanto, la signatura es $s + t - t = s$, que es el número de raíces reales.

□

Con esto hemos probado el teorema 2.1.1.

Se introduce a continuación un resultado que nos caracteriza la positividad de un polinomio a través de la inercia de su matriz de Hermite.

Teorema 2.1.3 *Sea $p(x)$ un polinomio mónico univariado de grado $2d$. Entonces las siguientes afirmaciones son equivalentes:*

1. *El polinomio $p(x)$ es estrictamente positivo.*
2. *El polinomio $p(x)$ no tiene raíces reales.*
3. *La inercia de la matriz de Hermite del polinomio es $\mathcal{I}(H_1(p)) = (k, 2d-k, k)$ para algún $1 \leq k \leq d$.*

PRUEBA: Es consecuencia del teorema 2.1.1. Es evidente que 1. se cumple si y sólo si se cumple 2. Gracias al teorema mencionado sabemos que el polinomio no tiene raíces reales si y solo si su signatura es nula. Por tanto el número de autovalores positivos y negativos debe ser el mismo, tal y como indica el punto 3. del teorema. □

Ejemplo 2.1.3 *Consideremos, una vez más, el polinomio $p(x) = x^2+p_1x+p_0$. Las sumas de potencias de sus raíces son $s_0 = 2, s_1 = -p_1$, y $s_2 = p_1^2 - 2p_0$. La matriz de Hermite queda*

$$H_1(p) = \begin{bmatrix} 2 & -p_1 \\ -p_1 & p_1^2 - 2p_0 \end{bmatrix}$$

Sea $\Delta = \det H_1(p) = p_1^2 - 4p_0$. La inercia de la matriz de Hermite es

$$\mathcal{I}\left(H_1(p)\right) = \begin{cases} (0,0,2) & \text{if } \Delta > 0 \\ (0,1,1) & \text{if } \Delta = 0 \\ (1,0,1) & \text{if } \Delta < 0 \end{cases}$$

y por tanto p es estrictamente positivo si y sólo si $p_1^2 - 4p_0 < 0$, como habíamos estimado 2.1.1.

2.1.2. Polinomios en varias variables

Tratemos ahora el caso más general que incluye varias variables. Sea $P_{n,2d}$ el conjunto de polinomios en n variables y grado $2d$ o menor no negativos.

$$P_{n,2d} = \{p \in \mathbb{R}[x]_{n,2d} : p(x) \geq 0 \quad \forall x \in \mathbb{R}^n\}$$

Podemos caracterizar un polinomio de este tipo por sus $N = \binom{n+d}{d}$ coeficientes, es decir, como un $N-$vector. En realidad, la condición $p(x) \geq 0$ no supone más que una restricción afín en dichos $N-$vectores para cada x fijo. Por lo tanto, $P_{n,2d}$ es un conjunto convexo en $\mathbb{R}[x]_{n,2d}$ isomorfo a $\mathbb{R}^N$. Más aún podemos asegurar lo siguiente.

Teorema 2.1.4 *$P_{n,2d}$ es un cono propio, es decir, un cono cerrado, convexo, sólido y con punta, en $\mathbb{R}[x]_{n,2d}$ isomorfo a $\mathbb{R}^N$.*

Prueba: Ya hemos comentado que es un conjunto convexo. Es fácil ver que es cerrado si lo vemos en $\mathbb{R}^N$. La restricción $p(x_0) \geq 0$ para un punto concreto es simplemente una condición afín sobre el vector de coeficientes de p que nos delimita un conjunto cerrado al ser una desigualdad no estricta. Por tanto, la condición $p(x) \geq 0 \; \forall x \in \mathbb{R}^n$ es la imposición de una restricción afín por cada punto de $\mathbb{R}^n$. Dado que la intersección arbitraria de conjuntos cerrados es cerrada $P_{n,2d}$ es cerrado.

Veamos que tiene punta. Si intersecamos $P_{n,2d}$ visto en $\mathbb{R}^N$ con su negativo nos da claramente el polinomio nulo ya que si para cualquier punto $x_0 \in \mathbb{R}^n$ un polinomio devuelve un valor positivo, el que tenga todos los coeficientes cambiados de signo devolverá un valor negativo, si $[p](x_0) \geq 0$ $[-p](x_0) \leq 0$, por lo que $-p$ no puede pertenecer a $P_{n,2d}$.

Es sólido ya que un conjunto de restricciones que sean desigualdades sobre $\mathbb{R}^N$ no reduce la dimensión a menos que resulte ser el vacío.

□

Excepto para casos especiales, no es fácil describir el conjunto $P_{n,2d}$ explícitamente de forma eficiente. Esto se debe a que la estructura de este conjunto

puede ser muy intrincada aunque se trate de un conjunto convexo. Por ello no merece la pena buscar un método general que ofrezca relaciones entre los coeficientes de los polinomios que determinen si son no negativos o no.

Observación. $P_{n,2d}$ es *semialgebraico* pero no es *semialgebraico básico*. Definamos estos conceptos.

Definición 2.1.3 *Sea un conjunto $S \subset \mathbb{R}^n$ definido como $S = \{x \in \mathbb{R}^n : f_i(x) \circ_i 0, i = 1, \ldots, t\}$, donde para cada i, $\circ_i$ es $\geq > = o \neq$ y $f_i(x) \in \mathbb{R}[x]$. Se denomina un conjunto semialgebraico básico. Un conjunto semialgebraico básico cerrado es de la forma $S = \{x \in \mathbb{R}^n : f_1(x) \geq 0, \ldots, f_t(x) \geq 0\}$.*

Definición 2.1.4 *Una unión finita de conjuntos semialgebraicos básicos en $\mathbb{R}^n$ se denomina semialgebraico y una unión finita de semialgebraicos cerrados básicos es un conjunto semialgebraico cerrado.*

Aunque a veces podamos describir $P_{n,2d}$ usando únicamente desigualdades o igualdades, en general, tenemos que recurrir a cuantificadores u operaciones lógicas. Veamos un caso que ejemplifica esto.

Ejemplo 2.1.4 *Sea $p(x) = x^4 + 2ax + b$ polinomio cuadrático de una variable. ¿Qué condiciones deben cumplir a y b para que sea no negativo? Debe cumplirse que el polinomio no tenga raíces reales para que sea estrictamente positivo. El discriminante de $p(x)$ (si tiene término líder unidad) es $\prod_{i<j}(r_i - r_j)^2$, siendo r_i y r_j las raíces del polinomio. Sabemos que se puede calcular el discriminante de un polinomio a partir de sus coeficientes. En nuestro caso*

$$Dis_x(p) = 256b(a^2 - b)^2$$

Para que el número de raíces reales cambie se debe cumplir que el discriminante se anule. Podemos por tanto, con ayuda de la expresión anterior, determinar dónde se anula el discriminante y hallar las zonas donde haya 0,2 ó 4 raíces reales del polinomio $p(x) = x^4 + 2ax + b$. Obtenemos que nuestro polinomio será positivo sólo en la clausura de la zona verde de la siguiente imagen (La línea azul indica dónde se anula el discriminante):

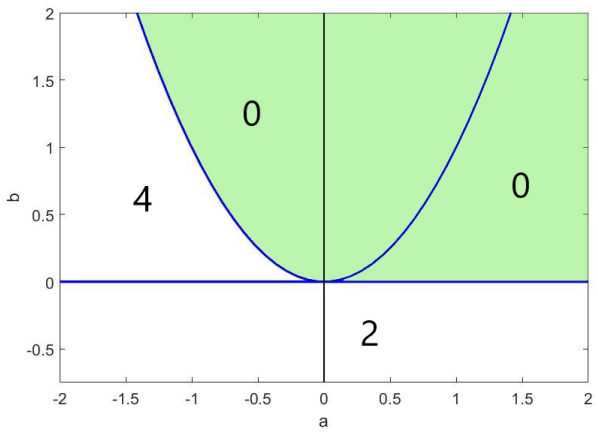

Figura 2.1.2

Como vemos en este ejemplo si $p(x)$, en una variable, se encuentra en la frontera de $P_{1,2d}$ entonces debe tener una raíz real de multiplicidad al menos 2. Ya que si no tuviera ninguna raíz real se encontraría estrictamente en el interior de $P_{1,2d}$, y si la raíz fuera simple $p(x)$ no pertenecería a $P_{1,2d}$. Por lo tanto en la frontera de $P_{1,2d}$ el discriminante debe ser nulo necesariamente.

En la figura vemos que además se anula en su interior. Esto supone una dificultad a la hora de trabajar con este tipo de conjuntos ya que no podemos hacernos una idea de lo "lejos"que estamos de la frontera del conjunto fijándonos en el discriminante. Esto sería una herramienta muy útil para optimizar numéricamente.

No podemos describir este conjunto únicamente con desigualdades o igualdades. En la figura, por ejemplo, se ve que necesitamos usar la unión para poder describir la clausura de la zona verde ($P_{1,4d}$ de nuestro polinomio).

Observación. En principio, podemos obtener condiciones explícitas que describan $P_{n,2d}$ usando técnicas de eliminación de cuantificadores [DBFC98]. Efectivamente si sustituimos las condición

$$\forall x \in \mathbb{R}^n \; p(x) \geq 0$$

por la descripción de un conjunto relativo a los coeficientes de p habremos conseguido un conjunto semialgebraico ya que hemos eliminado la variable cuantificada x. El problema de la no negatividad no es indecidible por tanto. Puede parecer que desde el punto de vista teórico esto simplifica el problema, pero en la realidad nos encontramos con serias dificultades prácticas, ya que la dependencia con el número de variables es doblemente exponencial. Así que sólo podemos proceder de esta manera para problemas de tamaño reducido.

Por tanto, vemos cómo la verificación de no negatividad para un polinomio puede complicarse, sobretodo si buscamos un método general. Es por ello que se han buscado otras soluciones a este problema. Una de ellas es la comprobación del carácter de suma de cuadrados que introducimos en la siguiente sección.

2.2. Suma de cuadrados

Si verificamos que un polinomio es suma de cuadrados, queda claro que será no negativo. Por lo tanto se podría abordar la cuestión de sustituir la verificación de no negatividad por la de suma de cuadrados. ¿Supone esto alguna ventaja?¿Tendrá algún inconveniente? A lo largo de esta sección caracterizamos los polinomios que cumple ser suma de cuadrados. Veremos cómo no siempre ser no negativo implica ser suma de cuadrados, por lo que puede que se pierda información al restringirnos a este conjunto, pero también veremos que existe un método general para comprobar si un polinomio es suma de cuadrados, y que se resuelve mediante programación semidefinida.

Definamos formalmente un polinomio que sea suma de cuadrados, en inglés se denominan "sums of squares"por lo que usamos las siglas *sos*.

Definición 2.2.1 *Un polinomio $p(x) \in \mathbb{R}[x]_{n,2d}$ es suma de cuadrados (sos) si existen polinomios $q_1(x), ..., q_m(x) \in \mathbb{R}[x]_{n,d}$ tales que se cumpla*

$$p(x) = \sum_{k=1}^{m} q_k(x)^2 \tag{2.2}$$

Denotamos al conjunto de polinomios *sos* en n variables y de grado menor o igual que $2d$ por $SOS_{n,2d}$. Es clara la contención $SOS_{n,2d} \subseteq P_{n,2d}$. Hay que tener en cuenta que si un polinomio admite una descomposición *sos*, ésta no tiene por qué ser única.

Ejemplo 2.2.1 *Si tenemos, por ejemplo, el siguiente polinomio* $p(x_1, x_2, x_3) = x_1^4 - (2x_2x_3 + 1)x_1^2 + (x_2^2x_3^2 + 2x_2x_3 + 2)$ *obtenemos que se puede formular como suma de cuadrados. Efectivamente si desarrollamos la siguiente expresión, que es una suma de cuadrados, obtendremos el polinomio inicial*

$$1 + x_1^2 + (1 - x_1^2 + x_2x_3)^2$$

Teorema 2.2.1 *El conjunto* $SOS_{n,2d}$ *de polinomios sos es un cono propio, es decir, un cono cerrado, convexo, sólido y con punta.*

PRUEBA: Es claro que el conjunto de polinomios *sos* es invariante frente a combinaciones convexas con coeficientes no negativos por lo que sería un cono convexo.

Tiene punta ya que hemos probado antes que $P_{n,2d} \cap -P_{n,2d} = \{0\}$ y como $SOS_{n,2d} \subset P_{n,2d} \Rightarrow -SOS_{n,2d} \subset -P_{n,2d} \Rightarrow SOS_{n,2d} \cap -SOS_{n,2d} = \{0\}$

Las dos propiedades que faltan se prueban en los comentarios sobre el Lema 2.2.1.

□

Ya hemos establecido una contención entre $SOS_{n,2d}$ y $P_{n,2d}$. Pero la cuestión que planteamos es si podemos concretar más la relación entre estos dos conjuntos. Nos preguntamos, por tanto, ¿cuándo es equivalente que un polinomio sea no negativo y sea suma de cuadrados?

Ya sabemos que ser suma de cuadrados implica ser no negativo, pero al contrario no tiene por qué ocurrir. Hace más de un siglo, David Hilbert demostró que la igualdad $SOS_{n,2d} = P_{n,2d}$ sólo ocurre en estas tres situaciones [Hil88]:

- Polinomios univariados ($n = 1$).

- Polinomios cuadráticos ($2d = 2$).
- Polinomios bivariados de grados 4 ($n = 2, 2d = 4$).

En el resto de casos, existe siempre un polinomio perteneciente a $P_{n,2d}$ pero que no sea suma de cuadrados. Un ejemplo de ello es el famoso caso expuesto por Motzkin. Si tomamos el polinomio $M(x, y) = x^4y^2 + x^2y^4 - 3x^2y^2 + 1$ vemos que es positivo $\forall x, y \in \mathbb{R}$. Efectivamente si hacemos uso de la desigualdad existente entre la media geométrica y la media aritmética aplicado a $1, x^2y^4, x^4y^2$, se comprueba que:

$$\frac{x^4y^2 + x^2y^4 + 1}{3} \geq ((x^2y^4)(x^4y^2))^{1/3} = x^2y^2$$

Lo que implica directamente la positividad de nuestro polinomio inicial $M(x, y)$ en todo el plano afín. Aún así demostraremos ahora que no se puede expresar como suma de cuadrados por reducción a lo absurdo. Supongamos que se puede expresar como suma de cuadrados:

$$x^4y^2 + x^2y^4 - 3x^2y^2 + 1 = f_1(x, y)^2 + ... + f_n(x, y)^2$$

Los f_i deben tener grado ≤ 3 ya que si no habría términos no nulos de grado ≥ 6 en el polinomio resultante, y el grado de $M(x, y)$ es 6. Si imponemos en el polinomio $x = 0$ tenemos que $1 = f_1(0, y)^2 + ... + f_n(0, y)^2$, por lo que deducimos que $f_i(x, y) = xg_i(x, y) + c_i$ con c_i una constante. Además se cumple $c_1^2 + ... + c_n^2 = 1$.

Por razones de simetría podemos operar igualmente con y y obtenemos que:

$$g_i(x, y) = yh_i(x, y) \Rightarrow f_i(x, y) = yxh_i(x, y) + c_i$$

Se debe cumplir que las funciones $h_i(x, y)$ sean lineales. La expresión de nuestro polinomio será:

$$x^4y^2 + x^2y^4 - 3x^2y^2 + 1 = x^2y^2 \sum_i h_i^2 + 2xy \sum_i h_i c_i + \sum_i c_i^2$$

Como sabemos que $\sum_i c_i^2 = 1$ podemos simplificar eliminando 1 a cada lado de la ecuación. Como ahora x^2y^2 divide a la expresión de la izquierda, debe dividir a la de la derecha, y por tanto a $2xy\sum_i h_i c_i$. Esto nos indica que xy debe dividir a $\sum_i h_i c_i$, pero habíamos dicho que $h_i(x,y)$ era de grado 1 por lo que $\sum_i h_i c_i = 0$. Tenemos:

$$x^4y^2 + x^2y^4 - 3x^2y^2 = x^2y^2\sum_i h_i^2$$

Dividiendo por x^2y^2 obtenemos $x^2 + y^2 - 3 = \sum_i h_i(x,y)^2$, si sustituimos en el punto $(0,0)$ vemos que no tiene sentido ya que una suma de cuadrados nos da un resultado negativo. Por lo tanto el polinomio $M(x,y)$ no se puede expresar como suma de cuadrados.

2.2.1. Formulación del caso general

Consideremos el polinomio $p(x_1, x_2, ..., x_n)$ de grado $2d$ y n variables (denotamos el *grado* del polinomio como el máximo de los grados de sus monomios). Podemos expresar, si tomamos $x = x_1x_2...x_n$, el polinomio de la siguiente manera

$$p(x) = \sum_\alpha p_\alpha x^\alpha \tag{2.3}$$

donde $x^\alpha = x_1^{\alpha_1}x_2^{\alpha_2}...x_n^{\alpha_n}$ son los posibles monomios descritos por las $n-$tuplas $\alpha \in \{(\alpha_1, \alpha_2, ..., \alpha_n) \in \mathbb{R}^n : \alpha_1 + ... + \alpha_n \leq 2d, \alpha_i \geq 0 \ \ \forall i = 1...n\}$. El *grado* de un monomio es la suma $\alpha_1 + ... + \alpha_n$. De este modo, el número de coeficientes de $p(x)$ será $N = \binom{n+2d}{2d}$ como ya habíamos comentado.

Otra manera de determinar la expresión de un polinomio de estas características, que nos resultará muy útil para saber si es *sos* o no, es la siguiente. Definimos $[x]_d = [1, x_1, ..., x_n, x_1^2, x_1x_2, ..., x_n^d]^T$ como el vector que tiene todos los $\binom{n+d}{d}$ monomios de grado d o menor en las variables $x_1, x_2, ..., x_n$. Entonces podemos formular

$$p(x) = [x]_d^T Q [x]_d \tag{2.4}$$

donde Q es una matriz simétrica $\binom{n+d}{d} \times \binom{n+d}{d}$. Comparando con la expresión 2.3 vemos que se debe cumplir $p_\alpha = \sum_{\beta+\gamma=\alpha} Q_{\beta\gamma}$.

Busquemos una condición para imponer que el polinomio sea suma de cuadrados. Si lo es debe ocurrir que $p(x) = \sum_{k=1}^{m} q_k^2(x)$ donde $q_k(x)$ son polinomios de grado d o menor en n variables. Podremos por tanto expresar

$$\begin{bmatrix} q_1(x) \\ q_2(x) \\ \vdots \\ q_m(x) \end{bmatrix} = V \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_n^d \end{bmatrix} = V[x]_d$$

donde V es una matriz de entradas reales y dimensión $m \times \binom{n+d}{d}$. Desarrollando la expresión del polinomio *sos* obtenemos

$$p(x) = \sum_{k=1}^{m} q_k^2(x) = (V[x]_d)^T V[x]_d = [x]_d^T V^T V[x]_d = [x]_d^T Q[x]_d \tag{2.5}$$

Hemos obtenido una condición para que $p(x)$ sea suma de cuadrados, que su matriz de representación Q sea de la forma $V^T V$. Veamos un resultado más formal.

Lema 2.2.1 *Sea $p(x)$ un polinomio en n variables y de grado $2d$. Entonces, será suma de cuadrados si y solamente si existe una matriz simétrica Q de dimensiones $\binom{n+d}{d} \times \binom{n+d}{d}$ que verifica*

$$p(x) = [x]_d^T Q[x]_d \qquad Q \succeq 0 \tag{2.6}$$

PRUEBA: Efectivamente, una de las implicaciones es clara. Si el polinomio es *sos* ya hemos visto que tiene que existir una matriz $Q = V^T V$ que represente a $p(x)$ como indica 2.6. Esta matriz será siempre semidefinida positiva

ya que $z^TQz = z^TV^TVz = (Vz)^TVz$ que es una suma de cuadrados siempre mayor o igual a 0.

Inversamente si para $p(x)$ existe Q que cumpla 2.6 sólo hace falta comprobar que podemos expresar $Q = V^TV$ para alguna matriz V de entradas reales y dimensión $m\times \binom{n+d}{d}$ y tendremos que $p(x)$ es suma de cuadrados por 2.5. Como Q es semidefinida positiva podemos realizar una factorización de Cholesky. □

Comprobar que se cumple la condición 2.6 (sistema de $\binom{n+2d}{2d}$ ecuaciones) es, en realidad, un problema de programación semidefinida. Observemos que la condición $p(x) = [x]_d^TQ[x]_d$ es en realidad un conjunto de condiciones afines en los coeficientes de Q ($p_\alpha = \sum_{\beta+\gamma=\alpha} Q_{\beta\gamma}$). Por tanto las posibles matrices que pueden representar nuestro polinomio son la intersección de un subespacio afín y el cono de la matrices semidefinidas positivas. Con esto probamos que $SOS_{n,2d}$ es cerrado y sólido.

Corolario 2.2.1 *Conocer la pertenencia a $SOS_{n,2d}$ se puede resolver mediante programación semidefinida.*

Ejemplo 2.2.2 *Queremos determinar si el siguiente polinomio se puede expresar como suma de cuadrados o no*

$$p(x,y) = 2x^4 + 5y^4 - x^2y^2 + 2x^3y + 2x + 2$$

El vector $[x]_d$ viene dado en este caso por $[1, x, y, x^2, xy, y^2]^T$. La expresión **??** *para nuestro polinomio será*

$$p(x,y) = \begin{bmatrix} 1 \\ x \\ y \\ x^2 \\ xy \\ y^2 \end{bmatrix}^T \begin{bmatrix} q_{00,00} & q_{00,10} & q_{00,01} & q_{00,20} & q_{00,11} & q_{00,02} \\ q_{00,10} & q_{10,10} & q_{10,01} & q_{10,20} & q_{10,11} & q_{10,02} \\ q_{00,01} & q_{10,01} & q_{01,01} & q_{01,20} & q_{01,11} & q_{01,02} \\ q_{00,20} & q_{10,20} & q_{01,20} & q_{20,20} & q_{20,11} & q_{20,02} \\ q_{00,11} & q_{10,11} & q_{01,11} & q_{20,11} & q_{11,11} & q_{11,02} \\ q_{00,02} & q_{10,02} & q_{01,02} & q_{20,02} & q_{11,02} & q_{02,02} \end{bmatrix} \begin{bmatrix} 1 \\ x \\ y \\ x^2 \\ xy \\ y^2 \end{bmatrix}$$

Desarrollando la expresión y comparándola con la inicial obtenemos $\binom{2+4}{4} = 15$ *ecuaciones lineales determinando los 15 coeficientes de* $p(x, y)$*. Por ejemplo, tres de ellas son*

$$\begin{array}{rc} x^4: & 2 = q_{20,20} \\ x^2y^2: & -1 = q_{00,22} + 2q_{01,21} + q_{11,11}, \\ y^2: & 2q_{00,02} + q_{01,01} \end{array}$$

Encontrar la matriz solución a este sistema de ecuaciones es un problema semidefinido como ya hemos comentado. Al solucionarlo obtenemos:

$$Q = \frac{1}{3}\begin{bmatrix} 6 & 3 & 0 & -2 & 0 & -2 \\ 3 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 \\ -2 & 0 & 0 & 6 & 3 & -4 \\ 0 & 0 & 0 & 3 & 5 & 0 \\ 2 & 0 & 0 & -4 & 0 & 15 \end{bmatrix}$$

que nos da una descomposición sos para nuestro polinomio:

$$\begin{aligned} p(x,y) = \frac{4}{3}y^2 + \frac{1349}{705}y^4 + \frac{1}{12}(4x+3)^2 + \frac{1}{15}(3x^2+5xy)^2 + \\ + \frac{1}{315}(-21x^2+20y^2+10)^2 + \frac{1}{59220}(328y^2-235)^2 \end{aligned}$$

Cualquier descomposición V^TV *de* Q *como la explicada anteriormente nos dará una forma sos de nuestro polinomio.*

Al contar con un método general para verificar si un polinomio es o no suma de cuadrados, podemos aprovecharlo en distintas aplicaciones. A continuación mostramos dos casos en los que los polinomios tienen una estructura concreta y eso nos permite ahondar más en la caracterización *sos*.

2.2.2. Casos especiales

Tratamos ahora casos concretos en los que los polinomios a tratar tienen una estructura determinada que se puede caracterizar algebraicamente. Explotamos esta cualidad para obtener métodos más concretos de resolución que son más eficientes.

Matrices suma de cuadrados

Podemos extender el concepto de polinomio no negativo y polinomio suma de cuadrados al caso de una matriz polinomial, es decir, una matriz con entradas en $\mathbb{R}[x_1, ..., x_n]$.

Definición 2.2.2 *Una matriz simétrica $P(x) \in \mathbb{R}[x]^{m\times m}$ es semidefinida positiva si $P(x) \succeq 0$ para todo $x \in \mathbb{R}^n$,es decir, que es semidefinida positiva en cada punto.*

Definición 2.2.3 *Una matriz polinomial simétrica $P(x) \in \mathbb{R}[x]^{m\times m}$, $x \in \mathbb{R}^n$ es una matriz suma de cuadrados si existe otra matriz polinomial $M(x) \in \mathbb{R}[x]^{s\times m}$ para algún $s \in \mathbb{N}$, tal que se cumple $P(x) = M^T(x)M(x)$.*

Para $m = 1$ es el caso escalar habitual que hemos estudiado. Si ocurre que P es una matriz escalar simplemente se tiene que cumplir que sea semidefinida positiva.

Ejemplo 2.2.3 *Veamos un ejemplo de matriz sos y su descomposición correpondiente*

$$P(x) = \begin{bmatrix} x^2 - 2x + 2 & x \\ x & x^2 \end{bmatrix} = \begin{bmatrix} 1 & x \\ x-1 & 0 \end{bmatrix}^T \begin{bmatrix} 1 & x \\ x-1 & 0 \end{bmatrix}$$

Damos ahora una caracterización de matriz suma de cuadrados basada en el hecho de que una matriz $m\times m$ simétrica se puede ver como la representación de una forma cuadrática en m variables con coeficientes en $\mathbb{R}[x]$. Por tanto, es como tener un polinomio $p(x, y)$ en las variables antiguas más m nuevas.

Lema 2.2.2 *Sea $P(x) \in \mathbb{R}^{m\times m}$ una matriz polinomial simétrica, $x \in \mathbb{R}^n$. Sea $p(x,y) = y^T P(x) y$ el polinomio asociado que se mueve en $m+n$ variables $[x;y]$. Entonces se cumple*

1. *La matriz $P(x)$ es semidefinida positiva si y solamente si $p(x,y)$ es no negativo.*
2. *La matriz $P(x)$ es suma de cuadrados si y solamente si $p(x,y)$ suma de cuadrados como polinomio en $\mathbb{R}[x;y]$.*

PRUEBA: Para la afirmación primera supongamos que $P(x)$ es semidefinida positiva para cualquier valor de x. Entonces $p(x,y) = y^T P(x) y \geq 0$ para cualquier valor de y o x. Si, al contrario, suponemos que $p(x,y)$ es no negativo se cumple que $y^T P(x) y \geq 0$ para cualquier y. Se deduce entonces que $P(x)$ es semidefinida positiva.

Probemos la segunda afirmación. Sea $P(x)$ una matriz suma de cuadrados. Se cumple, por tanto, que $P(x) = M^T(x)M(x)$. La expresión del polinomio $p(x,y)$ queda como

$$y^T P(x) y = y^T M^T(x) M(x) y = (M(x)y)^T M(x) y = \sum_{i=1}^{m} (M(x)y)_i^2$$

que es una suma de cuadrados. Contrariamente supongamos que el polinomio es suma de cuadrados. Al ser una expresión de tipo $y^T P(x) y$ es homogénea de grado 2 en y. Podemos escribir el polinomio como $\sum_i g_i(x,y)^2$ ya que es suma de cuadrados.

Debe cumplirse que cada polinomio $g_i(x,y)$ sea lineal en y, es decir, se puede expresar como $A_i(x)y$ siendo $A_i(x)$ una matriz polinómica y y el vector de coordenadas en y y $g_i(x,y)^2 = \sum_j (A_i(x)y)_j^2$. Queda la expresión

$$p(x,y) = \sum_i g_i(x,y)^2 = \sum_i (A_i(x)y)^T A_i(x) y = \sum_i y^T A_i(x)^T A_i(x) y =$$

$$= y^T \sum_i A_i(x)^T A_i(x) y = y^T M^T(x) M(x) y$$

Siendo $M(x) = \sum_i A_i(x)$. Queda probado que $P(x)$ es una matriz suma de cuadrados. □

Ilustremos esta caracterización con el ejemplo anterior. Podemos justificar que la matriz correspondiente a $P(x)$ es suma de cuadrados ya que la expresión $y^T P(x)y$ es suma de cuadrados.

$$y^T P(x)y = (y_1 + xy_2)^2 + (x-1)^2 y_1^2$$

Comprobamos, por tanto, que el lema visto es una potente herramienta para poder decidir si una matriz polinomial es *sos* o no, ya que se puede reducir a averiguar si un polinomio es suma de cuadrados o no. Puede parecer que no hemos realizado ningún avance dada la equivalencia de problemas, pero muchas cuestiones que se plantean en optimización pueden aparecer en forma de decidir si una matriz de polinomios es semidefinida positiva o no, por lo que esta caracterización sí es de utilidad.

La pregunta clara es: ¿Cúando una matriz de polinomios que sea semidefinida positiva es suma de cuadrados? Pues bien el siguiente resultado nos aclara el caso de una variable.

Teorema 2.2.2 *Sea $P(x) \in \mathbb{R}[x]^{m\times m}$ una matriz polinomial simétrica, donde la variable x es escalar. Entonces la matriz $P(x)$ es semidefinida positiva si y solo si es una matriz sos.*

PRUEBA: La implicación *sos* $\rightarrow$ *psd* es trivial. Probemos que si $P(x)$ es suma de cuadrados y univariada es suma de cuadrados. $P(x)$ es una matriz con entradas en $\mathbb{R}[x]_{2d}$ polinomios de grado $2d$. Podemos homogenizar estos polinomios para que sean de formas de grado $2d$ introduciendo una variable z. Tendremos ahora $P(x,z)$ matriz de las mismas dimensiones $m \times m$ cuyas entradas son formas en dos variables y grado $2d$. Si probamos que este tipo de matrices cumplen ser *sos* si y solo si son semidefinidas positivas habremos probado nuestro teorema ya que simplemente hay que particularizar z a 1.

Por el lema 2.2.2 basta probar que el polinomio $y^T P(x,z)y$ con y una variable $m-$dimensional es suma de cuadrados. Para probarlo usamos los siguientes lemas.

Lema 2.2.3 *Sean D_i y E_i vectores pertenecientes a $\mathbb{R}^{2d}$ con $1 \leqq i \leqq n$ cumpliendo*

$$D_i \cdot D_j = E_i \cdot E_j, \quad D_i \cdot E_j + D_j \cdot E_i = 0 \quad \text{para todo } i, j \tag{2.7}$$

entonces podemos hacer un cambio de variable ortonormal tal que

$$D_i = (s_{i1}, t_{i1}, \ldots, s_{id}, t_{id})$$
$$E_i = (-t_{i1}, s_{i1}, \ldots, -t_{id}, s_{id})$$

Se prueba por inducción en n. El caso $n = 0$ es trivial, así que tomamos $n \geq 1$. Por hipótesis D_1 y E_1 son ortogonales de la misma longitud, digamos s. Si $s = 0$, hemos acabado, suponiendo $s \neq 0$ podemos usar $\frac{1}{s}D_1$ y $\frac{1}{s}E_1$ como parte de la base ortonormal determinando

$$D_1 = (s, 0, 0, \ldots, 0), D_i = \left(s_{i_1}, t_{i1}, \overline{D}_i\right) \quad (i \geqq 2)$$
$$E_1 = (0, s, 0, \ldots, 0), E_i = \left(s'_{i1}, t'_{i1}, \overline{E}_i\right) \quad (i \geqq 2)$$

donde $\overline{D}_i, \overline{E}_i \in \mathbb{R}^{2(d-1)}$. Si sustituimos en la ecuaciones 2.7 con $j = 1$ se cumple que $t'_{i1} = s_{i1}$ y $s'_{i1} = -t_{i1}$ para todo $i \geq 2$. Operando en dichas ecuaciones comprobamos que $\overline{D}_i, \overline{E}_i (2 \leqq i \leqq n)$ guardan dichas relaciones, por lo que podemos proceder por inducción.

Lema 2.2.4 *Sean $a_{ij}(y, z)$ formas de grado m, con $a_{ij} = a_{ji}$ $(1 \leqq i, j \leqq n)$ y $a(y, z)$ una forma no negativa. Supongamos que existen vectores $A_i(y, z) = \left(a_i^{(1)}(y, z), \ldots, a_i^{(2d)}(y, z)\right)$ tales que $a(y, z)a_{ij}(y, z) = A_i(y, z) \cdot A_j(y, z)$ para todo i, j. Entonces existen vectores $B_i(y, z) = \left(b_i^{(1)}(y, z), \ldots, b_i^{(2d)}(y, z)\right)$ tales que $a_{ij}(y, z) = B_i(y, z) \cdot B_j(y, z)$ para todo i, j.*

La prueba de este lema es extensa y se encuentra en la publicación [CLR80, p. 21].

Con esto podemos probar el teorema por inducción en m la dimensión de la matriz. Si $m = 1$ tenemos que $p\,(x, z; y_1) = a_{11}(x, z)y_1^2$. Dado que $a_{11}(x, z)$ es no negativo, es una suma de cuadrados de formas y por tanto p también. Para $m > 1$ tomamos $a_{ij} = a_{ji}$, y escribimos p de la forma

$$p = a_{11}(x, z)y_1^2 + 2\left(\sum_{j\geqq 2} a_{1j}y_j\right) y_1 + \overline{p}\,(x, z; y_2, \ldots, y_m)$$

Consideramos $a_{11}(x, z) \neq 0$ ya que de otra manera $p = \overline{p}\,(x, z; y_2, \ldots, y_m)$ y acabaría la prueba. Se cumple que $a_{11}(x, z)$ es no negativo, por lo que el discriminante también lo es

$$D = a_{11}(x, z)\overline{p}\,(x, z; y_2, \ldots, y_m) - \left(\sum_{j\geqq 2} a_{1j}y_j\right)^2$$

Por la hipótesis de inducción, D es una suma de $2m - 2$ cuadrados de biformas en $\{x, z; y_2, \ldots, y_m\}$. La siguiente expresión

$$a_{11}(x, z) \cdot p = \left(a_{11}(x, z)y_1 + \sum_{j\geqq 2} a_{1j}y_j\right)^2 + D\,(x, z; y_2, \ldots, y_m)$$

que es una suma de $2m - 1$, en particular $2m$ cuadrados de biformas. Por tanto podemos expresar

$$a_{11}(y, z) \cdot p = \sum_{k=1}^{2n} \left(\sum_{i=1}^{n} a_i^{(k)}(y, z)x_i\right)^2 \tag{2.8}$$

Sea $A_i(y,z)$ el vector de formas $\left(a_i^{(1)}(y,z),\ldots,a_i^{(2n)}(y,z)\right)$. Si comparamos con 2.8 tenemos que $a_{11}\cdot a_{ij} = A_i\cdot A_j$. Por el lema 2.2.4 tenemos que $a_{ij} = B_i\cdot B_j$ para los vectores adecuados de la forma $B_i(y,z) = \left(b_i^{(1)}(y,z),\ldots,b_i^{(2n)}(y,z)\right)$. Por lo tanto hemos probado que $p = \sum_{k=1}^{2n}\left(\sum_{i=1}^{n} b_i^{(k)}(y,z)x_i\right)^2$ □

Para el caso de más variables esta propiedad no se cumple en general. Veamos un ejemplo de ello. Sea la matriz

$$C(x) = \begin{bmatrix} x_1^2+2x_2^2 & -x_1x_2 & -x_1x_3 \\ -x_1x_2 & x_2^2+2x_3^2 & -x_2x_3 \\ -x_1x_3 & -x_2x_3 & x_3^2+2x_1^2 \end{bmatrix}$$

Consideremos el polinomio $p(x,y) = y^TC(x)y$. Usamos el paquete *spot* [Meg10] y aplicamos al polinomio la función que nos indica si es suma de cuadrados o no, el resultado es negativo. Sin embargo, si calculamos los menores principales de la matriz obtenemos $x_1^2+2x_2^2$, $2(x_1^2x_3^2+x_2^4+2x_2^2x_2^2)$ y $4(x_1^2x_2^2x_3^2+x_1^2x_2^4+x_2^2x_3^4+x_2^4x_1^2)$ sucesivamente que son todos polinomios no negativos, por lo que la matriz $C(x)$ es semidefinida positiva para cualquier valor de x.

Politopos de Newton.

En el álgebra lineal numérica muchas veces se aprovecha lo vacías que sean las matrices con las que se trabaja. Se distinguen los algoritmos entre los que saben explotar el problema según su estructura y los que no.

Cuando hablamos de la estructura de una matriz y lo vacía que sea nos solemos referir a qué componentes son no nulos. En el álgebra computacional, sin embargo, existe un concepto de estructura de matriz mucho más concreto, que no sólo especifica qué coeficientes son no nulos sino que describe si hay algún patrón subyacente en dichos coeficientes.

Describimos la estructura de polinomios multivariados usando el concepto de politopos de Newton.

Definición 2.2.4 *Consideremos un polinomio del tipo* $p\left(x_1,\ldots,x_n\right)=\sum_\alpha c_\alpha x^\alpha$. *El politopo de Newton del polinomio* p, *que notamos como* $\mathcal{N}(p)$, *es el cierre convexo del conjunto de exponentes* α, *de los monomios si los entendemos como vectores en* $\mathbb{R}^n$.

Ejemplo 2.2.4 *Consideramos el polinomio* $p(x,y)=5-xy-x^2y^2+3y^2+x^4$. *Su politopo de Newton* $\mathcal{N}(p)$, *que vemos en la siguiente imagen es el cierre convexo de los puntos* $(0,0),(1,1),(2,2),(0,2),(4,0)$.

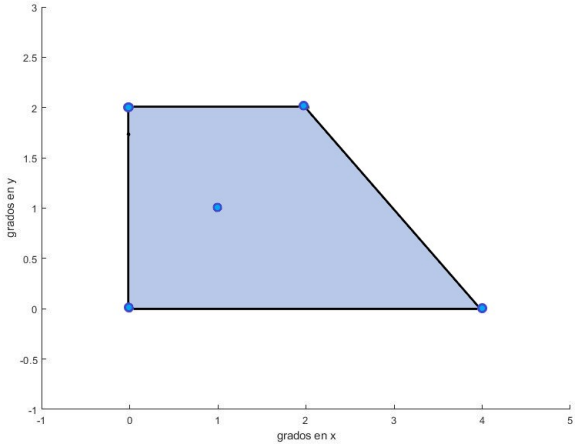

Figura 2.2.2

Los politopos de Newton nos resultan de gran utilidad a la hora de operar con polinomios ya que se cumple la siguiente propiedad, siendo la operación $+$ la suma de Minkowski para politopos, $A+B=\{\mathbf{a}+\mathbf{b}|\mathbf{a}\in A,\mathbf{b}\in B\}$.

$$\mathcal{N}(g\cdot h)=\mathcal{N}(g)+\mathcal{N}(h)$$

Claramente esto se cumple ya que los monomios que aparecen en el polinomio $g\cdot h$ son los productos de los monomios de h y g, es decir, los $\alpha_{hg}=\{\alpha_h+\alpha_g$ tal que α_h corresponde a un monomio de h y α_g a uno de $g\}$, es decir la suma de Minkowski.

Veamos un resultado de Reznick [Rez78] que nos permitirá realizar una reducción notable en la complejidad computacional de comprobar si un polinomio es suma de cuadrados.

Teorema 2.2.3 *Si* $p(x) = \sum_i q_t(x)^2$, *entonces* $\mathcal{N}(q_i) \subseteq \frac{1}{2}\mathcal{N}(p)$.

Introducimos una noción de estructura para un polinomio, en la que tenemos en cuenta el tamaño de su politopo de Newton. Gracias a este teorema, a la hora de estudiar el carácter *sos* de p, podemos restringirnos, sin pérdida de generalidad, a estudiar los monomios contenidos en el politopo de Newton de p reducido a la mitad. Esto hace que reduzcamos el tamaño del problema y su complejidad ya que la matriz Q a estudiar correspondiente al problema de optimización semidefinido se empequeñece notablemente.

Ejemplo 2.2.5 *Consideremos el siguiente polinomio para el queremos encontrar una descomposición sos.*

$$p(w,x,y,z) := \left(w^4+1\right)\left(x^4+1\right)\left(y^4+1\right)\left(z^4+1\right) + 2w + 3x + 4y + 5z$$

El polinomio p *tine gardo* $2d = 16$ *y cuatro variables* $n = 4$. *Podríamos intentar hallar una descomposición sos usando el método explicado anteriormente 2.2.1, pero para ello tendríamos que tratar con una matriz* Q *determinada por los coeficientes de todos los monomios en* (w,x,y,z) *de grado menor o igual que* 8, *que es un vector de tamaño* $\binom{n+d}{d} = 495$. *Este gran tamaño hace engorrosa la resolución.*

Si tomamos, sin embargo, su politopo de Newton $\mathcal{N}(p)$ *es fácil ver que sería un hipercubo con vértices opuestos* $(0,0,0,0)$ *y* $(4,4,4,4)$. *Por tanto, según el teorema 2.2.3, los polinomios* q_i *de la descomposición sos de* p *deben ser combinaciones lineales de monomios que se encuentren en* $\frac{1}{2}\mathcal{N}(p)$, *que será el hipercubo con vértices en* $(0,0,0,0)$ *y* $(2,2,2,2)$. *Este politopo resultante contiene* $3^4 = 81$ *posibles monomios, una cifra mucho menor que la inicial. Podemos por tanto hallar una descomposición sos resolviendo un problema semidefinido de mucho menor tamaño.*

2.3. Certificados de infactibilidad

Hasta ahora hemos dado caracterizaciones de no negatividad y suma de cuadrados para polinomios. Nos centramos ahora en estudiar la estructura de estas condiciones y además veremos como usar polinomios *sos* para comprobar que se cumplan ciertas propiedades en sistemas de ecuaciones.

2.3.1. Ideales y preórdenes

Normalmente determinamos un conjunto factible S, asociado a un problema, usando un número finito de ecuaciones polinomiales o desigualdades. Podemos, en principio, construir tantas restricciones nuevas como queramos que se sigan cumpliendo en S. Cabe preguntarse cuál sería el conjunto de todas las restricciones válidas para S, en particular, cómo generarlo algorítmicamente. Para ello, hacemos uso de dos conceptos algebraicos importantes: *ideales* y *preórdenes*.

Para el caso en que las restricciones que definan nuestro conjunto sean todas del tipo $f_{\mathrm{i}}(x) = 0$, generamos nuevas restricciones simplemente considerando las combinaciones lineales de las funciones $f_{\mathrm{i}}(x)$ con coeficientes polinomiales. El conjunto que se genera es un *ideal*. Demos una definición formal.

Definición 2.3.1 *Dado un conjunto de polinomios multivariados* $\{f_1, \ldots, f_m\}$*, el ideal generado por este conjunto es*

$$\langle f_1, \ldots, f_m \rangle := \{f : f = t_1 f_1 + \cdots + t_m f_m, \quad t_{\mathrm{r}} \in \mathbb{R}[x]\} \tag{2.9}$$

Análogamente, dado un conjunto de desigualdades $g_i(x) \geq 0$, se pueden generar nuevas restricciones válidas multiplicando cada $g_i(x)$ por polinomios *sos* o toando combinaciones cónicas de restricciones válidas. Formalicemos estas ideas.

Definición 2.3.2 *Dado un conjunto de polinomios multivariados* $\{g_1, \ldots, g_m\}$*, el modulo cuadrático generado por este conjunto es*

$$moduloq(g_1, \ldots, g_m) := \{g : g = s_0 + s_1 g_1 + \cdots + s_m g_m\} \tag{2.10}$$

donde $s_0, s_1, \ldots, s_m \in \mathbb{R}[x]$ *son sumas de cuadrados.*

Sin embargo, como ya hemos mencionado existe otra manera de generar restricciones válidas, es decir, el conjunto *moduloq* no contiene todas las restricciones válidas posibles. Podemos tomar los productos de las restricciones ya conocidas.

Definición 2.3.3 *Dado un conjunto de polinomios multivariados* $\{g_1, \ldots, g_m\}$*, el preorden generado por este conjunto es*

$$\text{preorden}\,(g_1, \ldots, g_m) := \left\{ g : g = s_0 + \sum_{\{i\}} s_i g_i + \sum_{\{i,j\}} s_{ij} g_i g_j + \sum_{\{i,j,k\}} s_{ijk} g_i g_j g_k + \cdots \right\}$$

donde cada término que suma es un producto de polinomios g_i *libre de cuadrados, con coeficientes* $s_\alpha \in \mathbb{R}[x]$ *sos. La suma es finita con un total de* 2^m *términos, uno por cada posible subconjunto de* $\{g_1, \ldots, g_m\}$*.*

Claramente, $\text{qmodule}(g_1, \ldots, g_m) \subseteq \text{preorden}(g_1, \ldots, g_m)$, por lo que en principio, el preorden define un conjunto mayor de restricciones válidas. Por construcción, los ideales, módulos cuadráticos y preórdenes sólo contienen polinomios que, en el caso de los ideales, al igualarlos a 0 darían restricciones válidas y, en el caso de módulos cuadráticos y preórdenes, al imponer que fueran no negativos también darían restricciones válidas.

Nos preguntamos si todas las restricciones válidas de cada tipo están contenidas en dichos conjuntos. En general no contienen todas las restricciones válidas posibles. Por ejemplo si definimos $S = \{x \in \mathbb{R} : x^2 = 0\}$, el ideal generado por el polinomio que da la restricción es $\langle x^2 \rangle$. Se cumple que $x = 0$ es una restricción válida para S ya que $S = \{0\}$ y $0 = 0$, pero $x \notin \langle x^2 \rangle$. Hemos encontrado una restricción válida que no está en el ideal. Para el caso de los módulos cuadráticos y preórdenes consideremos el conjunto $S = \{x \in \mathbb{R} : x^3 \geq 0\}$. El polinomio x cumple ser no negativo en S pero $x \notin \text{preorden}(x)$ y por tanto

$x \notin \text{moduloq}(x)$.

Aunque no tengamos garantía de conocer todas las restricciones válidas usando ideales, módulos cuadráticos o preórdenes podemos usarlos para detectar si nuestro conjunto es factible o no (es vacío o no).

Señalamos que, vistos como objetos geométricos, los ideales son conjuntos afines, y los módulos cuadráticos y preórdenes son cerrados bajo combinaciones convexas y multiplicaciones por un escalar positivo (en realidad son conos desde el punto de vista de la geometría convexa).

2.3.2. Certificados de infactibilidad

Si tenemos un conjunto definido por igualdades y/o desigualdades de polinomios y nos preguntamos si es factible o no, ¿podremos probar que no es factible?

Estamos buscando una garantía de que es un conjunto vacío, si fuera al contrario, probar que no es vacío, bastaría encontrar un elemento que cumpliera las restricciones.

Afortunadamente, para problemas de este tipo con estructura algebraica existen maneras de conseguir certificados de infactibilidad. Damos en esta sección resultados que caracterizan la no factibilidad de conjuntos en casos especiales y finalmente en el caso general de sistemas de polinomios sobre los números reales.

Vemos cómo podemos asegurar la infactibilidad para sistemas lineales de ecuaciones en los reales o en los complejos, e igual para sistemas de polinomios. Lo resumimos en la siguiente tabla.

Grado\Rango	Complejos	Reales
Lineal	Rango/núcleo	Lema de Farkas
Polinomial	Nullstellensatz	Positivstellensatz

Tabla 2.3.2

Ecuaciones lineales. Consideremos primeramente un sistema de ecuaciones lineales sobre un cuerpo cualquiera. Se conoce que si el sistema $Ax = b$ es incompatible, existe una combinación lineal de las ecuaciones tal que la parte de la izquierda se anula mientras que la de la derecha no. Se suele buscar este tipo de combinaciones usando eliminación Gaussiana. Formalicemos esto.

Teorema 2.3.1 *Consideremos el sistema $Ax = b$. Entonces*

$$\begin{array}{c} Ax = b \quad \textit{incompatible} \\ \Updownarrow \\ \exists \mu \ \textit{s.a.}\ A^T\mu = 0, b^T\mu = -1 \end{array}$$

PRUEBA: Si se cumple que $\exists \mu$ s.a. $A^T\mu = 0, b^T\mu = -1$ entonces

$$Ax = b \quad \Rightarrow \quad \mu^T Ax = \mu^T b \Rightarrow 0 = -1$$

que claramente es una contradicción. Contrariamente, haciendo uso del teorema Roché-Frobenius, si el sistema es incompatible el rango de la matriz A no es máximo y estrictamente menor que el rango de la matriz ampliada $A|b$ por lo ue existirá un vector λ s.a. $A^T\lambda = 0, b^T\lambda \neq 0$. Si tomamos el vector $\frac{-1}{b^T\lambda}\lambda$ habremos encontrado el vector μ requerido. □

Por tanto, si al tratar un sistema de ecuaciones somos capaces de encontrar un vector μ que cumpla las condiciones del teorema sabremos que dicho sistema es incompatible, hemos encontrado un certificado de infactibilidad. Además existe un método general para hallar dicho vector, la eliminación Gaussiana.

Sistemas polinomiales sobre $\mathbb{C}$. Para sistemas de polinomios sobre un cuerpo algebraicamente cerrado, podemos caracterizar la no factibilidad usando uno de los resultados centrales de geometría algebraica.

Teorema 2.3.2 ***(Nullstellensatz de Hilbert).*** *Sean $f_1(z), \ldots, f_m(z)$ polinomios en $z_1, \ldots, z_n$ variables complejas. Entonces,*

$$\begin{array}{c} f_i(z) = 0 \quad (i = 1, \ldots, m) \qquad \textit{infactible en } \mathbb{C}^n \\ \Updownarrow \\ -1 \in \langle f_1, \ldots, f_m \rangle \end{array}$$

Nos referimos a [Ful89, p. 10] para una prueba completa de este conocido resultado. Estudiemos la implicación más directa, si $-1 \in \langle f_1, \ldots, f_m \rangle$, quiere decir que existen $t_1(z), \ldots, t_m(z)$ tales que $t_1(z)f_1(z) + \cdots + t_m(z)f_m(z) = -1$.

Si sustituimos por z solución del sistema de polinomios el lado izquierdo de la ecuación se anula, por lo que llegamos a la contradicción $0 = -1$ $\Rightarrow$ problema no factible. Existe un método general para hallar los polinomios $t_1(z), \ldots, t_m(z)$ que definen la condición, consiste en hallar la base de *Gröbner* del ideal $\langle f_1, \ldots, f_m \rangle$. Éste método se ha estudiado en el curso de Álgebra combinatoria y computación del grado.

Señalamos que hasta ahora sólo se ha tratado el caso en el que el conjunto S estaba determinado por igualdades. Ahora introducimos desigualdades, por lo que tendremos que hacer distinción entre soluciones reales y complejas, ya que $\mathbb{R}$ es un cuerpo ordenado pero $\mathbb{C}$ no lo es.

Desigualdades lineales. Para sistemas de desigualdades lineales la dualidad fuerte estudiada en programación lineal nos proporciona certificados de infactibilidad eficientes. Es básicamente una interpretación algebraica del teorema de separación para poliedros. Lo formalizamos dando el *Lema de Farkas.*

Teorema 2.3.3 ***(Lema de Farkas).***

$$\begin{array}{c} \left\{ \begin{array}{l} Ax + b = 0, \\ Cx + d \geq 0 \end{array} \right. \qquad \textit{infactible} \\ \Updownarrow \\ \exists \lambda \geq 0,\ \mu \ \ s.a. \ \left\{ \begin{array}{l} A^T\mu + C^T\lambda = 0 \\ b^T\mu + d^T\lambda = -1 \end{array} \right. \end{array}$$

Se ha estudiado este resultado en el curso de programación matemática, podemos encontrar una prueba completa en [MSB77, p.70]. Análogamente a los casos anteriores, la dirección más sencilla de probar es trivial. Es equivalente al resultado de dualidad débil 1.1.1 comentado en el capítulo 1 , mientras que la otra implicación es equivalente a la dualidad fuerte 1.1.2. Podemos hallar el par (λ, μ) resolviendo el problema lineal, esto se puede realizar en tiempo polinomial usando el método del punto interior.

Podemos unificar estos resultados al caso en el que trabajemos con sistemas de ecuaciones polinomiales y desigualdades en los reales. Esto da una generalización del lema de Farkas y en el caso del Nullstellensatz, nos permite distinguir entre soluciones reales o complejas.

2.3.3. Positivstellensatz

Consideremos un sistema de ecuaciones y desigualdades polinomiales general. ¿Cómo podríamos probar que no tiene soluciones reales? Veremos a continuación un resultado que nos ofrece un certificado de infactibilidad, conocido como el *Positivstellensatz*. Es uno de los resultados fundamentales de geometría algebraica real. Se encuentra en [BCR98, Teorema 4.4.3].

Teorema 2.3.4 ***(Positivstellensatz)***.

$$\begin{cases} f_i(x) = 0 \quad (i = 1, \ldots, m), \\ g_i(x) \geq 0 \quad (i = 1, \ldots, p) \end{cases} \quad \textit{infactible en } \mathbb{R}^n$$

$$\Updownarrow$$

$$\exists F(x), G(x) \in \mathbb{R}[x] \ \textit{s.a.} \begin{cases} F(x) + G(x) = -1 \\ F(x) \in \langle f_1, \ldots, f_m \rangle \\ G(x) \in \ \textit{preorder} \ (g_1, \ldots, g_p) \end{cases}$$

El teorema nos da una caracterización de inexistencia de soluciones reales mediante una simple identidad algebraica. Para ellos usamos unos polinomios $F(x) \in \langle f_1, \ldots, f_m \rangle$ y $G(x) \in$ preorder $(g_1, \ldots, g_p)$ sin imponer condiciones sobre ellos.

Es razonable cuestionar la eficiencia con la que se pueden hallar estos polinomios. Hasta ahora hemos determinado que los certificados de infactibilidad vistos se pueden calcular usando álgebra lineal, programación lineal o bases de Gröbner para el caso de ecuaciones lineales, sistemas lineales y ecuaciones polinomiales respectivamente. Para el caso del Positivstellensatz tenemos que tener en cuenta que los ideales y los preórdenes son conos convexos en el espacio de los polinomios (la combinación lineal con coeficientes positivos o polinomios del conjunto es cerrada en ambos casos). Por tanto, las condiciones para que un certificado del tipo 2.3.4 existan son convexas, independientemente de que el problema original sea convexo o no. Si además imponemos, al buscar las funciones, $F(x)$ y $G(x)$ que tengan un grado máximo fijado las condiciones impuestas en 2.3.4 tienen la forma exacta de una condición del tipo *sos*, por lo que, recordando el corolario 2.2.1, se puede resolver usando programación semidefinida.

Un procedimiento sería ir buscando dichos polinomios de grado limitado D, aumentando D desde un mínimo. Obtendríamos una jerarquía de problemas semidefinidos que ir considerando.

Ejemplo 2.3.1 *Consideremos el sistema*

$$\begin{aligned} f_1 &:= x_1^2 + x_2^2 - 1 = 0 \\ g_1 &:= 3x_2 - x_1^3 - 2 \geq 0 \\ g_2 &:= x_1 - 8x_2^3 \geq 0 \end{aligned} \tag{2.11}$$

Probemos que no tiene soluciones reales $(x_1, x_2) \in \mathbb{R}^2$. *Usando el Positivstellensatz, el sistema es no factible si y solo si existen ciertos polinomios* $t_1, s_0, s_1, s_2, s_{12} \in \mathbb{R}[x_1, x_2]$ *tales que*

$$\underbrace{f_1 \cdot t_1}_{ideal\ \langle f_1 \rangle} + \underbrace{s_0 + s_1 \cdot g_1 + s_2 \cdot g_2 + s_{12} \cdot g_1 \cdot g_2}_{preorder\ (g_1, g_2)} = -1 \tag{2.12}$$

donde $s_0, s_1, s_2,$ *y* s_{12} *son sumas de cuadrados.*

Buscamos soluciones en las que todos los términos del lado izquierdo de la igualdad anterior tengan grado D fijado o menor. Para cada valor D que fijemos tenemos un problema de programación semidefinida que podemos programar para resolver. En este caso, para $D = 4$*, existen los polinomios*

$$\begin{gathered} t_1 = -3x_1^2 + x_1 - 3x_2^2 + 6x_2 - 2 \\ s_0 = \tfrac{5}{43}x_1^2 + \tfrac{387}{44}\left(x_1x_2 - \tfrac{52}{129}x_1\right)^2 + \tfrac{11}{5}\left(-x_1^2 - \tfrac{1}{22}x_1x_2 - \tfrac{5}{11}x_1 + x_2^2\right)^2 \\ + \tfrac{1}{20}\left(-x_1^2 + 2x_1x_2 + x_2^2 + 5x_2\right)^2 + \tfrac{3}{4}\left(2 - x_1^2 - x_2^2 - x_2\right)^2 \\ s_1 = 3, \quad s_2 = 1, \quad s_{12} = 0 \end{gathered}$$

Estos polinomios cumplen la condición 2.12 por lo que se ha probado la no factibilidad del sistema 2.11.

Capítulo 3

Optimización con DSOS y SDSOS

Vamos a definir unas clases de polinomios, que nos permitirán optimizar sobre polinomios no negativos más eficientemente. Esta clase de polinomios están contenidos exclusivamente en la categoría de polinomios no negativos, por lo que al trabajar sobre ellos estamos reduciendo las posibilidades del problema, es el precio a pagar por la ganancia en eficiencia.

3.1. Polinomios DSOS y SDSOS

Recordamos el conjunto de polinomios que son suma de cuadrados y denotamos por $SOS_{n,2d}$. Se cumple que $SOS_{n,2d} \subseteq PSD_{n,2d}$ exclusivamente ya que hemos visto que, por ejemplo, $M(x,y) = x^4y^2 + x^2y^4 - 3x^2y^2 + 1$ es positivo para todo par $(x,y) \in \mathbb{R}^2$ pero no se puede escribir como suma de cuadrados. Pasamos a definir los nuevos conjuntos de polinomios.

Definición 3.1.1 *Un polinomio p es suma de cuadrados diagonalmente dominantes (DSOS) si se puede escribir como*

$$p(x) = \sum_i \alpha_i m_i^2(x) + \sum_{ij} \beta_{ij}^+ (m_i(x) + m_j(x))^2 + \sum_{ij} \beta_{ij}^- (m_i(x) - m_j(x))^2$$

siendo $m_i(x), m_j(x)$ monomios y $\alpha_i, \beta_{ij}^+, \beta_{ij}^-$ escalares positivos. Llamaremos a estos polinomios $DSOS_{n,2d}$ (si están expresados en n variables y tienen grado $2d$).

Definición 3.1.2 *Un polinomio p es suma de cuadrados diagonalmente dominantes escalado (SDSOS) si se puede escribir como*

$$p(x) = \sum_i \alpha_i m_i^2(x) + \sum_{ij} (\widetilde{\beta}_{ij}^+ m_i(x) + \widehat{\beta}_{ij}^+ m_j(x))^2 + \sum_{ij} (\widetilde{\beta}_{ij}^- m_i(x) - \widehat{\beta}_{ij}^- m_j(x))^2$$

siendo $m_i(x), m_j(x)$ monomios y $\alpha_i, \widetilde{\beta}_{ij}^+, \widehat{\beta}_{ij}^+, \widetilde{\beta}_{ij}^-, \widehat{\beta}_{ij}^-$ escalares y α_i positivo. Llamaremos a estos polinomios $SDSOS_{n,2d}$ (si están expresados en n variables y tienen grado $2d$).

Estas descomposiciones no son necesariamente únicas para este tipo de polinomios. Observamos que el grado de los monomios no puede superar d, si nuestro polinomio fuera homogéneo de grado $2d$ dichos monomios tendrían que tener todos grado d exactamente. Con estas definiciones tenemos claramente las siguientes inclusiones:

$$DSOS_{n,2d} \subseteq SDSOS_{n,2d} \subseteq SOS_{n,2d} \subseteq PSD_{n,2d}$$

Estas inclusiones son exclusivas. Pongamos un ejemplo de ello, $q(x, y, z) = (x - y - z)^2$ es suma de cuadrados pero no es $DSOS_{n,2d}$. Es un polinomio al cuadrado por lo que obviamente es $SOS_{n,2d}$, probemos que no es $DSOS_{n,2d}$ por reducción a lo absurdo. Si desarrollamos la expresión obtenemos $q(x, y, z) = x^2 + y^2 + z^2 + 2zy - 2xy - 2xz$. Si, efectivamente $q \in DSOS_{n,2d}$ al ser de grado 2 homogéneo todos los monomios m_i de la expresión de la definición tendrán grado 1 necesariamente. El polinomio, por tanto, tendrá que poder expresarse de la forma

$$\begin{gathered} q(x, y, z) = \alpha_x x^2 + \alpha_y y^2 + \alpha_y y^2 + \beta_{xy}^+ (x + y)^2 + \beta_{yz}^+ (z + y)^2 \\ + \beta_{xz}^+ (x + z)^2 + \beta_{xy}^- (x - y)^2 + \beta_{xz}^- (x - z)^2 + \beta_{yz}^- (z - y)^2 \end{gathered}$$

Si comparamos coeficientes en las dos expresiones anteriores, nos surgen las siguientes restricciones.

$$\begin{cases} \alpha_x + \beta_{xy}^+ + \beta_{xz}^+ + \beta_{xy}^- + \beta_{xz}^- = 1 \\ \alpha_y + \beta_{xy}^+ + \beta_{yz}^+ + \beta_{xy}^- + \beta_{yz}^- = 1 \\ \alpha_z + \beta_{yz}^+ + \beta_{xz}^+ + \beta_{yz}^- + \beta_{xz}^- = 1 \\ \beta_{xy}^+ - \beta_{xy}^- = -1 \\ \beta_{xz}^+ - \beta_{xz}^- = -1 \\ \beta_{yz}^+ - \beta_{yz}^- = 1 \end{cases}$$

Si resolvemos el sistema imponiendo que todas las variables sean positivas, por ejemplo en MatLab con la función *linprog*, obtenemos que el problema no tiene solución.

Ejemplo 3.1.1 *Tomemos la forma general* $p(x_1, x_2) = x_1^4 + x_2^4 + ax_1^3x_2 + \left(1 - \frac{1}{2}a - \frac{1}{2}b\right)x_1^2x_2^2 + 2bx_1x_2^3$. *En la imagen se muestra el conjunto, según los valores de a y b, en el que el polinomio es SOS (azul claro), DSOS (verde) o SDSOS (amarillo).*

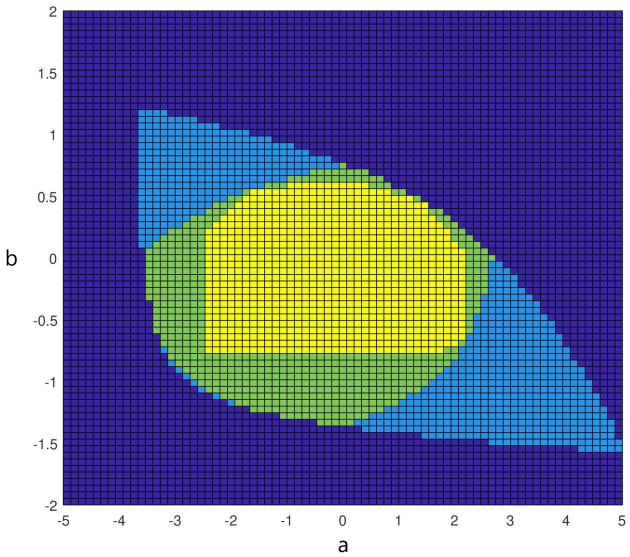

Figura 3.1.1

Un polinomio de grado $2d$ se puede expresar como $p(x) = z(x)^T Q z(x)$ siendo $z(x)$ el vector de monomios de grado $\leq d$. Vamos a definir dos tipos de matrices que nos caracterizarán la condición $DSOS$ y $SDSOS$.

Definición 3.1.3 *Una matriz simétrica $A = (a_{ij})$ es diagonalmente dominante (dd) si cumple*

$$a_{ii} \geq \sum_{j \neq i} |a_{ij}|$$

para todo i. Una matriz simétrica A se dice diagonalmente dominante escalada (sdd) si existe una matriz diagonal D, con entradas diagonales positivas, y que se cumpla que DAD sea diagonalmente dominante. Denotamos a este conjunto de matrices DD_n y SDD_n respectivamente. Nótese la inclusión obvia $DD_n \subseteq SDD_n$.

Además se tiene que las matrices DD_n son semidefinidas positivas. Esto se puede probar usando el *Teorema de Gershgorin* [Ger31] que nos dice que "dada una matriz $A = (a_{ij}) \in M_n(\mathbb{C})$ se definen los círculos $D_1, \ldots, D_n$ contenidos en $\mathbb{C}$ con centro en a_{ii} y radio $r_i = \sum_{j \neq i} |a_{ij}|$. Entonces los autovalores de la matriz A se encuentran en la unión de los n círculos". Por tanto, una matriz diagonalmente dominante (que tiene autovalores reales por ser simétrica real) cumplirá que todos sus autovalores son no negativos y será semidefinida positiva. DAD tiene autovalores del mismo signo que A por el teorema de Gerschgorin. Por tanto, las matrices SDD_n también serán semidefinidas positivas. Si denotamos a las matrices semidefinidas positivas de tamaño $n \times n$ por P_n se da:

$$DD_n \subseteq SDD_n \subseteq P_n$$

Por ejemplo , la matriz $\begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$ es semidefinida positiva pero no es diagonalmente dominante, sí es SDD_2 si escogemos $D = \begin{pmatrix} 3 & 0 \\ 0 & 1 \end{pmatrix}$ (DAD es diagonalmente dominante).

Relacionemos ahora estas clases de matrices con los polinomios $DSOS_{n,2d}$ y $SDSOS_{n,2d}$.

Teorema 3.1.1 *Un polinomio p de grado $2d$ es DSOS si y solamente si admite una representación del tipo $p(x) = z(x)^T Q z(x)$ con Q diagonalmente dominante (dd).*

Antes de probar este resultado enunciamos el siguiente lema.

Lema 3.1.1 ***(Barker y Carlson)*** *[BC75] Sea $\mathcal{V} = \{v_j\}_{j=1}^{2n^2}$ el conjunto de todos los vectores no nulos de $\mathbb{R}^n$ que tengan, a lo sumo, dos entradas no nulas, iguales a ± 1. Si denotamos como $\{V_i\}_{i=1}^{n^2}$ al conjunto de todas las matrices de rango 1 que se construyen de la forma $V_i = vv^T$ para algún $v \in \mathcal{V}$, una matriz simétrica M $n \times n$ será diagonalmente dominante si y solamente si se puede escribir como*

$$M = \sum_{i=1}^{n^2} \eta_i V_i$$

siendo $\eta_i \geq 0$.

PRUEBA: Supongamos que M es diagonalmente dominante. Sean los conjuntos $P = \{(i,j) | i < j, a_{ij} > 0\}$ y $N = \{(i,j) | i < j, a_{ij} < 0\}$ y u_i el vector que tiene un 1 en la posición i-ésima y 0 en el resto. Entonces podemos expresar M como suma de matrices de rango 1 del siguiente modo

$$M = \sum_{i=1}^{n} (a_{ii} - \sum_{i \neq j} |a_{ij}|) u_i u_i^T + \sum_{(i,j) \in P} a_{ij}(u_i + u_j)(u_i + u_j)^T + \sum_{(i,j) \in N} (-a_{ij})(u_i - u_j)(u_i - u_j)^T$$

Observamos que esta expresión es del tipo $\sum_{i=1}^{n^2} \eta_i V_i$ ya que cada matriz es de la forma $u_i u_i^T$ o bien $(u_i + u_j)(u_i + u_j)^T$, y en cada caso los vectores pertenecen a $\mathcal{V}$ ya que tienen a lo sumo dos elementos no nulos y son ± 1, además, como M es dd, todos los coeficientes escalares de los sumandos son positivos.

Contrariamente suponemos que podemos expresar $M = \sum_{i=1}^{n^2} \eta_i V_i$. La suma de matrices dd es dd ya que si A y B son diagonalmente dominantes ($a_{ii} \geq \sum_{j \neq i} |a_{ij}|$

y $b_{ii} \geq \sum_{j \neq i} |b_{ij}|$) entonces $A+B$ cumplirá $a_{ii}+b_{ii} \geq \sum_{j \neq i} |a_{ij}| + |b_{ij}| \geq \sum_{j \neq i} |a_{ij}+b_{ij}|$ por lo que $A+B$ es diagonalmente dominante. Si probamos que cada $\eta_i V_i$ es *dd*, habremos terminado la demostración.

En el caso de que $\eta_i V_i$ sólo tenga dos entradas no nulas, que estarán posicionadas en la diagonal (vector de sólo un elemento no nulo), es trivial comprobar que es *dd*. Si los vectores que definen la matriz V_i tienen dos entradas no nulas distinguimos dos casos, el primero, que estas entradas tengan el mismo signo, en cuyo caso V_i tiene cuatro elementos no nulos iguales a 1, dos en la diagonal y los otros dos formando una submatriz de cuatro 1. Al multiplicar por un escalar positivo η_i nos queda una matriz diagonalmente dominante ya que $\eta_i \geq |\eta_i|$.

En el caso de que los elementos del vector que define V_i tengan signo distinto, V_i tendrá dos 1 en la diagonal y dos -1 en las posiciones que forman una submatriz con los elementos positivos. Como se cumple $\eta_i \geq |-\eta_i|$, este tipo de matriz también será *dd*. El lema queda probado. □

Usamos el lema en la demostración del teorema 3.1.1 que damos a continuación.

PRUEBA: Supongamos ahora que p se puede expresar como $p(x) = z(x)^T Q z(x)$ con Q siendo diagonalmente dominante. El lema que acabamos de señalar nos asegura que:

$$p(x) = \sum_{i=1}^{n^2} (\eta_i z^T(x) v_i v_i^T z(x)) = \sum_{i=1}^{n^2} \eta_i (v_i^T z(x))^2,$$

donde $\eta_i \geq 0$ y v_i un vector de tamaño n con una ó dos entradas no nulas (y el resto nulas) iguales a ± 1. Separemos los sumandos de la expresión con el siguiente criterio, por un lado los sumandos asociados a un vector v_i de una sola entrada que generan un término del tipo $\alpha_i m_i^2(x)$ claramente. Por otro lado los términos que tengan v_i con dos entradas que corresponderán a términos $\beta_{ij}^+ (m_i(x) + m_j(x))^2$ ó $\beta_{ij}^- (m_i(x) - m_j(x))^2$ según tengan las componentes mismo signo (un par 1,1 ó -1,-1) o distinto signo (1,-1). Queda demostrado

entonces que el polinomio $p(x)$ es $DSOS_{n,2d}$.

Recíprocamente, si tengo un polinomio p que cumple ser $DSOS_{n,2d}$ admitirá una descomposición del tipo

$$\sum_i \alpha_i m_i^2(x) + \sum_{ij} \beta_{ij}^+ (m_i(x) + m_j(x))^2 + \sum_{ij} \beta_{ij}^- (m_i(x) - m_j(x))^2$$

Veamos que cada uno de estos términos se puede expresar como $p(x) = z(x)^T Q_k z(x)$ siendo Q_k *dd*.Ya hemos visto que la suma de matrices diagonalmente dominantes es diagonalmente dominante. Por tanto si cada Q_k definida antes es *dd* se cumplirá que nuestro polinomio se podrá expresar como $p(x) = z(x)^T Q z(x)$ siendo Q diagonalmente dominante.

Si la matriz Q_k corresponde a un término del tipo $\alpha_i m_i^2(x)$, será diagonal y por tanto diagonalmente dominante. Si corresponde a $\beta_{ij}^+ (m_i(x) + m_j(x))^2$ podremos expresar Q_k como una matriz de sólo cuatro entradas, dos de ellas diagonales de valor β_{ij}^+ y las dos restantes del mismo valor formando una submatriz con las diagonales por lo que también será *dd*. En el caso $\beta_{ij}^- (m_i(x) - m_j(x))^2$ ocurre lo mismo excepto que las entradas no diagonales no nulas serán negativas, pero se sigue cumpliendo la condición $\beta_{ij}^- \geq |-\beta_{ij}^-|$ por lo que estas matrices también serán *dd* y queda probado que p se puede expresar como $z(x)^T Q_k z(x)$ con Q diagonalmente dominante. □

Teorema 3.1.2 *Un polinomio p de grado* $2d$ *es SDSOS si y solamente si admite una representación del tipo* $p(x) = z(x)^T Q z(x)$ *con Q diagonalmente dominante escalada (sdd).*

PRUEBA: Supongamos primero que $p(x) = z(x)^T Q z(x)$ con Q siendo diagonalmente dominante escalada. Esto quiere decir que existe matriz diagonal D tal que DQD es *dd*. Si en la expresión del polinomio introducimos dos factores $I = D^{-1}D$ y DD^{-1} obtenemos

$$p(x) = z(x)^T D^{-1} DQD D^{-1} z(x) = (D^{-1} z(x))^T DQD (D^{-1} z(x))$$

Si seguimos el mismo argumento que en el apartado anterior, deducimos que es una forma $DSOS$ pero los monomios son entradas del vector $(D^{-1}z(x))$

no del vector $z(x)$, por lo que serán del tipo $d_{ii}m_i(x)$. Esto hace que la expresión del polinomio quede

$$\sum_i \alpha_i (d_{ii}m_i)^2(x) + \sum_{ij} \beta_{ij}^+ (d_{ii}m_i(x) + d_{jj}m_j(x))^2 + \sum_{ij} \beta_{ij}^- (d_{ii}m_i(x) - d_{jj}m_j(x))^2 = \sum_i \alpha_i m_i^2(x) + \sum_{ij} (\widetilde{\beta}_{ij}^+ m_i(x) + \widehat{\beta}_{ij}^+ m_j(x))^2 + \sum_{ij} (\widetilde{\beta}_{ij}^- m_i(x) - \widehat{\beta}_{ij}^- m_j(x))^2$$

por lo que nos queda la expresión de un polinomio $SDSOS$.

Al contrario, si un polinomio es $SDSOS$, admite una representación del tipo $\sum_i \alpha_i m_i^2(x) + \sum_{ij} (\widetilde{\beta}_{ij}^+ m_i(x) + \widehat{\beta}_{ij}^+ m_j(x))^2 + \sum_{ij} (\widetilde{\beta}_{ij}^- m_i(x) - \widehat{\beta}_{ij}^- m_j(x))^2$, hagamos corresponder cada uno de estos términos con uno del tipo $z(x)^T Q_k z(x)$. Si probamos que cada Q_k es diagonalmente dominante escalada tendremos que nuestro polinomio se puede expresar de manera $p(x) = z(x)^T Q z(x)$ con Q siendo *sdd* ya que la suma de matrices *sdd* es diagonalmente dominante escalada. Probar esta última implicación no es trivial pero la damos por sentado ya que en un lema posterior probaremos que el conjunto de matrices SDD_n es un cono.

Probemos por tanto que cada Q_k es diagonalmente dominante escalada. Si Q_k es la matriz asociada a un término del tipo $\alpha_i m_i^2(x)$, podemos tomar Q_k como matriz diagonal y, al igual que antes sería *dd*. Para los términos $(\widetilde{\beta}_{ij}^+ m_i(x) + \widehat{\beta}_{ij}^+ m_j(x))^2$ la matriz asociada tendrá todas sus entradas nulas menos 4 asociados a los términos $m_i(x)$ y $m_j(x)$ del vector $z(x)$, formando una submatriz 2×2 de la forma $\begin{pmatrix} \widetilde{\beta}_{ij}^+ \\ \widehat{\beta}_{ij}^+ \end{pmatrix} \begin{pmatrix} \widetilde{\beta}_{ij}^+ & \widehat{\beta}_{ij}^+ \end{pmatrix}$. Del mismo modo, si tenemos un término del tipo $(\widetilde{\beta}_{ij}^- m_i(x) - \widehat{\beta}_{ij}^- m_j(x))^2$ la matriz asociada será $\begin{pmatrix} \widetilde{\beta}_{ij}^- \\ -\widehat{\beta}_{ij}^- \end{pmatrix}$ $\begin{pmatrix} \widetilde{\beta}_{ij}^- & -\widehat{\beta}_{ij}^- \end{pmatrix}$. Probamos para los dos casos que son matrices *sdd*, si en general tenemos una matriz $\begin{pmatrix} a \\ b \end{pmatrix} \begin{pmatrix} a & b \end{pmatrix}$ se cumple que

$$\begin{pmatrix} \frac{signo(a)}{a} & 0 \\ 0 & \frac{signo(b)}{b} \end{pmatrix} \text{A} \begin{pmatrix} \frac{signo(a)}{a} & 0 \\ 0 & \frac{signo(b)}{b} \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

que es *dd* por lo que A es *sdd*. Si a ó b fueran nulos A sería *dd* directamente. □

Vemos ahora unas caracterizaciones de matriz *sdd* que necesitaremos para ver el tipo de programación con la que se pueden resolver problemas sobre polinomios $SDSOS_{n,2d}$.

Teorema 3.1.3 *Una matriz simétrica Q es sdd si y solamente si tiene un "factor de anchura" de 2 como mucho, es decir, una factorización de la matriz del tipo $Q = VV^T$, cumple que cada columna de V tiene, como mucho, dos entradas no nulas.*

Para una prueba del resultado nos referimos a los teoremas 8 y 9 de [BCPT05].

Lema 3.1.2 *Una matriz simétrica Q de dimensión $n \times n$ es sdd si y solamente si puede expresarse como*

$$Q = \sum_{i<j} M^{ij}$$

siendo cada M^{ij} una matriz con todas las entradas nulas excepto cuatro que forman una submatriz $\begin{bmatrix} (M^{ij})_{ii} & (M^{ij})_{ij} \\ (M^{ij})_{ji} & (M^{ij})_{jj} \end{bmatrix}$ simétrica y semidefinida positiva.

PRUEBA: Si Q es *sdd*, por el teorema anterior sabemos que se puede expresar como $Q = VV^T$, siendo V una matriz $n \times k$. Podemos desarrollar en términos $Q = \sum_{l=1} v_l v_l^T$ con v_l la columna l-ésima de V. Se cumple que cada v_l tiene como mucho dos entradas no nulas, por tanto cada término $v_l v_l^T$ tendrá, efectivamente, la forma M^{ij}.

Inversamente,supongamos que Q puede expresarse como indica el lema. Podemos escribir cada matriz M^{ij} como $w_{ij,1} w_{ij,1}^T + w_{ij,2} w_{ij,2}^T$ siendo $w_{ij,1}$ y

$w_{ij,2}$ vectores que tienen como mucho dos entradas no nulas. Si construimos una matriz V que tenga como columnas todos los vectores $w_{ij,1}$ y $w_{ij,2}$, se cumple que $Q = VV^T$.

Por tanto, el conjunto de matrices SDD_n es un cono, y se cumple que la suma de matrices diagonalmente dominantes escaladas es también diagonalmente dominante escalada. □

3.2. LP y SOCP en relación a polinomios DSOS y SDSOS

Teorema 3.2.1 *Para un d fijado, se puede optimizar sobre polinomios $DSOS_{n,2d}$ y $SDSOS_{n,2d}$ con programación lineal y cónica de segundo orden respectivamente de tamaño polinomial n en ambos casos.*

PRUEBA: En ambos casos se puede imponer la igualdad $p(x) = z(x)^T Q z(x)$ mediante una serie de igualdades lineales (una cantidad finita) impuestas sobre los coeficientes de p y los elementos de la matriz Q. Si queremos imponer que Q sea *dd* basta que se cumpla

$$Q_{ii} \geq \sum_{i \neq j} z_{ij} \ \forall i$$
$$-z_{ij} \leq Q_{ij} \leq z_{ij} \ \forall i, j \ i \neq j$$

que es, efectivamente, un programa lineal.

Para comprobar si la matriz Q es *sdd* podemos hacer uso del lema anterior. Dada una descomposición de Q en matrices M^{ij}, imponemos que cada una de ellas se semidefinida positiva. Para ello, como cada matriz sólo tiene 4 elementos no nulos, basta que se cumpla

$$(M^{ij})_{ii} + (M^{ij})_{jj} \geq 0$$

$$\left\| \begin{matrix} 2(M^{ij})_{ij} \\ (M^{ij})_{ii} - (M^{ij})_{jj} \end{matrix} \right\| \leq (M^{ij})_{ii} + (M^{ij})_{jj} \geq 0$$

que son restricciones compatibles con la programación cónica de segundo orden.

En ambos casos el tamaño polinomial es n ya que la matriz Q sobre la que imponemos las restricciones tiene $\binom{n+d}{d}$ que es polinomial en n para un d fijado. □

El hecho de poder sustituir problemas de programación semidefinida por problemas de programación lineal y cónicos de segundo orden supone un aumento de la eficencia si usamos métodos de punto interior para resolverlos 1.5.

3.3. Polinomios RDSOS y RSDSOS

Comentaremos en esta sección otra posibilidad de verificar si un polinomio es no negativo. Usamos los conjuntos $DSOS_{n,2d}$ y $SDSOS_{n,2d}$ de una nueva manera.

Definición 3.3.1 *Para un número entero $r \geq 0$, decimos que el polinomio $p := p(x_1, \ldots, x_n)$ is RDSOS (análogamente RSDSOS) si*

$$p(x) \cdot \left(\sum_{i=1}^{n} x_i^2 \right)^r$$

es dsos (sdsos respectivamente). Denotamos el conjunto de polinomios en n variables y grado 2d que son rdsos (rsdsos) como $RDSOS_{n,2d}$ ($RSDSOS_{n,2d}$).

Destacamos que para $r = 0$ recuperamos las definiciones originales *dsos*/*sdsos*. Mas aún se tiene que, como $(\sum_{i=1}^{n} x_i^2)^r$ es siempre no negativo

$$RDSOS_{n,2d} \subseteq RSDSOS_{n,2d} \subseteq PSD_{n,2d}$$

Es claro que optimizar sobre polinomios *rdsos* (*rsdsos* respectivamente) es un problema de programación lineal (cónico de segundo orden). La prueba del siguiente teorema es análoga a la del teorema 3.2.1 y, por tanto, omitida.

Teorema 3.3.1 *Para d y r fijados se cumple que optimizar sobre polinomios pertenecientes a $RDSOS_{n,2d}$ se puede realizar mediante programación lineal, y optimizar sobre polinomios en $RSDSOS_{n,2d}$ puede resolverse con programación cónica de segundo orden de tamaño polinomial n.*

Ejemplo 3.3.1 *Consideramos el mencionado polinomio de Motzkin, $M(x) = x_1^4x_2^2 + x_1^2x_2^4 - 3x_1^2x_2^2x_3^2 + x_3^6$, que se conoce históricamente como el primer ejemplo de polinomio no negativo que no es suma de cuadrados. La siguiente descomposición muestra que $M \in 2DSOS_{3,6}$, y por tanto M es no negativo:*

$$\begin{aligned} M(x) \cdot \left(\sum_{i=1}^{3} x_i^2\right)^2 &= \frac{1}{2}\left(x_3^5 - x_3^3x_2^2\right)^2 + \frac{1}{2}\left(x_2^4x_1 - x_3^4x_1\right)^2 + \frac{1}{2}\left(x_2^4x_1 - x_3^2x_2^2x_1\right)^2 \\ &+ \frac{1}{2}\left(x_3^5 - x_3^3x_1^2\right)^2 + \frac{1}{2}\left(x_3^3x_1^2 - x_3^3x_2^2\right)^2 + \frac{5}{2}\left(x_2^3x_1^2 - x_3^4x_2\right)^2 \\ &+ \frac{1}{2}\left(x_2^3x_1^2 - x_3^2x_2x_1^2\right)^2 + \frac{5}{2}\left(x_2^2x_1^3 - x_3^4x_1\right)^2 + \frac{1}{2}\left(x_2^2x_1^3 - x_3^2x_2^2x_1\right)^2 \\ &+ \frac{1}{2}\left(x_2x_1^4 - x_3^4x_2\right)^2 + \frac{1}{2}\left(x_2x_1^4 - x_3^2x_2x_1^2\right)^2 \end{aligned}$$

Esto constituye una prueba de que $2DSOS_{3,6} \nsubseteq SOS_{3,6}$.

Viendo que en algunos casos los conjuntos $RDSOS_{n,2d}$ contienen polinomios que $SOS_{n,2d}$ no, la imposición de que un polinomio pertenezca a $RDSOS_{n,2d}$ proporcionará un conjunto de soluciones factibles diferente, en ocasiones mayor. Por tanto, a parte de la ventaja en eficiencia, no siempre perderemos tamaño en la región factible.

Capítulo 4

Aplicaciones

Hemos demostrado que es más eficiente comprobar si un polinomio cumple ser $DSOS$ o $SDSOS$ que suma de cuadrados o no negativo. Vemos ahora cómo podemos aplicar esta cualidad en la resolución de problemas. Muchos de ellos ya se han tratado pero no desde esta nueva perspectiva. A la hora de comprobar el carácter $DSOS$, $SDSOS$ o SOS de los polinomios se ha usado el código de los autores [AM19] implementado en Matlab. Se usa el resolutor MOSEK en todos los casos, salvo en los que hallamos el mínimo global de un polinomio en donde usamos las herramientas de optimización de Matlab.

4.1. Mínimo de un polinomio en varias variables

Es común que se nos presente el problema de hallar el mínimo de cierta expresión, en concreto de un polinomio. Veamos un ejemplo de cómo podemos abordar y solucionar esta cuestión haciendo uso de las clases $DSOS$ y $SDSOS$ en contraposición al tipo SOS. Consideremos el caso especial de un polinomio $p(x)$ de grado $2d$ en la esfera unidad (el conjunto $\left\{x \in \mathbb{R}^n | x^T x = 1\right\}$). El problema de hallar el mínimo de esta función en ese conjunto ha sido ampliamente estudiado y se conoce que es de complejidad NP incluso para el caso $d = 2$.

Hallar su mínimo es equivalente a resolver el siguiente problema de optimización

$$\begin{aligned} \underset{\gamma}{\text{maximize}} \quad & \gamma \\ \text{s.a.} \quad & p(x) - \gamma \left(x^T x\right)^d \geq 0, \forall x \in \mathbb{R}^n \end{aligned} \tag{4.1}$$

Si sustituimos la condición de no negatividad por una restricción a los conjuntos $SOS/DSOS/SDSOS$ convertimos el problema en uno de tipo $SDP/LP/SOCP$.

Recordemos antes de ver algunos ejemplos que al sustituir la condición de no negatividad por otra más restrictiva estamos reduciendo la región factible, por lo que el mínimio resultante será mayor o igual que el real. Por ejemplo si consideramos el polinomio

$$p(x_1, x_2) = x_2^4 + bx_2^3x_1 + (1 - 0{,}5a + b)x_2^2x_1^2 + ax_1^3x_2 + x_1^4 \tag{4.2}$$

Dependiendo del valor de sus coeficientes vemos en la imagen si es $DSOS$ (y por tanto $DSOSS$ y SOS), $SDSOS$ (y por tanto SOS), SOS o ninguna de las anteriores.

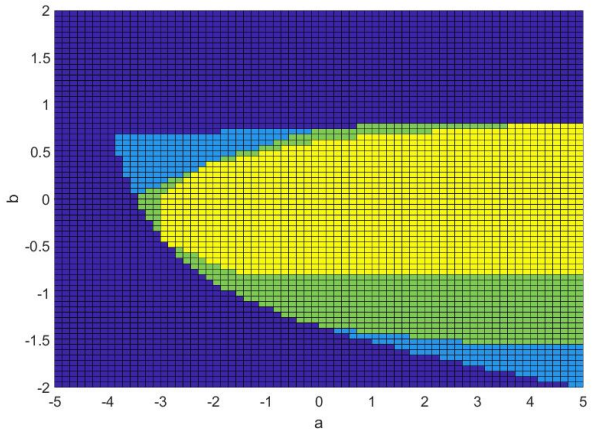

Figura 4.1

En azul oscuro la zona donde no es suma de cuadrados. En azul claro la zona *SOS* exclusiva, en verde *SDSOS* y en amarillo *DSOS*.

Por tanto se ve claramente cómo al resolver el problema 4.1 con programación *DSOS* o *SDSOS* estaremos reduciendo el recinto de búsqueda y perdiendo, en principio, información sobre el mínimo real.

Se adjuntan unas tablas con el resultado de resolver 4.1 para polinomios en distinto número de variables y coeficientes obtenidos aleatoriamente. Nos reducimos a formas de grado 4 en un número de variables entre 10 y 60.

Tabla de tiempos (s)			
Variables	DSOS	SDSOS	SOS
10	0.35	0.38	0.34
15	1.44	2.06	6.01
20	6.55	7.00	85.27
25	16.45	20.83	-
30	18.71	21.27	-
35	40.86	43.84	-
40	83.03	80.44	-
45	154.44	137.21	-
50	553.38	217.14	-
55	5236.67	319.73	-
60	14100.04	473.90	-

Tabla 4.1

Podemos comprobar que para un amplio rango, de 25 a 60 variables, los métodos *DSOS* y *SDSOS* ofrecen una solución mientras que la capacidad de la formulación *SOS* no es suficiente como para resolver e problema. Como hemos dicho estamos perdiendo optimalidad, pero vemos que puede ser un precio interesante a pagar si a cambio nos permite resolver problemas de gran tamaño que de otra manera no podríamos.

En la tabla se aprecia una diferencia apreciable de tiempos entre los métodos $DSOS$ y $SDSOS$ a partir de las 20 variables.

Debido a la capacidad limitada de memoria de los ordenadores con los que se han realizado estas pruebas solo alcanzamos 60 variables, pero todo parece indicar que un ordenador con más memoria RAM daría resultados para más variables, pasado el tiempo requerido.

4.2. Teoría de control

Uno de los campos en los que resulta más útil la programación semidefinida es el estudio de los sistemas dinámicos y la teoría de control. Veremos al final de la sección cómo podemos usar herramientas relacionadas con la formulación $DSOS$ y $SDSOS$ para resolverlos. Primero planteemos el problema.

4.2.1. Problema de estabilidad de sistemas lineales

Consideremos una ecuación en diferencias lineal.

$$x[k+1] = Ax[k], \qquad x[0] = x_0 \tag{4.3}$$

Esta clase de ecuación recursiva es un ejemplo de sistema dinámico discreto en el tiempo. La variable $x[k]$ evoluciona con el tiempo, comenzando en la condición inicial x_0. La ecuación análoga, para el caso de tiempo continuo sería $\frac{d}{dt}x(t) = Ax(t)$. Este modelo se usa para modelar la evolución en el tiempo de magnitudes como el voltaje de un circuito, la temperatura de objetos, la cantidad de población o la concentración química de una sustancia.

Una cuestión que se estudia es el comportamiento de estos sistemas cuando $k \to \infty$. Es razonable preguntarse bajo qué condiciones podemos garantizar que la variable $x[k]$ se mantiene acotada o incluso converge a 0.

Se puede probar que $x[k]$ converge a 0 para cualquier valor de x_0 si y solamente si el radio espectral de la matriz A es menor que 1, es decir, los au-

tovalores λ_i cumplen $|\lambda_i(A)| < 1$ para $i = 1, \ldots, n$. Efectivamente si tomamos $x[k] = A^k x_0$ y suponemos que la matriz A es diagonalizable, tendremos que

$$x[k] = A^k x_0 = (PDP^{-1})^k x_0 = PD^k P^{-1} x_0$$

Donde $x[k] = \underline{0}$ si y solo si $P^{-1}x[k] = \underline{0}$ ya que es una matriz no singular. Hacemos por tanto un cambio de variable y queda $y[k] = D^k y_0$ que es el vector de los valores y_{i0} multiplicados por λ_i^k. Es claro que $y[k]_i = \lambda_i^k y_{i0}$ tiende a 0 con k si y solo si $|\lambda_i(A)| < 1$ para $i = 1, \ldots, n$.

Decimos, por tanto, en este caso que la matriz A es estable en el sentido Schur. Esta caracterización de estabilidad puede ser muy útil, pero hay otros puntos de vista más convenientes. Damos ahora otra caracterización que proviene del concepto función de Lyapunov $V(x[k]) = x[k]^T P x[k]$.

Teorema 4.2.1 *Dada una matriz $A \in \mathbb{R}^{n\times n}$, las siguientes condiciones son equivalentes:*

1. *Todos los autovalores de A se encuentran dentro del disco unidad, es decir, $|\lambda_i(A)| < 1$ para todo $i = 1, \ldots, n$.*
2. *Existe una matriz $P \in \mathcal{S}^n$ tal que*

$$P \succ 0, \quad A^T P A - P \prec 0 \tag{4.4}$$

PRUEBA: Veamos cómo la segunda afirmación implica la primera. Sea $Av = \lambda v$, donde $v \neq 0$. Entonces

$$0 > v^* \left(A^T P A - P\right) v = \left(|\lambda|^2 - 1\right) \underbrace{v^* P v}_{>0}$$

y por tanto $|\lambda| < 1$.

Contrariamente sea $P := \sum_{k=0}^{\infty} \left(A^k\right)^T A^k$. La suma converge por la suposición hecha de los autovalores. Entonces

$$A^T P A - P = \sum_{k=1}^{\infty} \left(A^k\right)^T A^k - \sum_{k=0}^{\infty} \left(A^k\right)^T A^k = -I \prec 0$$

□

Por tanto, podemos estudiar la estabilidad del sistema mediante programción semidefinida ya que basta comprobar que las expresiones matriciales 4.4 son semidefinidas positivas. Si hacemos uso del lema 2.2.2 sabemos que una matriz M es semidefinida positiva si y solo si la forma $y^T My$ es no negativa. Podemos por tanto, imponer en su lugar que sea $DSOS$ o $SDSOS$ y tratar el problema de la manera deseada.

4.2.2. Diseño de control

Consideremos ahora la situación en la que hay una perturbación impuesta que podemos controlar $u[k]$.

$$x[k+1] = Ax[k] + Bu[k], \qquad x[0] = x_0 \tag{4.5}$$

donde $B \in \mathbb{R}^{n\times m}$. Lo que se pretende es hallar la perturbación $u[k] \in \mathbb{R}^m$ adecuada para conseguir que $x[k]$ tenga el comportamiento en el tiempo que deseemos. Estudiamos el caso en el que la matriz A no es estable y escogemos $u[k] = Kx[k]$ con K una matriz fijada. Es fácil ver que nos hallamos en el caso 4.3 con $A(K) = A + BK$. Si nuestro objetivo es obtener un sistema estable tendremos que encontrar una matriz K tal que $A + BK$ sea estable. Este problema es en principio bastante complicado ya que la dependencia de los autovalores de $A + BK$ con K no es lineal. Sin embargo, podemos reescribir las condiciones

$$(A + BK)^T P(A + BK) - P \prec 0, \quad P \succ 0$$

como

$$\begin{bmatrix} P & (A+BK)^T P \\ P(A+BK) & P \end{bmatrix} \succ 0$$

Aunque esta forma es más tratable, todavía no tiene la forma de un problema de programación semidefinida. Tomemos $Q := P^{-1}$ y multipliquemos a la derecha e izquierda la ecuación anterior por la matriz diagonal por bloques con dos bloques no nulos Q. Obtenemos

$$\begin{bmatrix} Q & Q(A+BK)^T \\ (A+BK)Q & Q \end{bmatrix} \succ 0$$

Todavía no es lineal en las variables Q y K por lo que definimos $Y := KQ$ y tenemos que se debe cumplir

$$\begin{bmatrix} Q & QA^T + Y^T B^T \\ AQ + BY & Q \end{bmatrix} \succ 0$$

Ahora el problema ya está formulado de manera que la matriz que debe ser semidefinida positiva es lineal en el par (Q, Y). Queda expresada como deseábamos y por tanto actuando como en el caso anterior, podremos usar métodos de comprobación $DSOS$ y $SDSOS$ para resolverlo.

4.3. Estabilidad de Lyapunov

Otra aplicación en el ámbito de control de sistemas es hallar funciones de Lyapunov adecuadas para sistemas diferenciales que no tienen por qué ser lineales. Introducimos algunos conceptos. Consideramos el sistema diferencial autónomo $y' = f(y)$ con $f : D \longrightarrow \mathbb{R}^N$ cuyas componentes son polinomios y

$D \subseteq \mathbb{R}^N$.

Definición 4.3.1 *Sean $\rho \in (0,\infty)$ y $V \in C^0\left(\overline{B}_\rho\right)$ una función real. Decimos que V es definida positiva en B_ρ si $V(0) = 0$ y $V(y) > 0$ para cualquier $y \in \overline{B}_\rho \backslash \{0\}$. Del mismo modo, se dice que V es definida negativa en B_ρ si $-V$ es definida positiva en B_ρ.*

Ejemplo 4.3.1 *La función $V(y) = \sum_{i=1}^N y_i^2$ cumple ser definida positiva en cualquier conjunto B_ρ. Sin embargo la función en $\mathbb{R}^3$ dada por $V(y_1, y_2, y_3) = y_1^2 + y_2^2$ no es definida positiva (ni negativa) en ninguna bola B_ρ, con $\rho > 0$.*

Definición 4.3.2 *Sea $V : B_\rho \longrightarrow \mathbb{R}$ una función tal que $V \in C^1(B_\rho)$ $(\rho > 0)$.*

- *Se denomina derivada respecto del sistema autónomo $y' = f(y)$ a la función $\dot{V} : B_\rho \longrightarrow \mathbb{R}$ dada por*

$$\dot{V}(y) = \sum_{i=1}^N \frac{\partial V}{\partial y_i}(y) f_i(y), \quad \forall y \in B_\rho$$

- *Se dice que $V \in C^1(B_\rho) \cap C^0\left(\overline{B}_\rho\right)$ es una función de Liapunov en B_ρ para el sistema autónomo $y' = f(y)$ si V es definida positiva en B_ρ y*

$$\dot{V}(y) \leq 0, \quad \forall y \in B_\rho$$

Es un resultado conocido [FB89, Teorema 5.2] que en estas condiciones si la función $\dot{V}$ es definida negativa en B_ρ entonces las solución de $y' = f(y)$ es uniformemente asintóticamente estable. Por tanto, para estudiar la estabilidad del sistema bastará con comprobar las condiciones

$$V(x) > 0, \qquad \dot{V}(x) = \left(\frac{\partial V}{\partial x}\right)^T f(x) < 0 \tag{4.6}$$

Si imponemos que la función de Lyapunov sea polinomial podremos sustituir estas condiciones por

$$V(x) \text{ sos/dsos/sdsos}, \qquad -\dot{V}(x) = -\left(\frac{\partial V}{\partial x}\right)^T f(x) \text{ sos/dsos/sdsos} \tag{4.7}$$

Se ha planteado el problema de manera que podemos tratarlo con las herramientas explicadas. Como hemos visto comprobar que un polinomio es no negativo cuesta mucho computacionalmente, por lo que estas alternativas, sobretodo para un gran número de variables pueden suponer una alternativa razonable. Puede que si imponemos *sdsos/sdsos* no obtengamos solución ya que estamos restringiendo nuestras posibilidades frente al caso *sos*. Es el precio a pagar por la ganancia en eficiencia, que en unos casos nos reduce el tiempo de computación y en otros nos da una solución cuando mediante programación *sos* no se consigue.

Ejemplo 4.3.2 *Suponemos el sistema*

$$\begin{aligned}
\dot{x}_1 &= -x_1^3 - x_2x_3 - x_1 - x_1x_3^2 \\
\dot{x}_2 &= -x_1x_3 + 2x_1^3 - x_2 \\
\dot{x}_3 &= -x_3 + 2x_1^2
\end{aligned}$$

Proponemos como función de Lyapunov $V(x) = \frac{1}{2}\left(x_1^2 + x_2^2 + x_3^2\right)$. *Comprobemos si* $\dot{V}(x)$ *es definida negativa.*

Usando las funciones del paquete [Meg10] verificamos que $-\dot{V}(x)$ *es suma de cuadrados.*

$$-\dot{V}(x) = x_1^2 + x_3^2 + \left(x_1^2 - x_1x_3 - x_2\right)^2$$

Sin embargo, al comprobar si es sdsos o dsos la respuesta es negativa. En esta ocasión la utilidad de la programación sdsos y dsos se ve limitada.

4.3.1. Búsqueda de función de Lyapunov

Otra aplicación es la búsqueda de la función de Lyapunov que cumpla las condiciones requeridas. Si proponemos una función $V(x)$ polinomial y dejamos sus coeficientes indeterminados, podremos optimizar sobre ellos para encontrar una solución factible mediante desigualdades matriciales. Expliquémoslo con un ejemplo, obtenido de la referencia [Par00].

Ejemplo 4.3.3 *Sea el sistema dinámico no lineal*

$$\begin{aligned}\dot{x}_1 &= -x_1 - 2x_2^2\\ \dot{x}_2 &= -x_2 - x_1x_2 - 2x_2^3\end{aligned}$$

Observemos que el sistema guarda una simetría, es invariante frente al cambio de signo de la segunda componente. Podríamos usar esta cualidad para limitar la búsqueda a funciones de Lyapunov con la misma simetría, pero en principio no procedemos así. En su lugar propongamos la función $V(x) = c_{20}x_1^2 + c_{02}x_2^2 + c_{11}x_1x_2 + c_{40}x_1^4 + c_{21}x_1^2x_2$*. La expresamos de manera matricial* $V(x) = \frac{1}{2}z^TQz$*.*

$$Q = \begin{bmatrix} 2c_{20} & 0 & c_{21}+\lambda_2 & c_{12}+\lambda_1 & c_{11} \\ 0 & 2c_{10} & 0 & -\lambda_3 & -\lambda_2 \\ c_{21}+\lambda_2 & 0 & 2\lambda_3 & 0 & -\lambda_1 \\ \lambda_1 & -\lambda_3 & 0 & 0 & 0 \\ c_{11} & -\lambda_2 & -\lambda_1 & 0 & 2c_{02} \end{bmatrix}$$

con $z = [x_1, x_1^2, x_1x_2, x_2^2, x_2]^T$*. Los factores* λ_i *son arbitrarios y se dejan como variables. Calculamos* $\dot{V}(x) = -2c_{20}x_1^2 - 2c_{11}x_1x_2 - 4c_{40}x_1^4 - (3c_{21} + c_{11})x_1^2x_2 - (2c_{02} + 4c_{20})x_1x_2^2 - 2c_{11}x_2^3 - 8c_{40}x_1^3x_2^2 - (4c_{21} + 2c_{11})x_1x_2^3 - 2c_{02}x_2^2 - c_{21}x_1^3x_2 - 4c_{02}x_2^4 - 2c_{21}x_1^2x_2^3$ *y la expresamos de manera matricial* $\dot{V}(x) =$

$-\frac{1}{2}w^T Rw$, *con* $w = [x_1, x_1^2, x_1x_2, x_2^2, x_2, x_1x_2^2, x_1^2x_2, x_2^3]^T$. *La matriz* R *queda como la concatenación horizontal de las siguientes dos matrices*

$$\begin{bmatrix} -4c_{20} & 0 & -3c_{21}-c_{11}+\nu_6 & -2c_{02}+4c_{20}+\nu_2 \\ 0 & -8c_{40} & -c_{21}+\nu_{13} & -\nu_8+\nu_9 \\ -3c_{21}-c_{11}+\nu_6 & -c_{21}+\nu_{13} & 2\nu_7 & -\nu_3+\nu_4 \\ -2c_{02}+4c_{20}+\nu_2 & -\nu_8+\nu_9 & -\nu_3+\nu_4 & -8c_{02}+2\nu_1 \\ -2c_{11} & -\nu_6 & -\nu_2 & 0 \\ -\nu_7+\nu_8 & -8c_{40}+\nu_{14} & -2c_{21}+\nu_{10} & -\nu_5 \\ -\nu_{13} & 0 & -\nu_{14} & -\nu_{11} \\ -4c_{21}-2c_{11}+\nu_3 & -\nu_{10}+\nu_{11} & \nu_5 & 0 \end{bmatrix};$$

$$\begin{bmatrix} -2c_{11} & -\nu_7+\nu_8 & -\nu_{13} & -4c_{21}-2c_{11}+\nu_3 \\ -\nu_6 & -8c_{40}+\nu_{14} & 0 & -\nu_{10}+\nu_{11} \\ -\nu_2 & -2c_{21}+\nu_{10} & -\nu_{14} & \nu_5 \\ 0 & -\nu_5 & -\nu_{11} & 0 \\ -4c_{02} & -\nu_4 & -\nu_9 & -\nu_1 \\ -\nu_4 & 2\nu_{12} & 0 & 0 \\ -\nu_9 & 0 & 0 & -\nu_{12} \\ -\nu_1 & 0 & -\nu_{12} & 0 \end{bmatrix}$$

Matriz R

Usamos Matlab, con el mismo paquete de optimización usado hasta ahora, para obtener la solución. El código se haya en el enlace [Meg10]. Resolvemos un problema de optimización que maximiza el valor de c_{11} *(una variable aleatoria, se puede escoger la que sea) sujeto a que los polinomios* $V(x) = \frac{1}{2}z^T Qz$ *y* $\dot{V}(x) = -\frac{1}{2}w^T Rw$ *sean sos, dsos y sdsos respectivamente. Obtenemos el siguiente resultado.*

Time (SOS) : 1,2624
Optimal value (SOS): 0,61213
$V(x) = 1{,}2457x_1^2 + 1{,}5041x_2^2 + 0{,}6121x_1x_2 + 0{,}5045x_1^4 - 0{,}1056x_1^2x_2$

Time (SDSOS): 0,31878
Optimal value (SDSOS): 2,195
$V(x) = 12{,}3431x_1^2 + 14{,}8638x_2^2 + 2{,}1950x_1x_2 + 4{,}0475x_1^4 - 0{,}4278x_1^2x_2$

Time (DSOS) : 0,24586
Optimal value (DSOS): 0,81283
$V(x) = 9{,}3219x_1^2 + 10{,}3006x_2^2 + 0{,}8128x_1x_2$

Se aprecia una diferencia en tiempo de computación, y al imponer distintos tipos de polinomios, se obtienen distintas funciones de Lyapunov. En esta ocasión la optimización dsos y sdsos supone una ganancia en tiempo de computación frente a sos. Si habláramos de un sistema más complejo con más variables la diferencia de tiempo sería mayor, en incluso ocurriría que sin llegar a una solución usando una restricción sos, en el caso dsos o sdsos sí se podría obtener una función.

Señalamos que, en este caso tratamos funciones de Lyapunov que proporcionan estabilidad global, no local.

4.3.2. Estimación de la región de atracción

Dada una función de Lyapunov $V(x)$ y un sistema dinámico, queremos esta vez, estimar la región de atracción de un punto asintóticamente estable dado (tomamos el origen sin pérdida de generalidad). Proponemos una bola de radio γ en torno al origen. Nos interesa conocer la región más amplia posible que cumpla esta cualidad, por lo que buscamos maximizar γ.

Una condición suficiente para que sea región de atracción relativa a $V(x)$ es que se cumpla

$$\dot{x} = f(x), \quad V(x) > 0 \quad x \neq 0$$
$$\dot{V}(x) = \left(\frac{\partial V}{\partial x}\right)^T f(x) < 0, \quad x \neq 0$$

en dicha región. Que se cumpla esta condición es equivalente a que se cumpla

$$\left(x^T x\right)(V(x) - \rho) + L(x)\dot{V}(x) \geq 0$$

siendo $L(x)$ un polinomio multiplicador que hace de controlador. Una vez más, podemos sustituir la condición de no negatividad con la imposición de que la expresión anterior sea *sos*, *dsos* o *sdsos*. Como nos interesa maximizar γ, podemos resolver la cuestión a través del siguiente problema de optimización

$$\begin{aligned} &\max_{\gamma, L(x)} \gamma \\ &\text{s.a. } \left(x^T x\right)(V(x) - \rho) + L(x)\dot{V}(x) \quad sos,\ dsos,\ sdsos \end{aligned} \tag{4.8}$$

Veámoslo con un ejemplo.

Ejemplo 4.3.4 *Sea el sistema dinámico*

$$\dot{x} = -x + y$$
$$\dot{y} = 0{,}1x - 2y - x^2 - 0{,}1x^3$$

Imponemos que la función de Lyapunov sea $V(x, y) := x^2 + y^2$*, por lo que la derivada queda* $\dot{V}(x) = -2x_1^2 - 4x_2^2 + 2{,}2x_1x_2 - 2x_1^2x_2 - 0{,}2x_2x_1^3$*. Programamos en Matlab, con el código usado hasta ahora, la resolución del problema de optimización*

$$\begin{aligned} &\max_{\gamma, p_1, p_2, p_3, p_4} \gamma \\ &s.a.\ (V(x,y) - \gamma)\left(x^2 + y^2\right) + \left(p_1 + p_2x + p_3y + p_4xy\right)\dot{V}(x,y) \quad sos,\ dsos,\ sdsos \end{aligned}$$

Obtenemos

Time (DSOS) : 0,17276
Optimal value (DSOS): 1,1643
Time (SDSOS) : 0,14236
Optimal value (SDSOS): 3,19
Time (SOS) : 0,89092
Optimal value (SOS): 7,1115

Vemos como el radio de la región disminuye si imponemos condiciones más restrictivas. Sin embargo, el tiempo empleado en alcanzar una solución es menor en los casos dsos y sdsos que en el caso sos. Aunque ara este caso, de pocas variables la diferencia entre las regiones obtenidas sea muy significativa, en el caso de que trabajemos con un gran número de variables pueden interesarnos las formulaciones dsos y sdsos por su ganancia en eficiencia.

Esta aplicación se puede extender a casos más complejos como el control de un robot humanoide. Este aspecto es fundamental en robótica y supone un gran reto debido a que los sistemas dinámicos que rigen los movimientos del dispositivo. Las ecuaciones que definen estos sistemas son no lineales y las dimensiones del espacio de estado suelen ser elevadas. Esto hace que pueda volverse intratable, o que el tiempo que se tarda en evaluar el estado del robot no sea lo suficientemente pequeño para que éste sea funcional.

Los autores del artículo [AM19] incluyen un modelo de robot basado en el ATLAS de Boston Dynamics Inc. El objetivo es conseguir que el robot mantenga el equilibriosobre el pie derecho. El controlador que aplican se construye buscando una función de Lyapunov y una ley de control lineal que maximicen el tamaño de la región de atracción resultante. Las condiciones que debe cumplir la función de Lyapunov se imponen usando programación *sdsos*. Las ecuaciones de movimiento del sistema se expresan de manera polinomial aplicando series de Taylor de orden 3.

El tiempo total de computación es de 22.5 minutos, se señala que usando programación *sos* es imposible obtener una respuesta debido a la limitación de memoria RAM del dispositivo usado. Incluimos unos enlaces con vídeos que muestran las simulaciones del controlador para distintas situaciones iniciales http://youtu.be/lmAT556Ar5c, http://www.youtube.com/watch?v=SD6Okylclb8.

Bibliografía

[AM19] Amir Ali Ahmadi and Anirudha Majumdar, *DSOS and SDSOS optimization: more tractable alternatives to sum of squares and semidefinite optimization*, SIAM J. Appl. Algebra Geom. **3** (2019), no. 2, 193–230.

[BC75] George Phillip Barker and David Carlson, *Cones of diagonally dominant matrices*, Pacific J. Math. **57** (1975), no. 1, 15–32.

[BCPT05] Erik G. Boman, Doron Chen, Ojas Parekh, and Sivan Toledo, *On factor width and symmetric h-matrices*, Linear Algebra and its Applications **405** (2005), 239 – 248.

[BCR98] Jacek Bochnak, Michel Coste, and Marie-Françoise Roy, *Real algebraic geometry*, Ergebnisse der Mathematik und ihrer Grenzgebiete (3) [Results in Mathematics and Related Areas (3)], vol. 36, Springer-Verlag, Berlin, 1998, Translated from the 1987 French original, Revised by the authors.

[BPR06] Saugata Basu, Richard Pollack, and Marie-Françoise Roy, *Algorithms in real algebraic geometry*, second ed., Algorithms and Computation in Mathematics, vol. 10, Springer-Verlag, Berlin, 2006. MR 2248869

[BTN01] Aharon Ben-Tal and Arkadi Nemirovski, *Lectures on modern convex optimization*, MPS/SIAM Series on Optimization, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA; Mathematical Programming Society (MPS), Philadelphia, PA, 2001, Analysis, algorithms, and engineering applications.

[CLR80] Man Duen Choi, Tsit Yuen Lam, and Bruce Reznick, *Real zeros of positive semidefinite forms. I*, Math. Z. **171** (1980), no. 1, 1–26.

[DBFC98] Dr. Bob F. Caviness Dr. Jeremy R. Johnson (eds.) Dr. Bob F. Caviness, Dr. Jeremy R. Johnson (auth.), *Quantifier elimination and cylindrical algebraic decomposition*, 1 ed., Texts and Monographs in Symbolic Computation, Springer-Verlag Wien, 1998.

[DOW55] George B. Dantzig, Alex Orden, and Philip Wolfe, *The generalized simplex method for minimizing a linear form under linear inequality restraints*, Pacific J. Math. **5** (1955), 183–195.

[FB89] John A. Nohel Fred Brauer, *The qualitative theory of ordinary differential equations: An introduction*, Dover Publications, 1989.

[Ful89] William Fulton, *Algebraic curves*, Advanced Book Classics, Addison-Wesley Publishing Company, Advanced Book Program, Redwood City, CA, 1989, An introduction to algebraic geometry, Notes written with the collaboration of Richard Weiss, Reprint of 1969 original.

[Ger31] S Gervsgorin, *Uber die abgrenzung der eigenwerte einer matrix*, 749–754.

[Hil88] David Hilbert, *Ueber die Darstellung definiter Formen als Summe von Formenquadraten*, Math. Ann. **32** (1888), no. 3, 342–350.

[Meg10] Alexander Megretsky, *Systems polynomial optimization tools (spot)*, https://github.com/spot-toolbox/spotless, 2010.

[Mey00] Carl Meyer, *Matrix analysis and applied linear algebra*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000, With 1 CD-ROM (Windows, Macintosh and UNIX) and a solutions manual (iv+171 pp.).

[MSB77] John J. Jarvis Mokhtar S. Bazaraa, *Linear programming and network flows*, first edition ed., Wiley, 1977.

[Par00] Pablo Parrilo, *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*, Ph.D. thesis, California Institute of Technology, 2000.

[Par13] Pablo A. Parrilo, *Semidefinite optimization*, Semidefinite optimization and convex algebraic geometry, MOS-SIAM Ser. Optim., vol. 13, SIAM, Philadelphia, PA, 2013, pp. 3–46.

[Rez78] Bruce Reznick, *Extremal PSD forms with few terms*, Duke Math. J. **45** (1978), no. 2, 363–374.