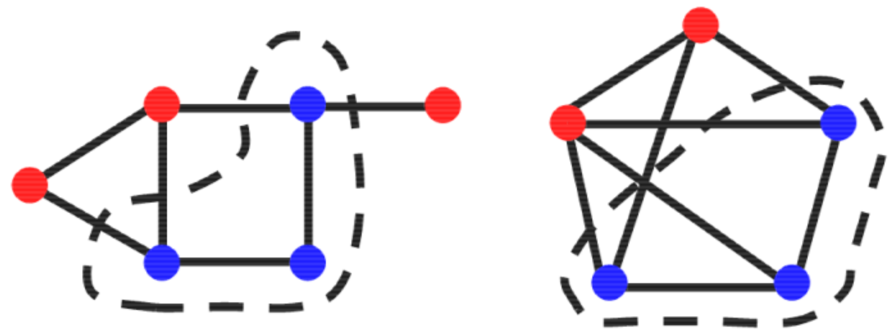
Optimización Semidefinida - Segundo Cuatrimestre 2021

Práctica 3 - Aplicaciones SDP

Para entregar: entregar un ejercicio a elección entre los marcados (♢).

Max-cut

1. En los siguientes ejemplos, se considera que el costo de las aristas que aparecen dibujadas en el grafo es 1, y que el costo de las aristas que no aparecen es 0. Llamamos K al conjunto de puntos azules dentro de la región marcada.



Para cada uno de los ejemplos:

(a) Calcular a mano el costo de los cortes marcados.

(b) Construir la matriz $\boldsymbol{Q}$ de costos.

(c) Numerar los nodos de 1 a n y representar el corte marcado como un vector $\boldsymbol{x} \in \{-1, 1\}^n$, donde $x_i = 1$ sii i pertenece a K

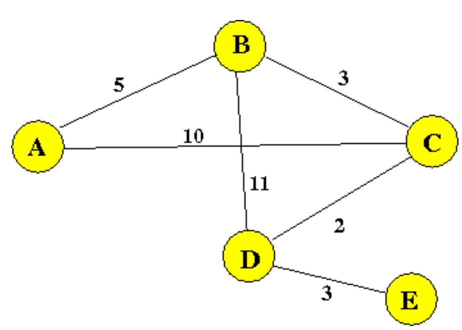
(d) Construir una matriz $\boldsymbol{C}$ tal que el costo de un corte determinado por un vector $\boldsymbol{x}$ sea $\boldsymbol{x}^T \boldsymbol{C} \boldsymbol{x}$.

(e) Para el corte dado, verificar las cuentas en Python.

(f) Buscar, por prueba y error, algún corte que tenga mayor costo que el corte dado.

2. Implementar un programa que reciba una matriz $\boldsymbol{Q}$ de costos y calcule el corte de mayor costo probando todos los cortes posibles. Aplicar el programa a los ejemplos anteriores y verificar si el corte coincide con el corte del ejemplo o con el mejor corte que haya encontrado en el último punto.

3. Utilizar el programa anterior para hallar el corte de mayor costo en el siguiente ejemplo.



4. **Programación lineal.**

   (a) Verificar que el siguiente problema de programación entera

   $$\begin{array}{ll} \text{maximizar:} & \displaystyle\sum_{1\leq i,j,\leq n} w_{ij} z_{ij} \\ \text{sujeto a:} & z_{ij} \leq x_i + x_j, \forall\ 1 \leq i,j, \leq n \\ & z_{ij} \leq 2 - (x_i + x_j), \forall\ 1 \leq i,j, \leq n \\ & x_i, z_{ij} \in \{0,1\} \forall\ 1 \leq i,j, \leq n \end{array}$$

   es una formulación del problema max-cut tomando para un subconjunto K de nodos, $x_i = 1$ sii $i \in K$ y $z_{ij} = 1$ sii $(i,j) \in \delta(K)$.

   (b) Para convertirlo en una problema de programación lineal podemos relajar la restricciones $x_i, z_{ij} \in \{0,1\} \forall i,j$, a $x_i, z_{ij} \in [0,1] \forall i,j$. ¿Considera que esta relajación es una buena aproximación al problema original?

   (c) (*) Utilizando que en un conjunto cualquiera de tres nodos del grafo, no pueden estar las tres aristas en el corte, formular una mejor relajación como problema de programación lineal.

   (d) (*) Calcular en los ejemplos de los ejercicios anteriores el cociente entre el valor óptimo de la relajación y el valor óptimo del problema original.

5. Implementar en Mosek la relajación SDP del problema max-cut. Aplicar el programa en los ejemplos anteriores y comparar el costo óptimo del problema original con el valor óptimo del problema relajado.

6. ($\diamondsuit$) Demostrar que la solución óptima del problema SDP

   $$\begin{array}{ll} \text{minimizar:} & \operatorname{Tr}(\Lambda) \\ \text{sujeto a:} & \boldsymbol{C} \preceq \Lambda \\ & \Lambda \text{ diagonal} \end{array}$$

   es una cota superior del costo óptimo $\boldsymbol{x}^T \boldsymbol{C} \boldsymbol{x}$, $\boldsymbol{x} \in \{-1,+1\}^n$, del problema max-cut.

   Verificar que dicho problema es el problema dual de la relajación SDP vista en clase del problema SDP.

   Implementar este problema en Mosek y calcular el óptimo para el caso de un pentágono.

7. Implementar un programa que dado $n \in \mathbb{N}$, devuelva una matriz simétrica cuyas entradas sean números aleatorios entre 0 y 1. Sugerencia: utilizar el comando `np.random.rand`.

8. Comparar el costo óptimo del problema original con el valor óptimo del problema relajado para matrices aleatorias de tamaño 5, 10 y 15.

9. (*) Dadas n variables aleatorias $r_1, \ldots, r_n$ con distribución normal (gaussiana) $N(0,1)$, probar que el vector

   $$\boldsymbol{v} = \frac{(r_1, \ldots, r_n)}{\|(r_1, \ldots, r_n)\|_2}$$

   es un vector en la esfera unitaria con distribución uniforme sobre la esfera.

   Sugerencia: probar que la densidad conjunta de $(r_1, \ldots, r_n)$ solo depende de la norma del vector.

10. Implementar en Python un programa que genere un vector aleatorio en la esfera unitaria de dimensión n, con distribución uniforme sobre la misma.

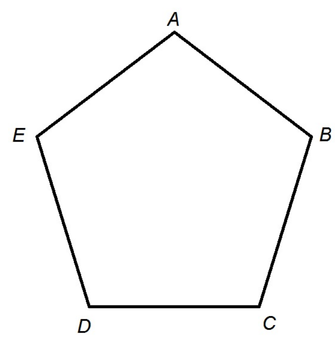
11. Implementar un programa que dado un vector unitario $\boldsymbol{r} \in \mathbb{R}^n$ y una matriz $\boldsymbol{P} \in \mathbb{R}^n$ cuyas columnas son vectores unitarios, devuelva un vector $\boldsymbol{x} \in \{-1,+1\}^n$ con $x_i = +1$ sii $\boldsymbol{r} \cdot \boldsymbol{p}_i > 0$, con $\boldsymbol{p}_i$ la i-ésima columna de $\boldsymbol{P}$.

12. Implementar un programa que a partir de la matrix $\boldsymbol{X}$ que se obtiene como solución óptima de la relajación SDP del problema max-cut con valor óptimo s, calcule un vector $\boldsymbol{x} \in \{-1,+1\}^n$ tal que el corte asociado a dicho vector satisfaga

$$w(\delta(\boldsymbol{x})) \geq 0.878s$$

13. En todos los ejemplos de los ejercicios anteriores, calcular la razón entre las soluciones óptimas del problema maxcut original y la relajación SDP.

14. Calcular la razón entre las soluciones óptimas del problema maxcut original y la relajación SDP para el pentágono



tomando costo 1 para los lados del pentágono y costo 0 para las demás aristas.

Teoría de control

15. Para la matriz

$$\boldsymbol{P} = \begin{pmatrix} 3 & x \\ x & 3 \end{pmatrix},$$

(a) calcular los autovalores en función de x.

(b) resolver a mano el problema SDP

$$\begin{array}{ll} \text{minimizar:} & x_{11} \\ \text{sujeto a:} & \boldsymbol{P} \succ 0, \end{array}$$

(c) resolver a mano el problema SDP

$$\begin{array}{ll} \text{minimizar:} & x_{11} \\ \text{sujeto a:} & \boldsymbol{P} \succeq \boldsymbol{I}, \end{array}$$

(d) resolver a mano el problema SDP

$$\begin{array}{ll} \text{minimizar:} & \eta \\ \text{sujeto a:} & \boldsymbol{I} \preceq \boldsymbol{P} \preceq \eta \boldsymbol{I}, \end{array}$$

16. ($\diamondsuit$) Implementar un programa que reciba una matriz $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ y resuelva el problema SDP

$$\begin{aligned} \text{minimizar:} \quad & p_{00} \\ \text{sujeto a:} \quad & \boldsymbol{A}^T\boldsymbol{P}\boldsymbol{A} - \boldsymbol{P} \succ 0, \\ & \boldsymbol{P} - \boldsymbol{I} \succeq 0, \end{aligned}$$

17. ($\diamondsuit$) Implementar un programa que reciba una matriz $\boldsymbol{A} \in \mathbb{R}^{2\times 2}$ y resuelva el problema SDP

$$\begin{aligned} \text{minimizar:} \quad & \eta \\ \text{sujeto a:} \quad & \boldsymbol{A}^T\boldsymbol{P}\boldsymbol{A} - \boldsymbol{P} \succ 0, \\ & \boldsymbol{P} - \boldsymbol{I} \succeq 0, \\ & \eta\boldsymbol{I} - \boldsymbol{P} \succeq 0 \end{aligned}$$

18. Para las siguientes matrices $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, $\boldsymbol{B} \in \mathbb{R}^{n\times m}$, determinar si existe $\boldsymbol{K} \in \mathbb{R}^{m\times n}$ tal que $\rho(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K}) \leq 1$.

    (a) $\boldsymbol{A} = \begin{pmatrix} 10 & 0 \\ 0 & 0.5 \end{pmatrix}$, $\boldsymbol{B} = \begin{pmatrix} 2 & 3 \\ 1 & 4 \end{pmatrix}$.

    (b) $\boldsymbol{A} = \begin{pmatrix} 10 & 0 \\ 0 & 0.5 \end{pmatrix}$, $\boldsymbol{B} = \begin{pmatrix} 2 & 0 \\ 1 & 0 \end{pmatrix}$.

    (c) $\boldsymbol{A} = \begin{pmatrix} 10 & 0 \\ 0 & 0.5 \end{pmatrix}$, $\boldsymbol{B} = \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix}$.

    Intentar resolver los problemas analíticamente y verificar los resultados con Mosek.

19. Un sistema de la forma

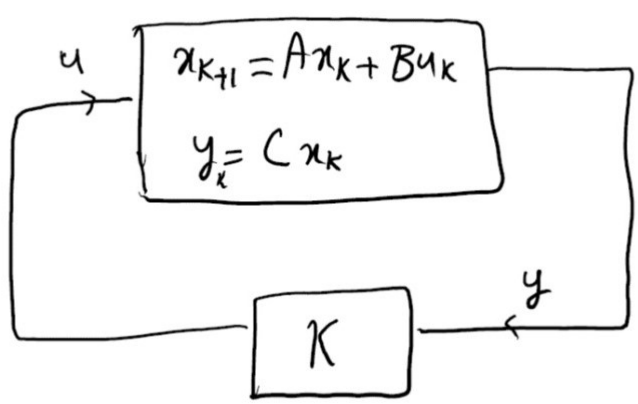
$$\boldsymbol{x}[k+1] = \boldsymbol{A}\boldsymbol{x}[k] + \boldsymbol{B}\boldsymbol{u}[k], \quad \boldsymbol{x}[0] = \boldsymbol{x}_0,$$

tiene un *modo no estabilizable* si la matriz $\boldsymbol{A}$ tiene un autovector a izquierda $\boldsymbol{w}$ tal que

$$\boldsymbol{w}^T\boldsymbol{A} = \lambda\boldsymbol{w}^T, \quad \boldsymbol{w}^T\boldsymbol{B} = 0 \quad \text{y} \quad |\lambda| \geq 1$$

- Probar que en ese caso, el problema SDP no es factible.
- Interpretar el resultado en término de los autovalores de $\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K}$.

20. (Estabilización con retroalimentación de salida.) Considerar el siguiente sistema de control para matrices $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, $\boldsymbol{B} \in \mathbb{R}^{n\times k}$ y $\boldsymbol{C} \in \mathbb{R}^{k\times r}$:



(a) Verificar que la estabilidad del sistema es equivalente a la existencia de una matriz $\boldsymbol{K} \in \mathbb{R}^{k \times r}$ tal que $\rho(A + BKC) < 1$.

(b) Plantear este problema como un problema SDP. Nota: la existencia de dicho problema SDP ¡es un problema abierto!

Los ejercicios marcados (*) son optativos y pueden involucrar temas no vistos en la materia.