



INSTITUTO TECNOLÓGICO AUTÓNOMO DE MÉXICO

## Optimización Numérica | Proyecto 1 – Reporte

Juan Carlos Sigler

Alonso Martinez

Paulina Carretero

### Índice general

<b>1</b>	<b>Marco teórico</b>	<b>1</b>
1.1	Algunos comentarios sobre la elección de lenguaje de programación . . . . .	1
<b>2</b>	<b>Problemas</b>	<b>1</b>
2.1	Problema 1: Problema chico . . . . .	2
2.2	Problema 2: Klee-Minty . . . . .	2
2.3	Problema 2: Problema Woodinfe . . . . .	3
<b>3</b>	<b>Conclusiones (?)</b>	<b>4</b>

### 1 Marco teórico

En el presente documento discutimos una implementación del algoritmo del conjunto activo para resolver problemas de programación cuadrática con restricciones tanto de igualdad como desigualdad, como en el problema (P) a continuación:

$$\min \frac{1}{2} \vec{x}^\top G \vec{x} + \vec{c}^\top \vec{x} \quad (\text{P})$$

#### 1.1 Algunos comentarios sobre la elección de lenguaje de programación

### 2 Problemas

Probamos nuestra implementación con tres problemas y presentamos los resultados de acuerdo a lo especificado en la asignación del proyecto.

## 2.1 Problema 1: Problema chico

$$\min q(x) = (x - 1)^2 + (y - 2.5)^2$$

Sujeto a

$$-x + 2y - 2 \leq 0$$

$$x + 2y - 6 \leq 0$$

$$x - 2y - 2 \leq 0$$

$$-x \leq 0$$

$$-y \leq 0$$

Con  $x_0 = (2, 0)^\top$  &  $W_0 = \{3\}$ .

Para este problema, nuestro algoritmo imprime lo siguiente:

```
1 Rama 1. ||d_k|| = 2.5, q(x) = -1.8125, α=0.5 k = 0
2 Rama 1. ||d_k|| = 0.9, q(x) = -0.8000000000000007, α=1.6666666666666667
3 Rama 2. j = 2, μ=0.0
```

Obtenemos que el punto Karush-Kuhn-Tucker (KKT) es:

$$(\vec{x}^*, \vec{\mu}^*) = \left( \begin{bmatrix} 1.4 \\ 1.7 \end{bmatrix}, \begin{bmatrix} 0.399 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \right)$$

## 2.2 Problema 2: Klee-Minty

$$\begin{aligned} \min & \quad \frac{1}{2} \vec{x}^\top G \vec{x} - \sum_{i=1}^n x_i \\ \text{sujeto a} & \quad x_1 \leq 1 \\ & \quad 2 \sum_{j=1}^{i-1} x_j + x_i \leq 2^i - 1, \quad i = 2, \dots, n \\ & \quad x_1, \dots, x_n \geq 0 \end{aligned}$$

Sea  $n = 15$ . Aplica el método empezando con  $W_0$  un subconjunto aleatorio de 5 entradas de los últimos 10 restricciones de positividad.

Falta documentar:

- $W_0$
- Numero de iteraciones

Para este problema, nuestro algoritmo imprime lo siguiente:

```
1 Rama 2. j = 1, μ=0.0
```

y muestra que el óptimo es

$$(\vec{x}^*, \vec{\mu}^*) = \left( \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 1 \end{bmatrix} \right)$$

La representación completa del vector  $\vec{\mu}$  es

```
1 [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0,
  1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
```

El valor óptimo de la función objetivo en  $\vec{x}_*$  es 0.0.

Para este problema la rutina quadprog de Matlab imprime:

```
1 >> quadprog(G,c,A,b)
2
3 Minimum found that satisfies the constraints.
4
5 Optimization completed because the objective function is non-decreasing in
6 feasible directions, to within the value of the optimality tolerance,
7 and constraints are satisfied to within the value of the constraint tolerance.
8
9
10 ans =
11
12     1.0e-10 *
13
14     0.2220
15     0.1175
16     0.0685
17     0.0744
18     0.1014
```

Nuestra solución coincide con la de Matlab salvo error de redondeo.

## 2.3 Problema 2: Problema Woodinfe

$$\begin{aligned} &\text{minimizar} && \frac{1}{2}\vec{x}^\top G\vec{x} + \vec{c}^\top \vec{x} \\ &\text{sujeto a} && A\vec{x} = \vec{b} \\ & && x_i \geq \ell_i \quad \text{si } \ell_i \text{ es finito,} \\ & && x_i \leq u_i \quad \text{si } u_i \text{ es finito.} \end{aligned}$$

- Definimos  $J \subset I$  como sigue:

- Para  $x_j \geq \ell_j$  definimos  $|g_j(x_0)| \leq 8\varepsilon_m \max\{|\ell_j|, 1\} \implies j \in J$
- Para  $x_j \leq u_j$  definimos  $|g_j(x_0)| \leq 8\varepsilon_m \max\{|u_j|, 1\} \implies j \in J$

Donde  $\varepsilon_m$  es el épsilon de la máquina `eps(Float64)`.

A continuación presentamos el código que se utilizó para construir el problema y encontrar las restricciones que pertenecen a  $J$ . El código se presenta recortado (excluimos el código para obtener los datos del archivo .mat y comentarios aclaratorios) en interés de la brevedad, pero se puede encontrar código completo en el script incluido `script3.ipynb`.

```
1 n_eq = length(b)
2 A_eq = problem["A"]
3 G = I(length(c))
4
5 # Como all(isfinite.(1)) == true, se complen todas las cotas inferiores
6 A = [A_eq; -I(length(1))]
7 b = [b; 1]
```

