ENG1467 - OTIMIZAÇÃO - PUC-Rio, 2022.1

Lista 2: Otimização

18/04/2022

Professor: Alexandre Street *Aluno: Thiago Novaes*

O código completo desenvolvido durante essa atividade pode ser encontrado em `https://github.com/Thiago-NovaesB/MestradoPuc.jl/tree/main/Programa%C3%A7%C3%A3o%20Linear/Lista%202`

1 Questão 1

1.1 Parte I

Como a função é convexa e linear por partes, podemos minimizar a projeção do epígrafo da função no eixo vertical. A figura nos fornece os coeficientes angulares das retas e as raízes:

$$0 = -5 * 3 + b \rightarrow b = 15$$
$$0 = 3 * 5 + b \rightarrow b = -15$$

Obtemos então:

$$\begin{array}{ll} \text{minimize} & y \\ \text{subject to} & y \geq -5x + 15 \\ & y \geq 3x - 15 \\ & y \geq 0 \end{array} \tag{1}$$

1.2 Parte II

1.2.1 (a)

Usando o fato que a função módulo é linear por partes e convexa, podemos trocar a parte não linear da função objetivo usando o epígrafo da função. A restrição apresenta a soma de 2 módulos menor ou igual a uma constante, isso pode ser substituído por 4 desigualdades lineares, uma vez que existem 4 possibilidades de sinais para as expressões dentro dos módulos. Obtemos assim:

$$\begin{array}{ll} \text{minimize} & 2x_1 + 3z \\ \text{subject to} & z \geq x_2 - 10 \\ & z \geq -x_2 + 10 \\ & 5 \geq x_1 + 2 + x_2 \\ & 5 \geq x_1 + 2 - x_2 \\ & 5 \geq -x_1 - 2 + x_2 \\ & 5 \geq -x_1 - 2 - x_2 \end{array} \tag{2}$$

1.2.2 (b)

De forma geral, o módulo na função objetivo faria a troca de minimização por maximização deixar o problema não linear, porém, uma particularidade desse problema possibilitará isso. Perceba que a restrição do problema pode ser escrita como $|x_2| \leq 5 - |x_1 + 2|$, isso significa $|x_2| \leq 5$, o que significa que $x_2 \in [-5, 5]$.

Perceba que para $x_2 \in [-5, 5]$ teremos que $|x_2 - 10|$, é equivalente à $-x_2 + 10$, ou seja, apesar de termos um módulo na função objetivo, o conjunto viável restringe o domínio dessa função para uma parte linear. Sendo assim:

$$\begin{array}{ll} \text{maximize} & 2x_1 - 3x_2 + 30 \\ \text{subject to} & 5 \geq x_1 + 2 + x_2 \\ & 5 \geq x_1 + 2 - x_2 \\ & 5 \geq -x_1 - 2 + x_2 \\ & 5 \geq -x_1 - 2 - x_2 \end{array} \tag{3}$$

2 Questão 2

2.1 a

$$\begin{array}{ll} \text{minimize} & 100g_1 + 150g_2 + 200g_3 + 5000g_d \\ \text{subject to} & g_1 + f_{1,2} - f_{3,1} = 0 \\ & g_2 + f_{2,3} - f_{1,2} = 0 \\ & g_3 + f_{3,1} - f_{2,3} + g_d = 15 \\ & -20 \leq f_{1,2} \leq 20 \\ & -20 \leq f_{2,3} \leq 20 \\ & -5 \leq f_{3,1} \leq 5 \\ & 0 \leq g_1 \leq 5 \\ & 0 \leq g_2 \leq 20 \\ & 0 \leq g_3 \leq 12 \end{array} \tag{4}$$

2.2 b

$$
\begin{array}{ll}
\text{minimize} & 100g_1 + 150g_2 + 200g_3 + 5000g_d \\
\text{subject to} & g_1 + f_{1,2} - f_{3,1} = 0 \\
& g_2 + f_{2,3} - f_{1,2} = 0 \\
& g_3 + f_{3,1} - f_{2,3} + g_d = 15 \\
& -20 \leq f_{1,2} \leq 20 \\
& -20 \leq f_{2,3} \leq 20 \\
& -5 \leq f_{3,1} \leq 5 \\
& -20 \leq f^1_{2,3} \leq 20 \\
& -5 \leq f^1_{3,1} \leq 5 \\
& -20 \leq f^2_{1,2} \leq 20 \\
& -5 \leq f^2_{3,1} \leq 5 \\
& -20 \leq f^3_{1,2} \leq 20 \\
& -20 \leq f^3_{2,3} \leq 20 \\
& 0 \leq g_1 \leq 5 \\
& 0 \leq g_2 \leq 20 \\
& 0 \leq g_3 \leq 12 \\
& g_1 - f^1_{3,1} = 0 \\
& g_2 + f^1_{2,3} = 0 \\
& g_1 + f^3_{1,2} = 0 \\
& g_3 - f^3_{2,3} + g_d = 15 \\
& g_2 - f^2_{1,2} = 0 \\
& g_3 + f^2_{3,1} + g_d = 15
\end{array}
\tag{5}
$$

2.3 c

O código baixo foi usado para resolver os 2 problemas. Com as linhas comentadas, o primeiro foi resolvido. Descomentando as linhas, o segundo foi resolvido.

```
g_max = [5.0, 20.0, 12.0]
f_max = [20.0, 20.0, 5.0]
f_c1_max = [0.0, 20.0, 5.0]
f_c2_max = [20.0, 0.0, 5.0]
f_c3_max = [20.0, 20.0, 0.0]
c = [100.0, 150.0, 200.0]
c_d = 5000
d = 15

model = Model(GLPK.Optimizer)
@variable(model, 0.0 <= g[i = 1:3] <= g_max[i])
@variable(model, 0.0 <= g_d)
@variable(model, -f_max[i] <= f[i = 1:3] <= f_max[i])
# @variable(model, -f_c1_max[i] <= f_c1[i = 1:3] <= f_c1_max[i])
# @variable(model, -f_c2_max[i] <= f_c2[i = 1:3] <= f_c2_max[i])
# @variable(model, -f_c3_max[i] <= f_c3[i = 1:3] <= f_c3_max[i])
```

```

@constraint(model, g[1] + f[1] - f[3] == 0.0)
@constraint(model, g[2] + f[2] - f[1] == 0.0)
@constraint(model, g[3] + f[3] - f[2] + g_d == d)
# @constraint(model, g[1] - f_c1[3] == 0.0)
# @constraint(model, g[2] + f_c1[2] == 0.0)
# @constraint(model, g[1] + f_c3[1] == 0.0)
# @constraint(model, g[3] - f_c3[2] + g_d == d)
# @constraint(model, g[2] - f_c2[1] == 0.0)
# @constraint(model, g[3] + f_c2[3] + g_d == d)

@objective(model, Min, sum(c[i]*g[i] for i = 1:3) + c_d*g_d)
optimize!(model)
```

	Prb 1	Prb 2 sem falha	Prb 2 falha 1	Prb 2 falha 2	Prb 2 falha 3
g_1	5	5	5	5	5
g_2	10	0	0	0	0
g_3	0	10	10	10	10
g_d	0	0	0	0	0
f_1	-10	-10	0	0	-5
f_2	-20	-10	0	0	-5
f_3	-5	-5	5	5	0
FO	2000	2500	2500	2500	2500

A segunda coluna da tabela acima indica a solução sem considerar contingência, devido as capacidades das linhas, é possível produzir com os geradores mais baratos. Um detalhe interessante é que como não existe custo atrelado ao fluxo ou perdas nas linhas, os fluxos estão levemente redundantes e desnecessariamente altos, um jeito simples de resolver isso, seria adicionar uma penalização pelos fluxos na FO, porém a penalidade precisaria ser calculada de forma a não alterar as gerações.

Já para as contingências, o primeiro gerador ainda pode ser usado, uma vez que com critério N-1, ainda é possível transferir energia. Porém o segundo gerador não pode ser usado, uma vez que a capacidade de fluxo em contingência foi totalmente usada pelo gerador 1 (quando a linha 1 ou 2 falham). Sendo assim, o gerador 3 foi usado, elevando o custo do sistema.

3 Questão 3

Como a ideia geral dessa questão é pensar na operação ao longos de vários dias, otimizando para um único dia, foi decidido conectar a hora 1 e a hora 24. Por exemplo, a restrição de rampa deve ser respeitada ao se passar da hora 24 pra a hora 1, assim como a conservação de carga na bateria. Dessa forma temos essa operação diária consistente em vários dias consecutivos. Além disso f foi usado para representar o corte de demanda, B a Carga da bateria e b a variação de carga na bateria.

3.1 a

$$\begin{array}{ll} \text{minimize} & \Sigma_{t=1}^{T}\Sigma_{i=1}^{n} g_i^t c_i \\ \text{subject to} & \Sigma_{i=1}^{n} g_i^t = d_t \qquad\qquad t = 1, ..., T \\ & -R_i^{down} \le g_i^t - g_i^{t-1} \le R_i^{up} \forall i, t = 2, ..., T \\ & -R_i^{down} \le g_i^1 - g_i^T \le R_i^{up} \ \ \forall i \\ & 0 \le g_i^t \le G_i \qquad\qquad \forall i, t \end{array} \tag{6}$$

3.2 b

$$\begin{array}{lll} \text{minimize} & \Sigma_{t=1}^{T}\Sigma_{i=1}^{n} g_i^t c_i + \Sigma_{t=1}^{T} z_t & \\ \text{subject to} & \Sigma_{i=1}^{n} g_i^t = d_t - f_t & t = 1, ..., T \\ & -R_i^{down} \le g_i^t - g_i^{t-1} \le R_i^{up} & \forall i, t = 2, ..., T \\ & -R_i^{down} \le g_i^1 - g_i^T \le R_i^{up} & \forall i \\ & z_t \ge C_1^{def} f_t & \forall t \\ & z_t \ge C_2^{def} f_t + 0.05 d_t (C_2^{def} - C_1^{def}) & \forall t \\ & 0 \le g_i^t \le G_i & \forall i, t \\ & 0 \le f^t \le d_t & \forall t \end{array} \tag{7}$$

3.3 c

$$\begin{array}{lll} \text{minimize} & \Sigma_{t=1}^{T}\Sigma_{i=1}^{n} g_i^t c_i + \Sigma_{t=1}^{T} z_t & \\ \text{subject to} & \Sigma_{i=1}^{n} g_i^t + b^t = d_t - f_t & t = 1, ..., T \\ & -R_i^{down} \le g_i^t - g_i^{t-1} \le R_i^{up} & \forall i, t = 2, ..., T \\ & -R_i^{down} \le g_i^1 - g_i^T \le R_i^{up} & \forall i \\ & z_t \ge C_1^{def} f_t & \forall t \\ & z_t \ge C_2^{def} f_t + 0.05 d_t (C_2^{def} - C_1^{def}) & \forall t \\ & 0 \le g_i^t \le G_i & \forall i, t \\ & 0 \le f^t \le d_t & \forall t \\ & 0 \le B^t \le G^{bat} & \forall t \\ & -G^{bat} \le b^t \le G^{bat} & \forall t \\ & B^t - B^{t-1} = -b^t & t = 2, ..., T \\ & B^1 - B^T = -b^1 & \end{array} \tag{8}$$

3.4 d

Podemos resolver o problema (c) usando a capacidade da bateria como 0, isso nos dará o custo de operação não tendo bateria. Podemos resolver usando a capacidade da bateria fornecida, isso dará o custo de operação com bateria. A redução do custo usando bateria, multiplicada por N deve ser superior à X, dessa forma existe uma vantagem econômica no uso da bateria.

3.5 e

```julia
function questao_3(use_bat::Bool)
    n = 10
    C_def_1 = 50
    C_def_2 = 100
    G_bat = 8*use_bat
    T = 24
    d = 60*[1 + sin(t/12) for t in 1:T]
    c = [2*i for i in 1:n]

    model = Model(GLPK.Optimizer)
    @variable(model, 0.0 <= B[t = 1:T] <= G_bat)
    @variable(model, -G_bat <= b[t = 1:T] <= G_bat)
    @variable(model, 0.0 <= g[i = 1:n,t = 1:T] <= 22 - 2*i )
    @variable(model, 0.0 <= f[t = 1:T] <= d[t] )
    @variable(model, z[t = 1:T])

    @constraint(model,[t = 1:T], z[t] >= C_def_1 * f[t] )
    @constraint(model,[t = 1:T], z[t] >= C_def_2 * f[t]  + 0.05*d[t]*(C_def_2 - C_def_1))
    @constraint(model, [t = 2:T], B[t] - B[t-1] == - b[t])
    @constraint(model, B[1] - B[T] == - b[1])
    @constraint(model, [i = 1:n, t = 2:T], -i <= g[i,t] - g[i,t-1] <= i)
    @constraint(model, [i = 1:n], -i <= g[i,1] - g[i,24] <= i)
    @constraint(model, [t = 1:T], sum(g[i,t] for i in 1:n) + b[t] == d[t] - f[t])

    @objective(model, Min, sum(c[i]*g[i,t] for i = 1:n, t = 1:T) + sum(z))
    optimize!(model)
    return objective_value(model)
end
```

Usando o codigo acima, foi encontrado 33290.9 como custo sem o uso de bateria e 32124.5 como o custo com bateria. Sendo assim, em um dia a redução de custo já é o suficiente para pagar o custo da bateria, então $N = 1$.

4 Questão 4

Nesta questão L_c^{ir} é o lucro após o imposto de renda no cenário c, L_c é o lucro antes o imposto de renda no cenário c, δ_{ck} é o uso do intervalo k pelo lucro antes do imposto no cenário c, α é o coeficiente da combinação convexa, z é o menor lucro obtido, P_t^{ir} é o lucro após o imposto de renda considerando a taxa do tempo t, δ_{tk}^p é o uso do intervalo k pelo lucro antes do imposto considerando a taxa do tempo t e P_t é o lucro antes do imposto de renda considerando a taxa do tempo t. $L_0 = 0$.

4.0.1 (a)

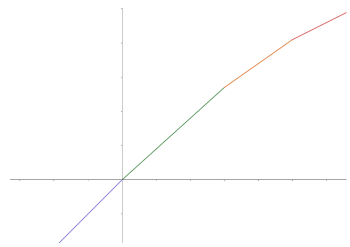

Figure 1: Exemplo de curva de transferência linear para o IR.

4.0.2 (b)

$$\begin{array}{ll} \text{maximize} & \Sigma_{c=1}^N L_c^{ir} p_c \\ \text{subject to} & x^{LR} + x^{CR} = 10000 \\ & L_c^{ir} = \Sigma_{k=1}^K \delta_{ck}(1 - \dfrac{IR_k}{100}) \qquad \forall c \\ & L_c = \Sigma_{k=1}^K \delta_{ck} \qquad \forall c \\ & L_c = x^{LR}(1 + r^{LR}) + x^{CR}(1 + r_c^{CR}) \forall c \\ & 0 \leq \delta_{ck} \leq L_k - L_{k-1} \qquad \forall k \\ & x^{LR}, x^{CR} \geq 0 \end{array} \tag{9}$$

4.0.3 (c)

$$\begin{array}{ll} \text{maximize} & \alpha \Sigma_{c=1}^N L_c^{ir} p_c + (1-\alpha) z \\ \text{subject to} & x^{LR} + x^{CR} = 10000 \\ & L_c^{ir} = \Sigma_{k=1}^K \delta_{ck}(1 - \dfrac{IR_k}{100}) \qquad \forall c \\ & L_c = \Sigma_{k=1}^K \delta_{ck} \qquad \forall c \\ & L_c = x^{LR}(1 + r^{LR}) + x^{CR}(1 + r_c^{CR}) \forall c \\ & 0 \leq \delta_{ck} \leq L_k - L_{k-1} \qquad \forall k \\ & z \leq L_c \qquad \forall c \\ & x^{LR}, x^{CR} \geq 0 \end{array} \tag{10}$$

4.0.4 (d)

Ambos os problemas são programação linear, logo as funções objetivo são lineares e os conjuntos viáveis são poliedros. Como funções lineares são côncavas e poliedros são convexos, esses problemas de maximização são convexos.

4.0.5 (e)

$$
\begin{array}{lll}
\text{maximize} & \alpha\Sigma_{c=1}^{N} L_c^{ir} p_c + (1-\alpha) z & \\
\text{subject to} & x^{LR} + x^{CR} = 10000 & \\
& L_c^{ir} = \Sigma_{k=1}^{K} \delta_{ck}(1 - \dfrac{IR_k}{100}) & \forall c \\
& L_c = \Sigma_{k=1}^{K} \delta_{ck} & \forall c \\
& L_c = x^{LR}(1+r^{LR}) + x^{CR}(1+r_c^{CR}) & \forall c \\
& 0 \le \delta_{ck} \le L_k - L_{k-1} & \forall k \\
& z \le L_c & \forall c \\
& x^{LR}, x^{CR} \ge 0 & \\
& P_t^{ir} = \Sigma_{k=1}^{K} \delta_{tk}^{p}(1 - \dfrac{IR_k}{100}) & \forall t \\
& P_t = \Sigma_{k=1}^{K} \delta_{tk}^{p} & \forall t \\
& P_t = x^{LR}(1+r^{LR}) + x^{CR}(1+r_t^{CR}) & \forall t \\
& 0 \le \delta_{tk}^{p} \le L_k - L_{k-1} & \forall k \\
& P_t^{ir} \ge 9000 & \forall t
\end{array}
\tag{11}
$$

4.0.6 (f)

```
function questao_4()

    r_LR = 0.05 + 1
    N = 4
    K = 2
    T = 24
    r = [0.4, -0.1, 0.2, -0.05] .+ 1
    p = [0.1, 0.2, 0.3, 0.4]
    IR_1 = 0.15
    IR_2 = 0.27
    L_1 = 3500
    L_2 = Inf
    r_t = [0.1*sin(t/6) for t in 1:T]

    model = Model(GLPK.Optimizer)

    @variable(model, x_LR >= 0)
    @variable(model, x_CR >= 0)
    @variable(model, 0 <= d_c1[1:N] <= L_1)
    @variable(model, 0 <= d_c2[1:N])
    @variable(model, 0 <= d_t1[1:T] <= L_1)
```

```
    @variable(model, 0 <= d_t2[1:T])

    @expression(model, L_ir[c=1:N], d_c1[c]*(1-IR_1/100) + d_c2[c]*(1-IR_2/100)  )
    @expression(model, L[c=1:N], d_c1[c] + d_c2[c]  )

    @constraint(model,[c=1:N], L[c] == x_LR*r_LR + x_CR*r[c]  )

    @expression(model, P_ir[t=1:T], d_t1[t]*(1-IR_1/100) + d_t2[t]*(1-IR_2/100)  )
    @expression(model, P[t=1:T], d_t1[t] + d_t2[t]  )

    @constraint(model,[t=1:T], P[t] == x_LR*r_LR + x_CR*r_t[t]  )
    @constraint(model,[t=1:T], P_ir[t] >= 9000  )

    @constraint(model, x_LR + x_CR == 10000)

    @objective(model, Max, sum(L_ir.*p))

    optimize!(model)
    return model
end
```

5 Questão 5

5.0.1 (a)

$$\begin{array}{ll} \text{maximize} & -cx + \Sigma_{i=1}^{N}(pq_i + ru_i) \\ \text{subject to} & 0 \leq x \leq X \\ & 0 \leq q_i \leq x \qquad \forall i \\ & q_i \leq d_i \qquad \forall i \\ & x = q_i + r_i \qquad \forall i \end{array} \tag{12}$$

5.0.2 (b)

$$\begin{array}{ll} \text{maximize} & L \\ \text{subject to} & 0 \leq x \leq X \\ & 0 \leq q_i \leq x \qquad \forall i \\ & q_i \leq d_i \qquad \forall i \\ & x = q_i + r_i \qquad \forall i \\ & L = -cx + \Sigma_{i=1}^{N}(pq_i + ru_i) \\ & \delta \geq -L \\ & \delta \geq 0 \\ & \delta \leq P \end{array} \tag{13}$$

5.0.3 (c)

Para este item, vamos usar a formulação binivel do problema do jornaleiro:

$$\begin{aligned}
\max \quad & -cx + R(x) \\
\text{s.t.} \quad & R(x) = \mathbb{E}[Q(x,d)] \\
& Q(x,d) = \begin{aligned}[t] \max \quad & pq(d) + ru(d) \\ \text{s.t.} \quad & q(d) \leq q \\ & q(d) + u(d) = x \\ & q(d), u(d) \geq 0 \end{aligned}
\end{aligned}$$

Aqui R representa a esperança da receita, Q representa a receita.

Começamos então resolvendo o problema de segundo estágio. A solução é trivial, ele deve vender o maior número de jornais passiveis e devolver o resto, ou seja:

$$q^*(d) = min(d, x)$$

$$u^*(d) = max(x - d, 0)$$

Temos então:

$$R(x) = \mathbb{E}[pmin(d, x) + rmax(x - d, 0)]$$

Pela definição de esperança:

$$R(x) = \int_{-\infty}^{x} (pt + r(x - t))f(t)dt + \int_{x}^{\infty} ptf(t)dt$$

Usando a definição de distribuição acumulada temos:

$$R(x) = (p - r)\int_{-\infty}^{x} tf(t)dt + rxF(x) + px(1 - F(x))$$

Integrando por partes:

$$R(x) = px - (p - r)\int_{-\infty}^{x} F(t)dt$$

Derivando:

$$R'(x) = p - (p - r)F(t)dt$$

Tendo por fim a solução:

$$\frac{p - c}{p - r} < F(0) \Rightarrow x^* = 0$$

$$\frac{p - c}{p - r} > F(X) \Rightarrow x^* = X$$

$$F(0) \leq \frac{p-c}{p-r} \leq F(W) \Rightarrow x^* = F^{-1}(\frac{p-c}{p-r})$$

6 Questão 6

Nesta questão σ e *error* representam o erro de previsão da optimização. y são os *targets*, X é a matrix de dados utilizados para prever, z é o vetor ajustado para fazer a previsão.

6.0.1 (a)

$$\begin{array}{ll} \text{minimize} & \Sigma\sigma \\ \text{subject to} & 0 \leq Xz \leq 1 \\ & \sigma \geq Xz - y \\ & \sigma \geq y - Xz \end{array} \tag{14}$$

6.0.2 (b)

$$\begin{array}{ll} \text{minimize} & \Sigma\sigma + \rho\Sigma w \\ \text{subject to} & 0 \leq Xz \leq 1 \\ & \sigma \geq Xz - y \\ & \sigma \geq y - Xz \\ & w \geq z \\ & w \geq -z \end{array} \tag{15}$$

6.0.3 (c)

```
df = CSV.read("Programação Linear\\Lista 2\\WDBC.dat", DataFrame);
X_train = Array(df[1:400,3:end])
Y_train = Array(df[1:400,2]) .== "B"

X_test = Array(df[401:end,3:end])
Y_test = Array(df[401:end,2]) .== "B"

model = Model(GLPK.Optimizer)
@variable(model, z[i = 1:30])
@variable(model, error[i = 1:400])

@constraint(model, 0 .<= X_train * z .<= 1)
@constraint(model, error .>= X_train * z - Y_train)
@constraint(model, error .>= Y_train - X_train * z)

@objective(model, Min, sum(error))
optimize!(model)

z = value.(z)
true_positive = 0
```

```
true_negative = 0
false_positive = 0
false_negative = 0

for i in 1:length(Y_test)
    if Y_test[i] == 1 && sum(z[j]*X_test[i,j] for j in 1:30) < 0.5
        false_positive += 1
    elseif Y_test[i] == 1 && sum(z[j]*X_test[i,j] for j in 1:30) >= 0.5
        true_negative += 1
    elseif Y_test[i] == 0 && sum(z[j]*X_test[i,j] for j in 1:30) < 0.5
        true_positive += 1
    else
        false_negative += 1
    end
end
```

O resultado obtido foi true_positive = 31, true_negative = 130, false_positive = 0 e false_negative = 8

7 Questão 7

7.1 Parte I

7.1.1 (a)

Uma função f é dita convexa, se e somente se, seu domínio é um conjunto convexo e:

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y) \text{ ,} \forall (x, y, t) \in domain(f) \times domain(f) \times [0, 1]$$

7.1.2 (b)

Um conjunto A é dito convexo, se e somente se:

$$tx + (1-t)y \in A \text{ ,} \forall (x, y, t) \in A \times A \times [0, 1]$$

7.1.3 (c)

A envoltória convexa de um conjunto A pode ser definida de diferentes formas. De um ponto de vista exterior, podemos definir como "A intercessão de todos os conjuntos convexos que contém A". De um ponto de vista interior, podemos definir como "Conjunto de todas combinações convexas de coleções finitas de pontos de A".

7.1.4 (d)

$$epi(f) = \{(x, y) : x \in domain(f), y \in codomain(f), f(x) \leq y\} \subseteq domain(f) \times codomain(f)$$

7.1.5 (e)

Um ponto x de um conjunto convexo A é dito extremo, se e somente se, não pode ser representado como uma combinação convexa **estrita** de dois pontos distintos de A. Ou seja:

$$x = ty + (1-t)z, (y, z, t) \in A \times A \times (0, 1) \rightarrow x = y = z$$

7.1.6 (f)

Vértice é um ponto de um conjunto onde pode ser construído um hiperplano que tem como intercessão no conjunto apenas esse ponto.

7.1.7 (g)

O poliedro é definido como o conjunto solução de um numero finito de igualdades e desigualdades lineares.

7.1.8 (h)

Um simples é a envoltória convexa de um conjunto de pontos *affinely independent*. Um simples unitário é a envoltória convexa do conjunto formato pelo vetor nulo e a base canônica. Um simples de probabilidade é a envoltória convexa do conjunto formato por vetores unitários.

7.1.9 (i)

Raio extremo é um raio que não pode ser escrito como uma combinação convexa de 2 outros raios LI.

7.2 Parte II

7.2.1 (a)

$$f(x^*) = f(\Sigma\alpha_i v_i) = \Sigma\alpha_i f(v_i) \leq \Sigma\alpha_i f(v^*) = f(v^*)\blacksquare$$

Na demostração acima, x^* é o ponto ótimo, v_i são os vértices, v^* é o vértice ótimo. A desigualdade é verdadeira pois α_i é positivo. A última igualdade vem do fato que $\Sigma\alpha_i = 1$

7.2.2 (b)

$$f(x^*) = f(\Sigma\alpha_i v_i) \leq \Sigma\alpha_i f(v_i) \leq \Sigma\alpha_i f(v^*) = f(v^*)\blacksquare$$

Quando é convexa, não temos mais a segunda igualdade. Porém, em funções convexas, a imagem de uma combinação convexa está sempre abaixa da combinação convexa das imagens. Sendo assim, podemos trocar a igualdade por desigualdade, sem mudar o resto da demonstração.

7.2.3 (d)

$$x, y \in \cap C_i \Rightarrow x, y \in C_i \forall i \Rightarrow tx + (1-t)y \in C_i \forall i, t \in [0,1] \Rightarrow tx + (1-t)y \in \cap C_i \in [0,1] \blacksquare$$

Aqui a ideia é que dois pontos que estejam em uma intercessão de uma família enumerável de conjuntos, pertencem a cada um desses conjuntos. Como esses conjuntos são convexos, eles contém as combinações convexas dessas pontos. Como todos eles possuem as combinações, a intercessão também possui.

7.2.4 (e)

Primeiro, vamos mostrar que o epigrafo de uma função f convexa é convexo, começamos pegando 2 pontos de $Epi(f)$:

$$(x_1, y_1), (x_2, y_2) \in Epi(f) \Rightarrow y_1 \geq f(x_1), y_2 \geq f(x_2)$$

Usando t no intervalo $[0, 1]$, fazemos uma combinação convexa dos pontos:

$$t(x_1, y_1) + (1-t)(x_2, y_2) = (tx_1 + (1-t)x_2, ty_1 + (1-t)y_2)$$

Queremos mostrar que esse novo ponto pertence ao $Epi(f)$, ou seja, que a segunda coordenada é maior que a imagem da primeira coordenada. Pela convexidade de f:

$$f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2) \leq ty_1 + (1-t)y_2 \blacksquare$$

Agora queremos mostrar a volta, ou seja, que caso $Epi(f)$ seja convexo, f é convexa. $Epi(f)$ contém todos os pontos (x, y), qual que $y \geq f(x)$, em particular, ele contém os pontos em que $y = f(x)$. Escolhendo então (x_1, y_1) e (x_2, y_2) do gráfico da função, temos que todas suas combinações convexas pertencem ao $Epi(f)$ (pois ele é convexo):

$$f(tx_1 + (1-t)x_2) \leq ty_1 + (1-t)y_2 = tf(x_1) + (1-t)f(x_2) \blacksquare$$

Que é justamente a definição de função convexa.

7.2.5 (c)

O item (d) e (e) foram resolvido primeiro, pois (c) é apenas um corolário. O epigrafo do máximo de funções é a intercessão dos epígrafos. Como funções afins tem epígrafos convexos e a intercessão de conjuntos convexos é um conjunto convexo, o máximo delas é convexo. Como epigrafo é convexo, a função é convexa.

7.2.6 (g)

Em um problema irrestrito, a condição de otimalidade local de primeira ordem diz que $f'(x^*) = 0$, como a função é convexa(min) ou côncava(max), usando esse fato para construir um plano de suporte, temos que $f(x) \geq f(x^*)$, caso seja um problema de minimização ou $f(x) \leq f(x^*)$, caso seja um problema de maximização. De qualquer forma, temos um ótimo global. Esse resultado pode ser estendido para funções não diferenciáveis usando subgradiente.

7.2.7 (h)

Como f é convexa, ela possui subgradientes em todo seu domínio, em particular em $\mathbb{E}[X]$, ou seja, podemos escolher a e b, de tal forma que $ax + b \leq f(x)$ e $a\mathbb{E}[X] + b = f(\mathbb{E}[X])$. Então:

$$\mathbb{E}[f(X)] \geq \mathbb{E}[aX + b] = \mathbb{E}[aX] + \mathbb{E}[b] = a\mathbb{E}[X] + [b] = f(\mathbb{E}[X])\blacksquare$$

8 Questão 8

8.1 Parte I

8.1.1 (a)

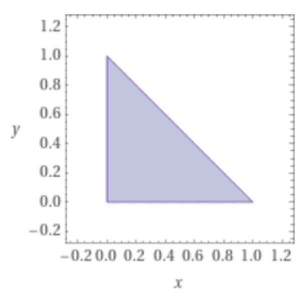

Figure 2: Conjunto C.

8.1.2 (b)

O conjunto C é definido como a intercessão de 3 semi-espaços. Como semi-espaços são convexos, suas intercessões também são. Logo C é convexo.

8.1.3 (c)

Uma inspeção visual permite ver que existe 3 pontos extremos, dados pelas intercessões 2 à 2. Sendo eles (0,0), (1,0) e (0,1).

8.1.4 (d)

C é um simplex unitário, pois possui o vetor nulo e a base canônica como pontos extremos.

8.1.5 (e)

Os elementos de C são da forma $\alpha_1(0.0) + \alpha_2(1.0) + \alpha_3(0.1), \alpha_1 + \alpha_2 + \alpha_3 = 1$

8.2 Parte II

8.2.1 (a)

Como a função é côncava seu hipografo é convexo. Os pontos do tipo (x, y) tal que $y \geq c$ formam um semi-espaço, logo é convexo. A intercessão desses conjuntos é convexa, como já foi mostrado nesta lista. O conjunto S é a projeção de um conjunto convexo em uma de suas coordenadas, logo é convexo.

8.2.2 (b)

Um representação do conjunto foi feita no GeoGebra e está disponível em https://www.geogebra.org/3d/nrnpxawg. A imagem abaixo foi um print feita diretamente do site.

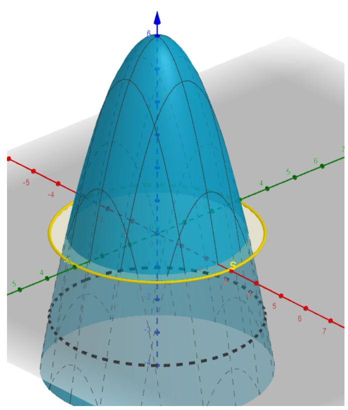

Figure 3: Representação do conjunto S

9 Questão 9

9.0.1 (a)

$$c_i \leq 0 \Rightarrow x_i^* = h_i$$

$$c_i > 0 \Rightarrow x_i^* = b_i$$

$$z^* = c^T x^*$$

Este problema é sempre viável. pode ser ilimitado de b, h ou c forem ilimitados.

9.0.2 (b)

A solução deste problema é a ordem de mérito, ou seja, x_i do menor c_i será $min(d, h_i)$. Após isso, a demanda pode ser atualizada, para $d = d - x_i$. Caso a nova demanda não seja zero, tomamos o menor c_j restando e repetimos o processo.

A inviabilidade ocorre se d for maior que a soma de h.

10 Questão 10

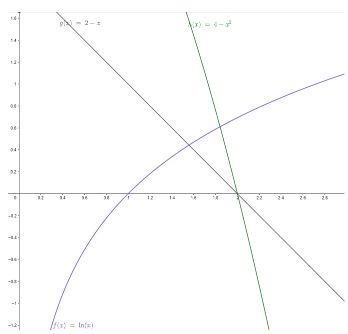

Figure 4: Curvas separadas.

As funções estão definidas em toda a reta, por exceção de $ln(x)$, que esta definida apenas para $x > 0$, logo f está definida para $x > 0$.

Por 7.II.d sabemos que a intercessão de conjuntos convexos é um conjunto convexo. Por 7.II.e sabemos que o epigrafo de uma função é um conjunto convexo se e somente se tal função for convexa.

As função $ln(x)$, $2 - x$ e $4 - x^2$ são côncavas, logo $-ln(x)$, $-2 + x$ e $-4 + x^2$ são convexas. Por 7.II.e os epígrafos de $-ln(x)$, $-2 + x$ e $-4 + x^2$ são convexos, logo os hipografos de $ln(x)$, $2 - x$ e $4 - x^2$ são convexos. O hipografo de f é a intercessão dos hipografos de $ln(x)$, $2 - x$

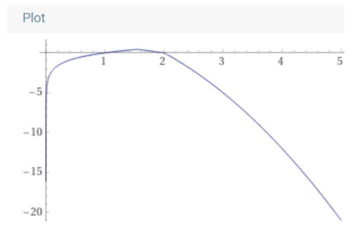

Figure 5: Mínimo entre as curvas.

e $4 - x^2$, por 7.II.d é um conjunto convexo. Logo por 7.II.e epigrafo de $-f$ é convexo, então o hipografo de f é convexo, logo f é côncava.