Pontos Interiores

Pedro Thiago - 1612702

2 de Julho de 2020

Declaração:

Declaro para devidos fins que Eu, Pedro Thiago de Souza M. Marmello fiz todos os exercícios da lista sem auxílio de outros alunos. Apenas consultando os livros, os slides, notas de aula e as aulas gravadas.

Questão 1

Para a criação da função PrimalDual (Pontos Interiores), precisamos primeiro definir o que será passado para a função. A Matriz A e os vetores C e b serão os mesmos do problema de produção; assim como o MaxCount o contador do loop que iremos precisar. Também iremos precisar de duas constantes: a e p, respectivamente, para utilizar junto com o TX e TS, e para o μ inicial.

Começamos com a definição dos vetores X,Y e S e seus respectivos índices (IndX,IndY,IndS); onde X e S = ones(n) e Y= ones(m). Calcula-se o μ e o gap:

$$\mu = p * X' * S/n$$

e o

$$gap = X' * S$$

Inicializamos TX e TS =1 podemos começar o loop. Enquanto gap $>$ 0.000001 ele irá executar. Dentro do While temos a definição de Fk e Jk, onde $E = ones(length(x), 1)$

$$Fk = [A' * y + s - C'; A * x - b; Diagonal(x) * Diagonal(s) * E - E * \mu]$$

$$Jk = \begin{bmatrix} zeros(m,m) & A' & diagm(0 => fill(1.,m)); \\ A & zeros(n,n) & zeros(n,n); \\ Diagonal(s) & zeros(n,n) & Diagonal(x) \end{bmatrix}$$

Fk e Jk servem para encontrarem a solução de dk, tal que $Jk*dk = -Fk$. Desta forma podemos definir dx = dk[IndX],dy= dk[IndY] e ds= dk[IndS]. Depois disso, basta encontrar o mínimo de $1, a * min(-x_k/dx_k)$ e o mínimo de $1, a * min(-s_k/ds_k)$.

Depois de encontrar os mínimos, podemos reestabelecer X,Y e S tal que $X = X + TX * dx \quad Y = Y + TS * dy \quad S = S + TS * ds$**. Podemos agora também reestabelecer o** μ **com o novo X e S. Analisaremos também se** $b - A * x == 0$ **ou** $C^T - A^T * Y - S == 0$**, caso um dos dois sejam, analisamos se** $dx > 0 C * dx < 0$ **ou se** $ds > 0 b * dy > 0$**;em ambos os casos o problema é dado como ilimitado. Ou seja, se em algum momento do loop alguma das hipóteses anteriores for satisfeita, o problema é dado como ilimitado. Depois disso, aumentamos o counter de 1 unidade. E o loop termina quando Gap** $<$ **0.001, retornando o Counter, X, S, Y e os valores das funções objetivo Primal e Dual.**

Questão 2

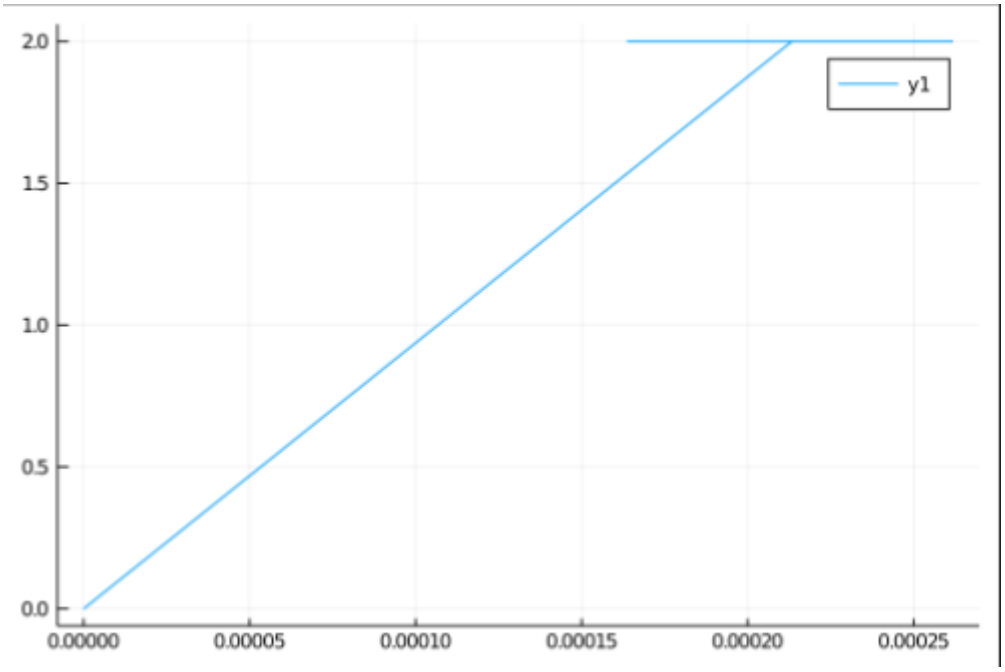
Para fins de simplicidade, para a comparação foram utilizados o Programa de Pontos interiores feito acima, GLPK solver do pacote JuMP e o Simplex criado para a lista anterior na Questão 8 que fora modificado para não precisar da Matriz R e bR.

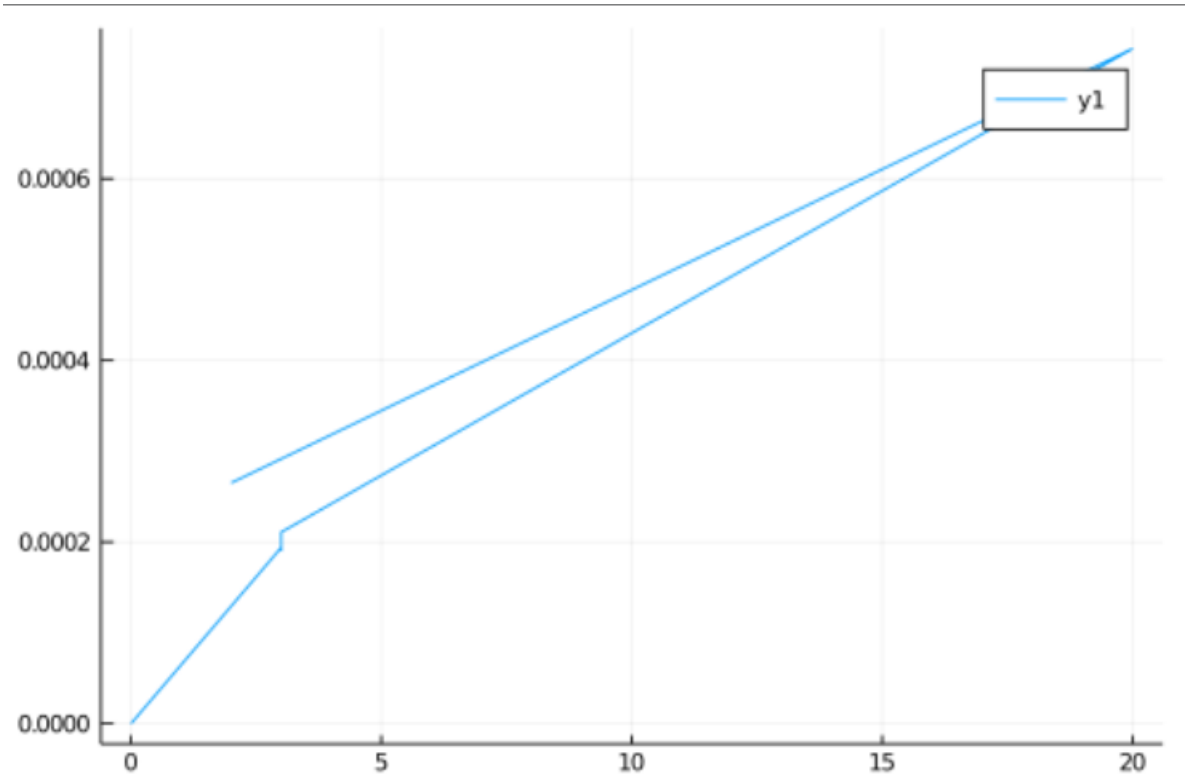
Comparando o número de iterações de Pontos interiores com o GLPK temos a partir do comando ... que o número de iterações realizadas pelo GLPK é ..., enquanto - como foi mostrado na questão anterior - utilizamos um Counter para ver o número de vezes que o loop é realizado para o modelo Primal-Dual; tendo um total de 2. Possuindo o mesmo número de iterações que o Simplex. O que indica que ambos tomam o mesmo nível processamento para encontrar a resposta do problema primal, dado o loop realizado com While e o número de iterações realizadas. Outra coisa a se observar é que ambos, Pontos interiores e GLPK, são capazes de encontrar a solução do problema dual (caso haja um), enquanto o Simplex apenas resolve o primal.

Já com relação ao tempo de demora para processamento, ao utilizar o comando "solve-time(model)"podemos obter o tempo que o GLPK processou o ótimo do modelo: 0.00200 segundos. Para o Simplex e Pontos Interiores, usaremos a função @timed a qual retorna o tempo de processamento de cada uma; no caso da função Pontos Interiores, o tempo resultante foi de 5.0167 segundos, enquanto para o Simplex foi 8.373 segundos. Ao testar a função "@timed"com o GLPK temos que a demora foi de 15.670 segundos. O que nos mostra que para o Problema Linear utilizado, Pontos Interiores realmente é o que consome menos recursos de processamento, enquanto gera as respostas requisitadas em relação ao PL Primal e Dual.

Podemos observar que para o problema de produção o gap do Primal-Dual é na 7ª casa decimal já que nossa tolerância é de 10^{-6}. Poserva-se também que os valores de X ótimo para os 3 programas são iguais, enquanto o resultado da Função objetivo o Primal-Dual distoa na 6ª casa do GLPK e do Simplex. Ambos Simplex e GLPK possuem os mesmos valores para X e Fobj.

Todavia, ao examinar para matrízes maiores com mais iterações fica claro o quanto o p e o a iniciais afetam no desempenho e no número de iterações que a função PrimalDual terá. Quanto mais p estiver longe de 0, maior o número de iterações; assim como, quanto mais próximo de 0 a estiver, maior o número de iterações no counter. Além disso, para matrízes maiores, o programa utilizado no Simplex e no PrimalDual também distoam em relação ao X ótimo do GLPK. Devido problemas no computador utilizado, não foi possível realizar testes com um grande número de iterações e os gráficos (a seguir) terão pequenas iterações.


2.0
1.5
1.0
0.5
0.0
0.00000
0.00005
0.00010
0.00015
0.00020
0.00025
y1


y1
0.0006
0.0004
0.0002
0.0000
0
5
10
15
20