HW#2
Simplex Algorithm

Problem (1)

1. Klee-Minty

- 問題描述 :

在 Klee-Minty 示例上應用不同的pivot, 以顯示您可以解決的最大“n”是多少！ 。

General Case

$$\max \sum_{j=1}^{n} 10^{n-j} x_j$$

$$s.t.\ 2\sum_{j=1}^{i-1} 10^{i-j} x_j + x_i \le 100^{i-1}$$

$$\forall x_j \ge 0,\ 1 \le i \le n$$

Case: n=3

$$\max\ 100x_1 + 10x_2 + x_3$$

$$x_1 \le 1$$

$$s.t.\quad 20x_1 + x_2 \le 100$$

$$200x_1 + 20x_2 + x_3 \le 1000$$

$$x_1, x_2, x_3 \ge 0$$

1. Klee-Minty

1. 套件 scipy
   - 使用 scipy.optimize.**linprog**(method='**revised simplex**', options={'**pivot**' : '**mrc**'/'**bland**'}) 求解。
2. 說明並比較兩個不同的pivot結果(mrc、bland)。

參考資料 : linprog(method='revised simplex') — SciPy v1.9.3 Manual

1. Klee-Minty

status : *int*

An integer representing the exit status of the algorithm.

0 : Optimization terminated successfully.

1 : Iteration limit reached.

2 : Problem appears to be infeasible.

3 : Problem appears to be unbounded.

4 : Numerical difficulties encountered.

5 : Problem has no constraints; turn presolve on.

6 : Invalid guess provided.

3. 作法要求:

- n從1開始,每次加1, 直到最後求解的status不等於0, 則停止(即求出最大可解的n)。

如果status=1,表示達到linprog()限制的最大迭代次數, 可去調整其參數maxiter來更改最大迭代次數。

- 可自動生成在不同n所對應的objective function跟constraint的係數陣列(會做為求解時要使用的函數linprog()的輸入)。

Problem (2)

2. An example problem

- 問題描述 :

  假設您負責高中午餐的飲食計劃,

  你的工作是minimize所要花費的錢,

  使學生從所選食物中獲得的營養平衡(每日最小攝取量)。

- 資料集 : diet.xls (會上傳到ecourse2)

2. An example problem – Dataset

Foods	Price/Serving	Serving Size	Calories	Cholesterol (mg)	Total_Fat (g)	Sodium (mg)	Carbohydrates (g)	Dietary_Fiber (g)	Protein (g)	Vit_A (IU)	Vit_C (IU)	Calcium (mg)	Iron (mg)
Frozen Broccoli	$0.48	10 Oz Pkg	73.8	0	0.8	68.2	13.6	8.5	8	5867.4	160.2	159	2.3
Frozen Corn	$0.54	1/2 Cup	72.2	0	0.6	2.5	17.1	2	2.5	106.6	5.2	3.3	0.3
Raw Lettuce Iceberg	$0.06	1 Leaf	2.6	0	0	1.8	0.4	0.3	0.2	66	0.8	3.8	0.1
Baked Potatoes	$0.18	1/2 Cup	171.5	0	0.2	15.2	39.9	3.2	3.7	0	15.6	22.7	4.3
Tofu	$0.93	1/4 block	88.2	0	5.5	8.1	2.2	1.4	9.4	98.6	0.1	121.8	6.2
Roasted Chicken	$2.52	1 lb chicken	277.4	129.9	10.8	125.6	0	0	42.2	77.4	0	21.9	1.8
Spaghetti W/ Sauce	$2.34	1 1/2 Cup	358.2	0	12.3	1237.1	58.3	11.6	8.2	3055.2	27.9	80.2	2.3
Raw Apple	$0.72	1 Fruit,3/Lb,Wo/Rf	81.4	0	0.5	0	21	3.7	0.3	73.1	7.9	9.7	0.2
Banana	$0.45	1 Fruit,Wo/Skn&Seeds	104.9	0	0.5	1.1	26.7	2.7	1.2	92.3	10.4	6.8	0.4
Wheat Bread	$0.15	1 Sl	65	0	1	134.5	12.4	1.3	2.2	0	0	10.8	0.7
White Bread	$0.18	1 Sl	65	0	1	132.5	11.8	1.1	2.3	0	0	26.2	0.8
Oatmeal Cookies	$0.27	1 Cookie	81	0	3.3	68.9	12.4	0.6	1.1	2.9	0.1	6.7	0.5
Apple Pie	$0.48	1 Oz	67.2	0	3.1	75.4	9.6	0.5	0.5	35.2	0.9	3.1	0.1
Scrambled Eggs	$0.33	1 Egg	99.6	211.2	7.3	168	1.3	0	6.7	409.2	0.1	42.6	0.7
Turkey Bologna	$0.45	1 Oz	56.4	28.1	4.3	248.9	0.3	0	3.9	0	0	23.8	0.4
Beef Frankfurter	$0.81	1 Frankfurter	141.8	27.4	12.8	461.7	0.8	0	5.4	0	10.8	9	0.6
Chocolate Chip Cookies	$0.09	1 Cookie	78.1	5.1	4.5	57.8	9.3	0	0.9	101.8	0	6.2	0.4
		Minimum daily intake	800	30	20	800	130	60	100	1000	400	700	10

2. An example problem

1. 作法要求:

先將最小化問題轉化為它的對偶問題, 在使用套件求解。

Primal:

$$\max z = c_1x_1 + c_2x_2 + \dots c_nx_n$$

$$\text{s.t. } a_{11}x_1 + a_{12}x_2 + \dots a_{1n}x_n \le b_1$$
$$a_{21}x_1 + a_{22}x_2 + \dots a_{2n}x_n \le b_2$$
$$\vdots \quad \vdots \quad \vdots$$
$$a_{m1}x_1 + a_{m2}x_2 + \dots a_{mn}x_n \le b_m$$

$$x_i \ge 0 \quad (i=1,2,\dots,n)$$

Dual:

$$\min w = \pi_1b_1 + \pi_2b_2 + \dots \pi_mb_m$$

$$\text{s.t. } \pi_1a_{11} + \pi_2a_{21} + \dots \pi_ma_{m1} \ge c_1$$
$$\pi_1a_{12} + \pi_2a_{22} + \dots \pi_ma_{m2} \ge c_2$$
$$\vdots \quad \vdots \quad \vdots$$
$$\pi_1a_{1n} + \pi_2a_{2n} + \dots \pi_ma_{mn} \ge c_n$$

$$\pi_j \ge 0 \quad (i=1,2,\dots,m)$$

2. An example problem

2. 套件 scipy
   - 使用 scipy.optimize.linprog(method='**highs-ds**', options={'**simplex_dual_edge_weight_strategy**' : ' dantzig'/' devex'/' steepest '}) 求解。
3. 說明並比較三個不同的simplex_dual_edge_weight_strategy (dantzig、devex、steepest)。

參考資料 : linprog(method='highs-ds') — SciPy v1.9.3 Manual

繳交檔案

1. 程式碼 (.ipynb)
2. .pdf檔

有問題可寄mail詢問：ytc1006@alum.ccu.edu.tw