**Analysis of Algorithms**

**Assignment 3**

**Ryan Farber, Zach Singleton, Huarui Wu**

Problem 1: mmmm . . . pork

**Mathematical Model**

|  |
| --- |
| Objective Function |
| max(8 \* hamf + 12 \* hamr + 11 \* hamo+ 4 \* belliesf + 12 \* belliesr  + 7 \* bellieso+ 4 \* picnicsf + 13 \* picnicsr + 9 \* picnicso) |

Where **f** = fresh, **r** = smoked in regular time and **o** = smoked in overtime

|  |
| --- |
| Constraints |
| hamf + hamr + hamo 480  belliesf + belliesr + bellieso ≤ 400  picnicsf + picnicsr + picnicso ≤ 230  hamr + belliesr + picnicsr ≤ 420  hamo + bellieso + picnicso ≤ 250 |

**Standard Form**

|  |
| --- |
| Objective Function |
| max(8 \* hamf + 12 \* hamr + 11 \* hamo+  4 \* belliesf + 12 \* belliesr + 7 \* bellieso+  4 \* picnicsf + 13 \* picnicsr + 9 \* picnicso) |

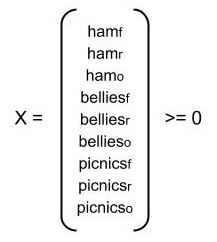
|  |
| --- |
| Constraints |
| hamf + hamr + hamo + hamremain = 480  belliesf + belliesr + bellieso + belliesremain = 400  picnicsf + picnicsr + picnicso + picnicsremain = 230  hamr + belliesr + picnicsr + smokereg = 420  hamo + bellieso + picnicso + smokeover = 250  hamremain; belliesremain; picnicsremain; smokereg; smokeover 0 |

**Matrix Form**

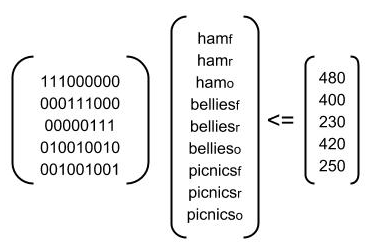
We wish to find max(cx), where

c =( 8 14 11 4 12 7 4 13 9)

and



Also,



Solution

Net Profit: $10,910

|  |  |  |  |
| --- | --- | --- | --- |
|  | fresh | smoked on regular time | smoked on overtime |
| hams | 440 | 0 | 40 |
| bellies | 0 | 400 | 0 |
| picnics | 0 | 20 | 210 |

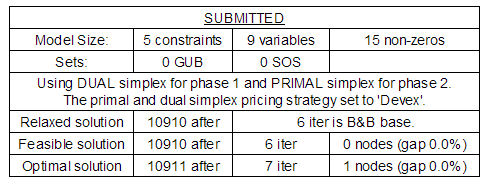
Using an LP-solver

We used the GNU Linear Programming Kit to solve this problem. The help page tells us that we shouldgive a model \*le (provided in the next section) to glpsol, which we do by running: glpsol -m pork.mod -o pork.sol.

**Problem 1 Code and Results**

|  |
| --- |
| /\* Objective function \*/ |
| max: 8 h\_f + 12 h\_r + 11 h\_o +  4 b\_f + 12 b\_r + 7 b\_o +  4 p\_f + 13 p\_r + 9 p\_o; |
| /\* Variable bounds \*/ |
| C1: h\_f + h\_r + h\_o <= 480;  C2: b\_f + b\_r + b\_o <= 400;  C3: p\_f + p\_r + p\_o <= 230;  C4: h\_r + b\_r + p\_r <= 420;  C5: h\_o + b\_o + p\_o <= 250; |
| /\* Integer definitions \*/ |
| int h\_f, h\_r, h\_o,  b\_f, b\_r, b\_o,  p\_f, p\_r, p\_o; |

## result:



Thus,

Excellent numeric accuracy ||\*|| = 5.68434e-014

|  |  |
| --- | --- |
| VARIABLES | RESULT |
| h\_f | 440 |
| h\_r | 0 |
| h\_o | 40 |
| b\_f | 0 |
| b\_r | 400 |
| b\_o | 0 |
| p\_f | 0 |
| p\_r | 20 |
| p\_o | 210 |

Problem 2: least squares isn’t good enough for me

min(t)

Given a set of points (x1, y1), (x2, y2) ... (xn; yn),

|xi - b| t, and

|axi + byi =c| t,

for 1 i n.

Given a set of points (x1, y1), (x2, y2)... (xn, yn),

xi - b + vi = t

xi - b -t

axi + byi - c + zi = t

axi + byi - c -t

vi; zi 0

for 1 i n.

**SOLUTION**

a = -10.33333

b = 5.5

c = 4.33333

This best fit line is shown in Figure 1.

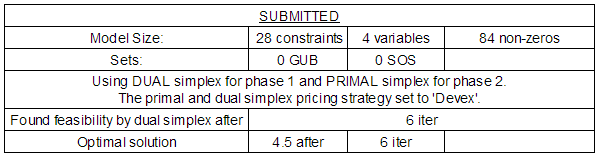
# 

# Problem 2 code and results

## code:

|  |
| --- |
| /\* Objective function \*/ |
| min: t; |
| /\* Variable bounds \*/  // x values |
| C1: 1 - b <= t;  C2: 1 - b >= -t;  C3: 2 - b <= t;  C4: 2 - b >= -t;  C5: 3 - b <= t;  C6: 3 - b >= -t;  C7: 5 - b <= t;  C8: 5 - b >= -t;  C9: 7 - b <= t;  C10: 7 - b >= -t;  C11: 8 - b <= t;  C12: 8 - b >= -t;  C13: 10 - b <= t;  C14: 10 - b >= -t; |
| // abs deviation |
| C8: 1a + 3b - c <= t;  C9: 1a + 3b - c >= -t;  C10: 2a + 5b - c <= t;  C11: 2a + 5b - c >= -t;  C12: 3a + 7b - c <= t;  C13: 3a + 7b - c >= -t;  C14: 5a + 11b - c <= t;  C15: 5a + 11b - c >= -t;  C16: 7a + 14b - c <= t;  C17: 7a + 14b - c >= -t;  C18: 8a + 15b - c <= t;  C19: 8a + 15b - c >= -t;  C20: 10a + 19b - c <= t;  C21: 10a + 19b - c >= -t; |
| /\* Integer definitions \*/ |
| free a, b, c, t; |

## result:



Thus,

Excellent numeric accuracy ||\*|| = 0

|  |  |
| --- | --- |
| VARIABLES | RESULT |
| a | -10.33333 |
| b | 5.5 |
| c | 4.333333 |
| t | 4.5 |

