AAE 526: More Linear Programming

Thomas F. Rutherford

Fall Semester, 2019
Department of Agricultural and Applied Economics
University of Wisconsin, Madison

September 9, 2019

Outline

- Another LP example
- Linear algebra and indexed GAMS code
- Linear programs in standard form
- Supersize me!
- Visualization

Another Tiny LP

The Wyndor Glass Co. produces high-quality glass products, including windows and glass doors. It has three plants. Aluminum frames and hardware are made in Plant 1, wood frames are made in Plant 2 and Plant 3 cuts the glass and assembles the products.
Profits are 3 for product 1 and 5 per unit of product 2.
Because of declining earnings, top management has decided to revamp the product line. Unprofitable products are to be discontinued, and product capacity will be reassigned to launch two new products.

Describe Constraints

The data indicates that each batch of product 1 produced per week uses 1 hour of production time per week in Plant 1, whereas only 4 hours per week are available. The restriction is written mathematically as

$$x_1 \leq 4$$

Similarly, Plant 2 imposes the restriction

$$2x_2 \leq 12$$

The number of hours of production time used per week in Plant 3 given x_1 and x_2 is $3x_1 + 2x_2$ (3 hours of production per batch of good 1 and 2 hours per batch of good 2). The available capacity in Plant 3 provides the constraint:

$$3x_1 + 2x_2 \leq 18$$

Finally, production rates cannot be negative, so it is necessary to impose restrictions $x_1 \geq 0$ and $x_2 \geq 0$

Write Down an Algebraic Formulation

$$\max Z = 3x_1 + 5x_2$$

subject to the restrictions

$$\begin{array}{rrcr} x_1 & & \leq & 4 \\ & 2x_2 & \leq & 12 \\ 3x_1 & +2x_2 & \leq & 18 \end{array}$$

and

$$x_1 \geq 0, x_2 \geq 0$$

gamside: D:\newproject.gpr - [d:\Optimization\lectures\Lecture2\wyndor.gms]

File Edit Search Windows Utilities Model Libraries Help

wyndor.gms

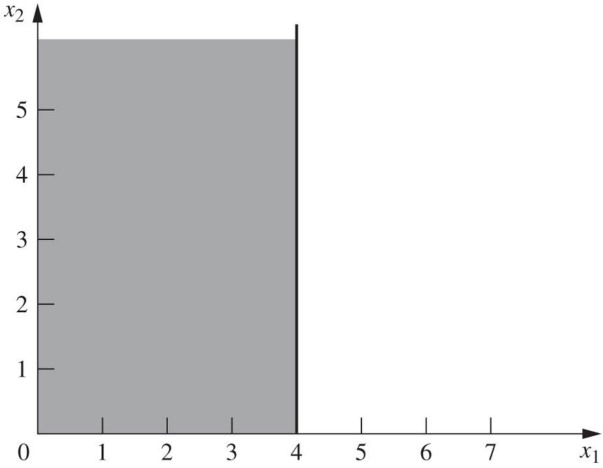
```
VARIABLES
        Z        Objective function ($1000 per week)
        X1       Glass doors (batches per week)
        X2       Wood framed windows (batches per week);

EQUATIONS
        Zdef     Defines the objective function
        plant1   Constraint imposed by plant 1
        plant2   Constraint imposed by plant 2
        plant3   Constraint imposed by plant 3;

zdef..            Z =E= 3*X1 + 5*X2;
plant1..          X1 =L= 4;
plant2..          2*X2 =L= 12;
plant3..          3*X1 + 2*X2 =L= 18;

MODEL   wyndor /zdef, plant1, plant2, plant3/;

SOLVE wyndor USING LP MAXIMIZING Z;
```

Graphical Solution: for Two Variable Problems

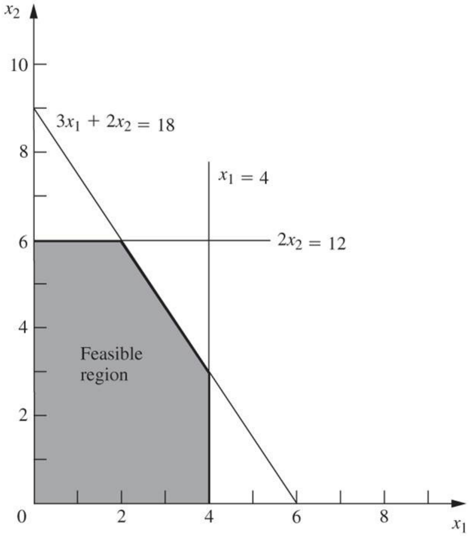
This problem has only two dimensions, so a graphical procedure can be employed. We use label the axes as x_1 and x_2. The first step is then to identify on the graph values of (x_1, x_2) which are *feasible* (consistent with the restrictions).

Upper and Lower Bounds

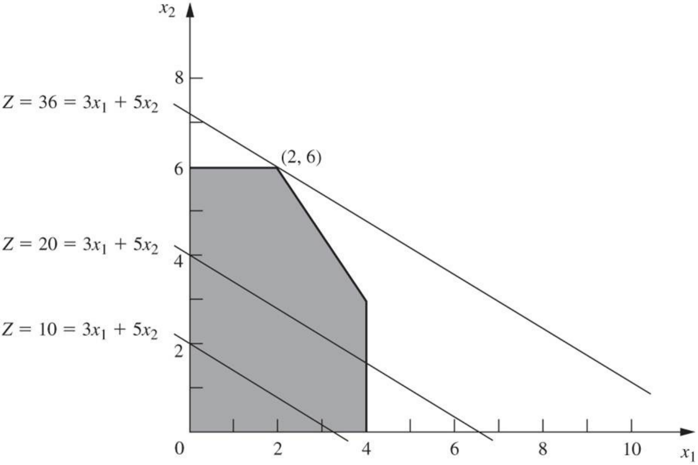
Values of (x_1, x_2) consistent with the constraints $0 \le x_1 \le 4$ and $0 \le x_2$:



The Feasible Region



The optimal solution is $x_1^* = 2, x_2^* = 6$ with $Z^* = 36$. This implies 2 batches of product 1 and 6 batches of product 2 will be produced per week, providing a total profit of \$36,000 per week.



Outline

- Another LP example
- Linear algebra and indexed GAMS code
- Linear programs in standard form
- Supersize me!
- Visualization

Most Practical GAMS Models are Indexed

```
$TITLE  Brewery Profit Maximization

set     j       Products /lager, ale/

        i       Ingredients /
                        malt    Malt,
                        yeast   Yeast,
                        dehops  German hops,
                        wihops  Wisconsin hops/;

parameter
        p(j)    Profit by product /lager 12, ale 9/
        s(i)    Supply by ingredient /malt 4800, yeast 1750,
                                        dehops 1000, wihops 1750/;

table           a(i,j)  Requirements
                        lager   ale
        malt            4       2
        yeast           1       1
        dehops          1       0
        wihops          0       1;
```

GAMS Models are Indexed (cont.)

```
variables        Y(j)    Production levels,
                 Z       Profit (maximand);

nonnegative variable   Y;

equations        supply(i)        Ingredient supply
                 profit           Defines Z;

supply(i)..      sum(j, a(i,j)*Y(j)) =L= s(i);

profit..         Z =E= sum(j, p(j)*Y(j));

MODEL   BREWERY /supply, profit/;

solve BREWERY using LP maximizing Z;
```

Matrix basics

A matrix is an array of numbers. $A \in \mathrm{R}^{m \times n}$.

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$$

which has m rows and n columns.

The `table` statement in GAMS can be used to define a matrix:

```
set      i        Row indices  /1*3/,
         j        Column indice /1*2/;

table a(i,j)      Matrix with three rows and two columns
                  1        2
         1        0.23     12.3
         2        -0.1     2.4
         3        3.2      0.1 ;
```

Table versus Parameter

A matrix may be specified either the `table` or `parameter` statement:

```
set     i       Row indices  /1*3/,
        j       Column indice /a,b/;

table a(i,j)    Matrix with three rows and two columns
                a       b
        1       0.23    12.3
        2       -0.1    2.4
        3       3.2     0.1 ;

parameter b(i,j)    The same matrix in database format /
        1.a     0.23
        2.a     -0.1
        3.a     3.2
        1.b     12.3
        2.b     2.4
        3.b     0.1 /;

parameter c(i,j)  Check that a=b;  c(i,j) = a(i,j) - b(i,j); display c;

----     22 PARAMETER c  check that a=b
                    ( ALL       0.000 )
```

Matrix Multiplication

Two matrices can be multiplied *if their inner dimensions agree*. In matrix notation (MATLAB style):

$$\underbrace{C}_{(m \times p)} = \underbrace{A}_{(m \times n)} \underbrace{B}_{(n \times p)}$$

In detached coefficient notation (GAMS style) we write:

$$c_{ij} = \sum_k a_{ik} b_{kj}$$

In GAMS syntax, we have:

```
c(i,j) = sum(k, a(i,k) * b(k,j));
```

Transposition

- The transpose operator A^T swaps rows and columns. If $A \in \mathrm{R}^{m \times n}$, then $A^T \in \mathrm{R}^{n \times m}$ and $(A^T)_{ij} = A_{ji}$
- It follows that:

$$(A^T)^T = A$$
$$(AB)^T = B^T A^T$$

Linear and affine functions

- A function $f(x_1, \ldots, x_m)$ is *linear* in the variables $x_1, \ldots, x_m$ if there exists constants $a_1, \ldots, a_m$ such that

$$f(x_1, \ldots, x_m) = a_1 x_1 + \ldots + a_m x_m = \sum_i a_i x_i = a^T x$$

- A function $f(x_1, \ldots, x_m)$ is *affine* in the variables $x_1, \ldots, x_m$ if there exists constants $b, a_1, \ldots, a_m$ such that

$$f(x_1, \ldots, x_m) = b a_1 x_1 + \ldots + a_m x_m = b + \sum_i a_i x_i = b + a^T x$$

- Examples:
  1. $3x - y$ is *linear* in (x, y).
  2. $2xy + 1$ is *affine* in x and y, but not in (x, y).
  3. $x^2 + y^2$ is neither linear nor affine.

Linear and affine functions (cont.)

Several linear or affine functions can be combined:

$$\begin{array}{llll} a_{11}x_1 & +\dots & +a_{1n}x_n + b_1 \\ a_{21}x_1 & +\dots & +a_{2n}x_n + b_2 \\ & \vdots & \vdots \\ a_{m1}x_1 & +\dots & +a_{mn}x_n + b_m \end{array} \Rightarrow \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$$

which can be written simply as $Ax + b$. Same definitions apply:

- A vector-valued function $F(x)$ is linear in x if there exists a constant matrix A such that $F(x) = Ax$.
- A vector-valued function $F(x)$ is affine in x if there exists a constant matrix A and vector b such that $F(x) = Ax + b$.

Matrix basics: inner and outer products

A vector is a column matrix. We write $x \in \mathrm{R}^n$ to mean that

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$

This is an $n \times 1$ matrix.

Two vectors $x, y \in \mathrm{R}^n$ can be multiplied together in two ways. Both are valid matrix multiplications:

- **inner product**: produces a scalar, $x^T y = x_1 y_1 + \cdots + x_n y_n$.
- **outer product**: produces an $n \times n$ matrix.

$$xy^T = \begin{bmatrix} x_1 y_1 & \cdots & x_1 y_n \\ \vdots & \ddots & \vdots \\ x_n y_1 & \cdots & x_n y_n \end{bmatrix}$$

Calculating Inner and Outer Products in GAMS

```
set     i/i1*i3/;

parameter       x(i), y(i), xy, xyt(i,i);

*       Generate two random arrays containing values
*       between zero and one:

x(i) = uniform(0,1); y(i) = uniform(0,1);

*       Inner product:

xy = sum(i, x(i)*y(i));


*       Need a second symbol to refer to the set i:

alias (i,j);

*       Outer product:

xyt(i,j) = x(i)*y(j);

display x, y,  xy, xyt;
```

Calculating Inner and Outer Products in GAMS

```
----      23 PARAMETER x

i1 0.172,    i2 0.843,    i3 0.550


----      23 PARAMETER y

i1 0.301,    i2 0.292,    i3 0.224


----      23 PARAMETER xy                   =        0.421

----      23 PARAMETER xyt

            i1          i2          i3

i1       0.052       0.050       0.038
i2       0.254       0.246       0.189
i3       0.166       0.161       0.123
```

Outline

- Another LP example
- Linear algebra and indexed GAMS code
- Linear programs in standard form
- Supersize me!
- Visualization

The linear program

A linear program is an optimization model with:

- real-valued variables ($x \in \mathrm{R}^n$)
- linear cost function ($c^T x$)
- constraints may be:
  - affine equations ($Ax = b$)
  - affine inequalities ($Ax \leq b$ or $Ax \geq b$)
  - combinations of the above
- individual variables may have:
  - bounds ($p \leq x_i$, or $x_i \leq q$, or $p \leq x_i \leq q$)
  - no bounds (x_i is unconstrained)

There are many equivalent representations of any linear program.

Linear Programming

Standard Form:

$$
\begin{array}{rrl}
\text{maximize} & c_1x_1 + c_2x_2 + \cdots + c_nx_n & \\
\text{subject to} & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & \leq b_1 \\
 & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & \leq b_2 \\
 & \vdots & \\
 & a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n & \leq b_m \\
 & x_1, x_2, \ldots, x_n & \geq 0
\end{array}
$$

Why it's hard:

- Lots of variables (n of them)
- Lots of boundaries to check (the inequalities)

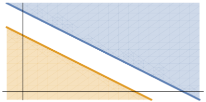
Why it's not impossible:

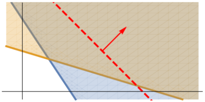
- All expressions are linear

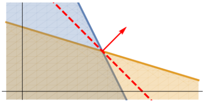
Solutions of a Linear Program

For any given linear programming problem, exactly one of the following statements applies:

1. *The model is infeasible*: there is no x that satisfies all the constraints. (is the model correct?)
2. *The model is feasible, but unbounded*: the cost function can be arbitrarily improved. (forgot a constraint?)
3. *Model has a solution which occurs on the boundary of the feasible polyhdron.* Note that there is no guarantee that the solution is unique – there may be many solutions!


infeasible


unbounded


boundary

Standard Form

- Every linear program can be put into the form:

$$\max_{z \in \mathrm{R}^n} c^T z$$

subject to:

$$Az \leq b$$

$$z \geq 0$$

- This is call the *standard form* of a linear program.

Brewery Profit: Standard Form

$$\begin{array}{llcl} \max_{x,y} & 120x + 90y & & \\ \text{s.t.:} & 4x + 2y & \leq & 4800 \\ & x + y & \leq & 1750 \end{array}$$

$$0 \leq x \leq 1000, \quad 0 \leq y \leq 1500$$

is equivalent to:

$$\max_{x,y} \begin{bmatrix} 120 \\ 90 \end{bmatrix}^T \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\text{s.t.} \quad \begin{bmatrix} 4 & 2 \\ 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \leq \begin{bmatrix} 4800 \\ 1750 \\ 1000 \\ 1500 \end{bmatrix}$$

$$x, y \geq 0$$

Hence, our brewery profit maximization model can be transformed into standard inequality form with the assignments:

$$z = \begin{bmatrix} x \\ y \end{bmatrix} \qquad c = \begin{bmatrix} 120 \\ 90 \end{bmatrix}$$

$$A = \begin{bmatrix} 4 & 2 \\ 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad b = \begin{bmatrix} 4800 \\ 1750 \\ 1000 \\ 1500 \end{bmatrix}$$

Applications of Linear Programming

A partial list, taken from Ferris et al., Chapter 1:

- Resoure allocation
- The diet problem
- Linear surface fitting
- Load balancing
- Classification
- Minimum cost network flow
- ...

Resource Allocation

A company has m products which are produced with n resources. The value of product i is c_i, while each unit of resource j costs d_j dollars. One unit of product i requires a_{ij} units of resource j, and a maximum of b_j units of resource j are available:

$$\max_{x,y} z = \sum_i c_i y_i - \sum_j d_j x_j$$

subject to

$$x_j = \sum_i a_{ij} y_i, \quad x_j \leq b_j, \quad x_j \geq 0, y_i \geq 0$$

Note that the constraints can be written in *detached coefficient form* as:

$$x_j = \sum_i a_{ij} y_i = a_{1j} y_1 + a_{2j} y_2 + \ldots + a_{mj} y_m$$

The Diet Problem

Given the prices p_j of food type j, the content of nutrient i in food j (a_{ij}) and the dietary requirement of nutrient i, b_i, solve:

$$\min_{x} z = \sum_{j} p_j x_j$$

subject to

$$\sum_{j} a_{ij} x_j \geq b_i, \quad \forall i$$

$$x_j \geq 0$$

Linear Surface Fitting

Given a set of observations $A = [a_{ij}]$ and b_i. Find weights on the columns of A and a scalar constant γ which best "predicts" the value of b on the basis of observations a_{ij}, assuming a linear model:

$$\min_{x,\gamma} \sum_{i=1}^{m} \left| \sum_{j} a_{ij} x_j + \gamma - b_i \right|$$

or, equivalently

$$\min_{x,\gamma,y} z = \sum_{i} y_i$$

subject to

$$-y_i \leq \sum_{j} a_{ij} x_j + \gamma - b_i \leq y_i$$

Note that the constraint ensures that each y_i is no smaller than the absolute value $|\sum_j a_{ij} x_j + \gamma - b_i|$.

Load Balancing

Balance computational work among n processors, distributing the load in such a way that the lightest-loaded processor has as heavy a load as possible:

- p_i Current load of processor $i = 1, 2, \ldots, n$
- L Total load to be distributed
- x_i Fraction of additional load L to be distributed to processor i, with $x_i \geq 0$ and $\sum_i x_i = 1$.
- γ minimum final loads after distribution of the new workload L

$$\max_{x,\gamma} \gamma$$

subject to

$$\gamma \leq p_i + x_i L, \quad \sum_i x_i = 1, \quad x_i \geq 0 \ \forall i$$

Outline

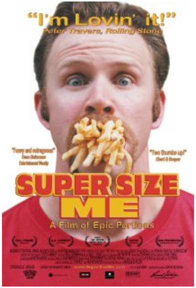
- Another LP example
- Linear algebra and indexed GAMS code
- Linear programs in standard form
- Supersize me!
- Visualization

The McDonald's Diet Problem

In words:

Minimize: the cost (or calories) of eating at McDonald's

Subject to: the total amounts of food or nutrients fall between certain minimum and maximum values



A GAMS Model

```
set     n       Nutritional needs /
                calo    Calories
                carbo   Carbohydrates
                protein Protein
                vita    Vitamin A
                vitc    Vitamin C
                calc    Calcium
                iron    Iron /,

        f       Foods /
                "Quarter Pounder w/ Cheese"
                "McLean Deluxe w/ Cheese"
                "Big Mac",
                "Filet-O-Fish",
                "McGrilled Chicken",
                "Fries, small",
                "Sausage McMuffin",
                "1% Lowfat Milk",
                "Orange Juice" /;
```

Source Data

http://nutrition.mcdonalds.com/getnutrition/nutritionfacts.pdf

McDonald's USA Nutrition Facts for Popular Menu Items

We provide a nutrition analysis of our menu items to help you balance your McDonald's meal with other foods you eat. Our goal is to provide you with the information you need to make sensible decisions about balance, variety and moderation in your diet.

																			% DAILY VALUE			
Nutrition Facts	**Serving Size**	**Calories**	**Calories from Fat**	**Total Fat (g)**	**% Daily Value****	**Saturated Fat (g)**	**% Daily Value****	***Trans* Fat (g)**	**Cholesterol (mg)**	**% Daily Value****	**Sodium (mg)**	**% Daily Value****	**Carbohydrates (g)**	**% Daily Value****	**Dietary Fiber (g)**	**% Daily Value****	**Sugars (g)**	**Protein (g)**	**Vitamin A**	**Vitamin C**	**Calcium**	**Iron**
Burgers & Sandwiches																						
Angus Chipotle BBQ Bacon†+++	10.3 oz (294 g)	800	350	39	60	18	88	2	145	49	2020	84	66	22	4	14	16	45	10	2	25	35

The Data

```
table a(f,n)    Nutritional content of foods
                                calo    carbo   protein vita    vitc    calc    iron
"Quarter Pounder w/ Cheese"     510     34      28      15      6       30      20
"McLean Deluxe w/ Cheese"       370     35      24      15      10      20      20
"Big Mac"                       500     42      25      6       2       25      20
"Filet-O-Fish"                  370     38      14      2       0       15      10
"McGrilled Chicken"             400     42      31      8       15      15      8
"Fries, small"                  220     26      3       0       15      0       2
"Sausage McMuffin"              345     27      15      4       0       20      15
"1% Lowfat Milk"                110     12      9       10      4       30      0
"Orange Juice"                  80      20      1       2       120     2       2;

table nutr(n,*) Nutrient requirements
        min     max
calo    2000    inf
carbo   350     375
protein 55      inf
vita    100     inf
vitc    100     inf
calc    100     inf
iron    100     inf;

table food(f,*) Food cost and requirements
                                cost    min     max
"Quarter Pounder w/ Cheese"     1.84    0       inf
"McLean Deluxe w/ Cheese"       2.19    0       inf
"Big Mac"                       1.84    0       inf
"Filet-O-Fish"                  1.44    0       inf
"McGrilled Chicken"             2.29    0       inf
"Fries, small"                  0.77    0       inf
"Sausage McMuffin"              1.29    0       inf
"1% Lowfat Milk"                0.6     0       inf
"Orange Juice"                  0.72    0       inf;
```

Model Equations

```
nonnegative
variables         Y(n)    Nutritional content
                  X(f)    Purchased quantity;

free
variable          COST      Total cost;

equations         ydef, objdef;

ydef(n)..         Y(n) =e= sum(f, X(f)*a(f,n));

objdef..          COST =e= sum(f, X(f) * food(f,"cost"));

model   mincost /ydef, objdef /;

Y.LO(n) = nutr(n,"min"); Y.UP(n) = nutr(n,"max");
X.LO(f) = food(f,"min"); X.UP(f) = food(f,"max");
```

First Run

```
solve mincost using lp minimizing COST;

*       Generate reports of the menu and diet:

parameter       menu    Resulting menu,;
menu("Cost","MinCost") = COST.L;
menu("Calories","MinCost") = Y.L("calo");
menu("ModelStat","MinCost") = mincost.modelstat;
menu("solvestat","MinCost") = mincost.solvestat;
menu(f,"MinCost") = X.L(f);
display menu;
```

First Run

	MinCost
Cost	14.856
Calories	3965.369
Quarter Pounder w/ Cheese	4.385
Fries, small	6.148
1% Lowfat Milk	3.422

Cheap, but
4000 calories!

Second Run

Put an upper bound on calories.

```
Y.UP("calo") = 2500;
solve mincost using lp minimizing COST;
```

Calories are down, cost is up and the diet looks better.

	MinCost	Cal2500
Cost	14.856	16.671
Calories	3965.369	2500.000
Quarter Pounder w/ Cheese	4.385	0.232
McLean Deluxe w/ Cheese		3.855
Fries, small	6.148	
1% Lowfat Milk	3.422	2.043
Orange Juice		9.134

Third Run

Try for a 2000 calorie diet.

```
Y.UP("calo") = 2000;
solve mincost using lp minimizing COST;
```

Not possible!

```
     MODEL   mincost                   OBJECTIVE  C
     TYPE    LP                        DIRECTION  MINIMIZE
     SOLVER  CPLEX                     FROM LINE  121

**** SOLVER STATUS     1 Normal Completion
**** MODEL STATUS      4 Infeasible
**** OBJECTIVE VALUE                 0.9121
```

Minimize calories, ignoring cost.

```
variable        CALO          Objective value -- calories;
equation        objcalo       Objective -- minimize calories;

objcalo..       CALO =e= Y("calo");

model mincal /ydef, objdef, objcalo/;

solve mincal using lp minimizing CALO;
```

Minimum calories is 2467 at a cost of $16.75:

	MinCost	Cal2500	MinCal
Cost	14.856	16.671	16.745
Calories	3965.369	2500.000	2466.981
Quarter Pounder w/ Cheese	4.385	0.232	
McLean Deluxe w/ Cheese		3.855	4.088
Fries, small	6.148		
1% Lowfat Milk	3.422	2.043	2.044
Orange Juice		9.134	9.119

Add some variety

```
X.UP(f) = 2;
solve mincost using lp minimizing COST;
```

More intersting cuisine. Cost is up a bit, calories are down relative to original solution.

	MinCost	MinCal	MinCostV
Cost	14.856	16.745	16.766
Calories	3965.369	2466.981	3798.077
Quarter Pounder w/ Cheese	4.385		2.000
McLean Deluxe w/ Cheese		4.088	2.000
Big Mac			2.000
Fries, small	6.148		1.423
Sausage McMuffin			1.000
1% Lowfat Milk	3.422	2.044	2.000
Orange Juice		9.119	2.000

Keep variety but minimize calories

```
X.UP(f) = 2;
solve mincal using lp minimizing COST;
```

Almost 3500 calories. Wow!

	MinCost	MinCal	MinCostV	MinCalV
Cost	14.856	16.745	16.766	17.248
Calories	3965.369	2466.981	3798.077	3488.287
Quarter Pounder w/ Cheese	4.385		2.000	1.952
McLean Deluxe w/ Cheese		4.088	2.000	2.000
Big Mac			2.000	
Filet-O-Fish				0.359
McGrilled Chicken				2.000
Fries, small	6.148		1.423	2.000
Sausage McMuffin			1.000	
1% Lowfat Milk	3.422	2.044	2.000	2.000
Orange Juice		9.119	2.000	2.000

Whole Numbers Solution

Move from *linear program* (LP) to *mixed integer programming* (MIP).

```
integer variable  XI(f)   Food purchase;
equation xidef;  xidef(f).. X(f) =e= XI(f);

XI.LO(f) = 0; XI.UP(f) = 2;

model   integerdiet /ydef, objdef, objcalo, xidef /;
solve integerdiet using MIP minimizing CALO;
```

This only makes a small increase in price and calories.

	MinCost	MinCal	MinCostV	MinCalV	MinCalInt
Cost	14.856	16.745	16.766	17.248	17.490
ModelStat	1.000	1.000	1.000	1.000	1.000
SolveStat	1.000	1.000	1.000	1.000	1.000
Quarter Pounder w/ Cheese	4.385		2.000	1.952	2.000
McLean Deluxe w/ Cheese		4.088	2.000	2.000	2.000
Big Mac			2.000		
Filet-O-Fish				0.359	1.000
McGrilled Chicken				2.000	2.000
Fries, small	6.148		1.423	2.000	1.000
Sausage McMuffin			1.000		
1% Lowfat Milk	3.422	2.044	2.000	2.000	2.000
Orange Juice		9.119	2.000	2.000	2.000
Calories	3965.369	2466.981	3798.077	3488.287	3530.000

Some GAMS Program Details

Include a set definition to sequence rows in the report – the *universal element list*:

```
set seq /Cost, Calo, ModelStat, SolveStat/;

              MinCost      Cal2500      Cal2000 ...
Cost           14.856       16.671       17.796
Calo         3965.369     2500.000     2000.000
ModelStat       1.000        1.000        4.000
SolveStat       1.000        1.000        1.000
...
```

Some GAMS Program Details

Uses a *batinclude subroutine* to add scenario results to the report parameters `menu` and `diet`:

```
parameter       menu  Resulting menu,
                diet  Resulting diet;

$onechov >%gams.scrdir%report.gms
menu("Cost","%1") = C.L;
menu("Calories","%1") = Y.L("calo");
menu("ModelStat","%1") = mincost.modelstat;
menu("solvestat","%1") = mincost.solvestat;
menu(f,"%1") = X.L(f);
diet("Cost","%1") = C.L;
diet("ModelStat","%1") = mincost.modelstat;
diet("SolveStat","%1") = mincost.solvestat;
diet(n,"%1") = Y.L(n);
$offecho

*       Define an environment variable to label the report:

solve mincost using lp minimizing COST;
$batinclude %gams.scrdir%report MinCost
```

Some GAMS Program Details

Compute the integer programming solution by introducing an integer variable and assigning it to the LP decision variable. All variables and equations from the LP remain in the model so reporting routine is unchanged!

```
integer variable        XI(f)   Food purchase;

equation        xidef;

xidef(f)..      X(f) =e= XI(f);

XI.LO(f) = 0;
XI.UP(f) = 2;

model   integerdiet /ydef, objdef, objcalo, xidef /;

solve integerdiet using MIP minimizing CAL;
```

Outline

- Another LP example
- Linear algebra and indexed GAMS code
- Linear programs in standard form
- Supersize me!
- Visualization

Recall the Brewery Profit Model

$$\max_{x,y} 120x + 90y$$

subject to:

$$4x + 2y \leq 4800$$

$$x + y \leq 1750$$

$$0 \leq x \leq 1000$$

$$0 \leq y \leq 1500$$

In which:

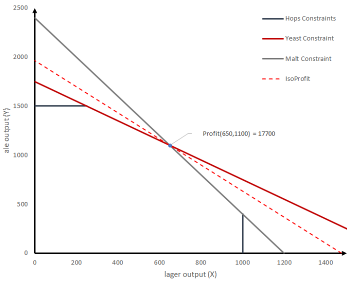
x : number of batches of lager produced

y : number of batches of ales produced

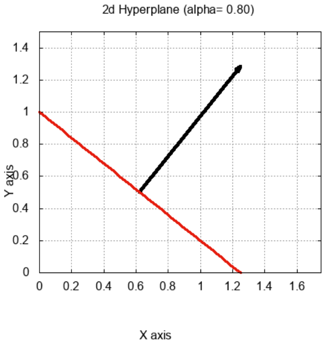
Visualization: Scatter Plots in Excel

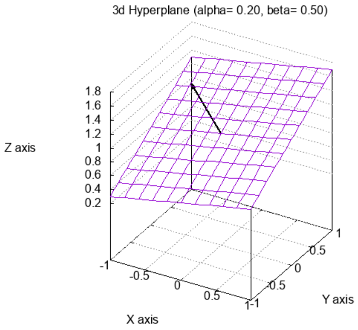
		Value	Scale
Quantity of malt on hand	*maltq*	4800	1
Quantity of yeast on hand	*yeastq*	1750	1
Malt requirements -- lager	*maltx*	4	1
Malt requirements -- ale	*malty*	2	1
Unit profit -- lager	*profitx*	12	1
Unit profit -- ale	*profity*	9	1
Isoprofit	*isoprofit*	17700	1
Maximum Profit	*maxprofit*	17700	

	x	*y*	formula(x)	formula(y)
WI Hops	0	1500	0	1500
	250	1500	=yeastq-1500	1500
DE Hops	1000	0	1000	0
	1000	400	1000	=(maltq-maltx*1000)/malty
Malt	1200	0	=maltq/maltx	0
	0	2400	0	=maltq/malty
Yeast	0	1750	0	=yeastq
	1750	0	=yeastq	0
Isoprofit	1475	0	=isoprofit/profitx	=0
	0	1966.666667	0	=isoprofit/profity
Optimum	650	1100	=(maltq-malty*yeastq)/(maltx-malty)	=yeastq-B27
Label:	Profit(650,1100) = 17700		="Profit("&TEXT(B27,"0")&","&TEXT(C27,"0")&") = "&TEXT(maxprofit,"0")	

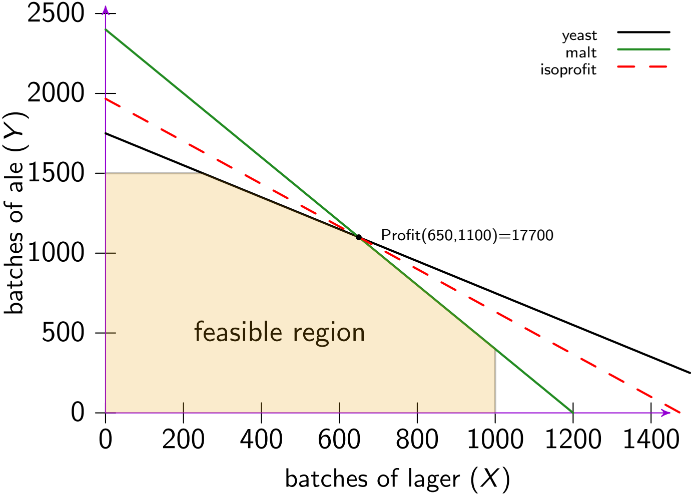


Visualization: GNUPLOT





Visualization: GNUPLOT



GNUPLOT Command File

```
#       Begin with a reset so we can make changes and immediately reload:

reset

#       Define parameters are user-defined GNUPLOT variables:

maltq=4800
yeastq=1750
maltx=4
malty=2
profitx=12
profity=9

#       Calculate the optimum, assuming that it is where the yeast
#       and malt constraints are both binding:

xmax=(maltq-malty*yeastq)/(maltx-malty)
ymax=yeastq-xmax
maxprofit=xmax*profitx+ymax*profity

#       Calculate points where the hops constraints intersect
#       the yeast and malt constraints:

xlim = yeastq-1500
ylim = (maltq-maltx*1000)/malty
set style arrow 1 nohead linecolor rgb "gray" linewidth 2 dashtype solid
set arrow 1 from 1000,0,0 to 1000,ylim,0 arrowstyle 1
set arrow 2 from 0,1500,0 to xlim,1500,0 arrowstyle 1
```

GNUPLOT Command File (cont)

```
#       Define linear functions representing the yeast and malt
#       constraints as well as the isoprofit line at the optimal
#       point:

yeast(x)=yeastq-x
malt(x)=(maltq-maltx*x)/malty
isoprofit(x) = ymax-profitx*(x-xmax)/profity


#       Set up axes:

set xrange [0:1500]
set yrange [0:2500]
set ylabel 'batches of ale (y)' offset -1,0
set xlabel 'batches lager (x)' offset 0,-1
set xtics axis
set ytics axis
unset border
set arrow 3 from 0,0 to 1450,0  linestyle 1
set arrow 4 from 0,0 to 0,2550  linestyle 1

#       Label the feasible region:

set label "feasible region" at 200,500

#       Put a black circle at the optimum:

optimum = sprintf('(%.f,%.f)',xmax,ymax)
set label optimum at xmax+30,ymax+30
set object circle at xmax,ymax front size 6 \
      fillstyle solid 1 fillcolor rgb "black"
```

GNUPLOT Command File (cont)

```
#       Generate a data file with extreme points of the feasible region:

set print "feasible.dat"
print sprintf('%.f %.f',0,0)
print sprintf('%.f %.f',0,1500)
print sprintf('%.f %.f',xlim, 1500)
print sprintf('%.f %.f',xmax, ymax)
print sprintf('%.f %.f',1000, ylim)
print sprintf('%.f %.f',1000, 0)
unset print

#       Define the style to be used for "filledcurves" to denote the
#       feasible region:

set style fill transparent solid 0.2 noborder

set style line 1 linecolor "black" linewidth 2 dashtype 1
set style line 2 linecolor "forest-green" linewidth 2 dashtype 1
set style line 3 linecolor "red" linewidth 2 dashtype 2

plot yeast(x) ls 1, malt(x) ls 2, isoprofit(x) ls 3, \
      'feasible.dat' using 1:2 with filledcurves below notitle
```