Linear Programming

Example #2: Dahlby Outfitters

- Dahlby Outfitters wants to introduce a new trail mix. Each ingredient contain amounts of vitamins, minerals, proteins, and calories. Product needs to have certain minimal nutritional profile. Want to find the optimal mix that minimizes cost.

Information

	Grams per Pound					
Component	Seeds	Raisins	Flakes	Pecans	Walnuts	Nutritional Requirements
Vitamins	10	20	10	30	20	20
Minerals	5	7	4	9	2	10
Protein	1	4	10	2	1	15
Calories	500	450	160	300	500	600
Cost/pound	$4	$5	$3	$7	$6	

Optimization Set up

- Objective Function:

  Minimize 4S + 5R + 3F + 7P +6W

- Constraints

10S + 20R + 10F +30P + 20W $\geq$ 20

5S + 7R +4F + 9P + 2W $\geq$10

S + 4R + 10F + 2P + W $\geq$15

500S + 450R + 160F + 300P +500W $\geq$600

In SAS

```
proc optmodel;
var Seeds>=0, Raisins>=0, Flakes>=0, Pecans>=0, Walnuts>=0;
min Cost=4*Seeds + 5*Raisins + 3*Flakes + 7*Pecans +6*Walnuts;
con Vitamins: 10*Seeds + 20*Raisins + 10*Flakes + 30*Pecans +
20*Walnuts>=20;
con Minerals: 5*Seeds + 7*Raisins + 4*Flakes + 9*Pecans +
2*Walnuts>=10;
con Protein: Seeds + 4*Raisins + 10*Flakes + 2*Pecans + Walnuts>=15;
con Calories: 500*Seeds + 450*Raisins + 160*Flakes + 300*Pecans +
500*Walnuts>=600;
solve with lp;
print Seeds Raisins Flakes Pecans Walnuts;
quit;
```

In R

```
f.obj=c(4,5,3,7,6)
f.con=matrix(c(10,20,10,30,20,5,7,4,9,2,1,4,10,2,1,500,450,160,3
00,500),nrow=4,byrow=T)
f.dir=c(">=",">=",">=",">=")
f.rhs = c(20,10,15,600)

lp.model=lp (direction="min", f.obj, f.con, f.dir,
f.rhs,compute.sens = 1)
lp.model$status
f.names=c('Seeds','Raisins','Flakes', 'Pecans','Walnuts')
names(lp.model$solution)=f.names
lp.model$solution
lp.model$objval
```

Gurobi (in R)

```
model <- list()
model$obj        <- c(4,5,3,7,6)
model$modelsense <- "min"
model$rhs        <- c(20,10,15,600)
model$sense      <- c(">=",">=",">=",">=")
model$vtype      <- "C"
model$A          <-
matrix(c(10,20,10,30,20,5,7,4,9,2,1,4,10,2,1,500,450,160,300,500),nr
ow=4,byrow=T)
result <- gurobi(model, list())
print(result$status)
f.names=c('Seeds','Raisins','Flakes', 'Pecans','Walnuts')
names(result$x)=f.names
print(result$x)
print(result$objval)
```

Output

```
> lp.model$status
[1] 0
> lp.model$solution
    Seeds    Raisins          Flakes       Pecans       Walnuts
0.4773270 0.3341289 1.3186158 0.0000000 0.0000000
> lp.model$objval
[1] 7.5358
```

Binding Constraints

- A constraint is **binding** if the LHS of a constraint equals the RHS of the constraint (LHS is the value of the constraint when the solution values are used…RHS is the original value of the parameter set by the problem)
- For example, in previous example, take the Mineral constraint:

Mineral: **5***Seeds + **7***Raisins + **4***Flakes + **9***Pecans + **2***Walnuts ≥ 10

5*(0.4773270) + **7***(0.3341289) + **4***(1.3186158) + **9***(0) + **2***(0) = 10

Binding Constraints

- Easiest software to get this from is Gurobi.
- It is called slack:

result$slack

From this example

result$slack

-4.642005 0.000000 0.000000 0.000000

BINDING CONSTRAINTS!

Sensitivity

- How sensitive is the solution to changes in the parameter values in the constraints? SHADOW PRICE
- How sensitive is the solution to changes in the coefficients in the objective function? REDUCED COST
- We can perform a sensitivity analysis to investigate these ideas further.

SHADOW PRICE

Illustration of shadow price

- Let's go back to the furniture example and see if there are any binding constraints:

result$slack

[1] 0 325 550 360 25 0

- So the first constraint (fabrication) is a binding constraint
- See what happens when we increase fabrication by 1 hour: (go to Gurobi)

Shadow/Dual Prices

- The **shadow price** of a constraint indicates the amount by which the objective function value changes given a unit increase (or decrease…depending on the direction of the constraint) in the RHS (or parameter) of the constraint, assuming all other parameters remain constant.
- "If we had one additional unit of a constraint, then the objective function would change by the shadow price."
- Shadow prices for nonbinding constraints are always zero.

Getting shadow prices/dual prices

In SAS:

print fabrication.dual assembly.dual shipping.dual demandc.dual demandd.dual demandt.dual;

Fabrication.DUAL	Assembly.DUAL	Shipping.DUAL	DemandC.DUAL	DemandD.DUAL	DemandT.DUAL
4	0	0	0	0	10

In R:

lp.model$duals

```
[1]  4  0  0  0  0 10 -1  0  0
```

In Gurobi:

result$pi

```
[1]  4  0  0  0  0 10
```

Another application for shadow prices

- Veerman Furniture Company
  - Considering new product: Stools
  - Marginal profit is $10 per stool
  - Each stool takes 3 hours for fabrication, 2 hours for assembly and 2 hours for shipping (assuming that we are still constrained to same totals from before for fabrication, assembly and shipping)
  - Is it profitable to make this new product?

$10 – 3*($4) – 2*($0) – 2*($0) = -$2

***Be careful, if other quantities change, it would be better to put into the problem and rerun (only good for small changes)

REDUCED COST

Reduced Costs

- The **reduced cost** is the amount by which an objective function coefficient would have to improve (so increase for maximization problem, decrease for minimization problem) before it would be possible for a corresponding variable to assume a positive value in the optimal solution (take absolute value for maximization).
- Will be nonzero when the optimal amount of a decision variable is 0.

Reduced Cost

In SAS:

print chairs.rc desks.rc tables.rc;

Chairs.RC	Desks.RC	Tables.RC
-1	0	0

In R:

lp.model$duals

[1] 4 0 0 0 0 10 -1 0 0

In Gurobi:

result$rc

[1] -1 0 0

BLENDING MODELS

Blending Models

- Blending models are usually stated as proportions (for example, we want chairs to constitute at least 25% of the overall product mix).
- We can rewrite this percent in terms of a linear constraint:

$$\frac{C}{C + D + T} \geq 0.25$$

$$0.75C - 0.25D - 0.25T \geq 0$$

Example

- Diaz Coffee Company blends three types of coffee beans (Brazilian, Colombian, Peruvian). These different blends have different aroma and strength ratings (see table below). They would like an average aroma rating of at least 78 and an average strength rating of at least 16. Goal: Make 4,000,000 pounds with lowest cost.

Bean	Aroma Rating	Strength Rating	Cost/lb	Pounds available
Brazilian	75	15	$0.5	1,500,000
Colombian	60	20	$0.6	1,200,000
Peruvian	85	18	$0.7	2,000,000

Set up

Min: 0.5B + 0.6C + 0.7P

Constraints:

-3*B - **18***C + **7***P >=**0**;

-B + **4***C + **2***P >=**0**;

B + C + P = **4000000**;

B <=**1500000**;

C <=**1200000**;

P <=**2000000**;

In SAS

```
proc optmodel;
var B>=0, C>=0, P>=0;
min Cost=0.5*B + 0.6*C + 0.7*P;
con Aroma: -3*B - 18*C + 7*P >=0;
con Strength: -B + 4*C + 2*P >=0;
con Total: B + C + P = 4000000;
con Brazil: B <=1500000;
con Colom: C <=1200000;
con Purivian: P <=2000000;
solve with lp;
print B C P;
print B.RC C.RC P.RC;
print B.dual C.dual P.dual Aroma.dual
Strength.dual Total.dual
        Brazil.dual Colom.dual Purivian.dual;
quit;
```

Algorithm	Dual Simplex
Objective Function	Cost
Solution Status	Optimal
Objective Value	2448000
Primal Infeasibility	0
Dual Infeasibility	0
Bound Infeasibility	0

B	C	P
1500000	520000	1980000

B.RC	C.RC	P.RC
0	0	0

B.DUAL	C.DUAL	P.DUAL	Aroma.DUAL	Strength.DUAL	Total.DUAL	Brazil.DUAL	Colom.DUAL	Purivian.DUAL
0	0	0	0.004	0	0.672	-0.16	0	0

In R

```
f.obj=c(0.5,0.6,0.7)
f.con=matrix(c(-3,-18,7,-1,4,2,1,1,1,1,0,0,0,1,0,0,0,1),nrow=6,byrow=T)
f.dir=c('>=','>=','=','<=','<=','<=')
f.rhs=c(0,0,4000000,1500000,1200000,2000000)
lp.model=lp (direction="min", f.obj, f.con, f.dir, f.rhs,compute.sens = 1)
lp.model$status
f.names=c('Brazilian','Colombian','Peruvian')
names(lp.model$solution)=f.names
lp.model$solution
lp.model$objval
lp.model$duals
```

Output

```
> lp.model$solution
Brazilian Colombian  Peruvian
  1500000    520000   1980000
> lp.model$objval
[1] 2448000
> lp.model$duals
[1]  0.004  0.000  0.672 -0.160  0.000  0.000  0.000  0.000  0.000
```

In Gurobi

```
model <- list()
model$obj        <- c(0.5,0.6,0.7)
model$modelsense <- "min"
model$rhs        <- c(0,0,4000000,1500000,1200000,2000000)
model$sense      <- c(">=",">=","=","<=","<=","<=")
model$vtype      <- "C"
model$A          <- matrix(c(-3,-18,7,-
1,4,2,1,1,1,1,1,0,0,0,1,0,0,0,1),nrow=6,byrow=T)
result <- gurobi(model, list())
print(result$status)
f.names=c('Brazilian','Colombian','Peruvian')
names(result$x)=f.names
print(result$x)
print(result$objval)
result$rc
result$pi
```

Output

```
> print(result$x)
Brazilian Colombian  Peruvian
  1500000    520000   1980000
> print(result$objval)
[1] 2448000
> result$rc
[1] 0 0 0
> result$pi
[1]  0.004  0.000  0.672 -0.160  0.000  0.000
```