AAE 706

Optimization Models: Linear Programming

Thomas F. Rutherford

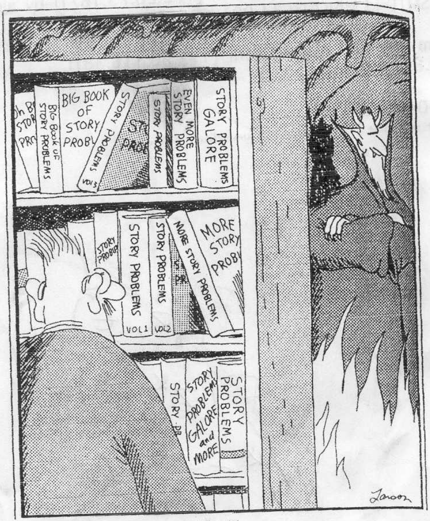
Department of Agricultural and Applied Economics
University of Wisconsin, Madison

January 30, 2023

Today's Agenda

- LP models with absolute value (the $\mathcal{L}_1$ norm)
- Min-max (*box* norm) LP models (the $\mathcal{L}_\infty$)
- Getting started with electricity

Why Linear Programming?

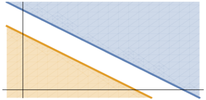


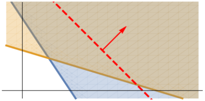
Hell's library

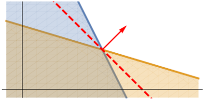
Solutions of a Linear Program

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

Grade Inflation

	MAT	CHE	ANT	REL	POL	ECO
John			C+	A	B+	A-
Paul	B+	A-			A-	B
George	B	B+		A	A-	
Ringo	A		B-	A		A-

where

MAT 402	Advanced Addition
CHE 333	Intermediate Explosives
ANT 536	The Behavior of Anthropologists
REL 101	Atheism
POL 242	Constitutional Manipulation
ECO 666	The Root Of All Evil

The Model

We assume that every grade, g_{ij} for student i in course j, can be decomposed as a sum

1. aptitude, a_i, of student i,
2. easiness, e_j, of course j,
3. plus some small error ϵ_{ij}

That is, $g_{ij} = a_i + e_j + \epsilon_{ij}$

The g_{ij}'s are data. We wish to find the a_i's and the e_j's that minimizes the sum of the absolute values of the ϵ_{ij}'s:

$$\min \sum_{i,j} |\epsilon_{ij}|$$

subject to

$$\begin{aligned} g_{ij} &= a_i + e_j + \epsilon_{ij} \quad \forall i, j \\ \sum_j e_j &= 0 \end{aligned}$$

The Absolute Value Trick

$$\min \sum_{i,j} |\epsilon_{ij}|$$

subject to

$$\begin{aligned} g_{ij} &= a_i + e_j + \epsilon_{ij} \quad \forall i,j \\ \sum_j e_j &= 0 \end{aligned}$$

is equivalent to

$$\min \sum_{i,j} t_{ij}$$

subject to

$$\begin{aligned} t_{ij} &\geq g_{ij} - (a_i + e_j) \quad \forall i,j \\ t_{ij} &\geq -(g_{ij} - (a_i + e_j)) \quad \forall i,j \\ \sum_j e_j &= 0. \end{aligned}$$

Grade Estimation

```
$title  Grade Inflation -- L1 Estimation

set     seq /Overall/;

set     s       Students /
        Alex, Andy, Ariel, Billy, Bobby, Brett, Brook, Cameron, Cary, Casey,
        Chris, Dale, Dara, Darcy, Daryl, Devyn, Dominique, Drew, Emerson, Esme,
        Harley, Jade, Jordan, Kelly, Kim, Lindsey, Lou, Max, Meryl, Morgan,
        Peyton, Porntip, Reese, Robin, Sam, Skye, Sunny, Sydney, Tanner, Tracy /;

set     c       Courses /
        Apology, Astrology, Chronology, Cosmology, Demonology, Ecology,
        Epistemology, Etymology, Eulogy, Genealogy, Geology, Gynecology,
        Ideology, Immunology, Methodology, Morphology, Nephrology, Ontology,
        Pathology, Pharmacology, Philology, Phrenology, Psychology,
        Scientology, Seismology, Sociology, Statistics, Tautology, Technology,
        Terminology, Theology, Topology, Urology /;

set     g       Grades /A+, A, A-, B+, B, B-, C+, C, C-, D+, D, D-, F/;

set     marks(s,c,g) /
$ondelim
$include grades.txt
$offdelim
/;
```

grades.txt

```
Alex    Epistemology    B-
Alex    Topology        B-
Alex    Demonology      A-
Alex    Ecology         B
Alex    Immunology      A-
Alex    Methodology     A+
Alex    Nephrology      B+
Alex    Terminology     A
Andy    Phrenology      A-
Andy    Apology         A
Andy    Philology       A-
Andy    Demonology      A+
Andy    Eulogy  A+
Andy    Pathology       A+
Andy    Technology      A+
Ariel   Epistemology    B
Ariel   Topology        B+
Ariel   Etymology       A+
Ariel   Eulogy  A+
Ariel   Genealogy       A-
Ariel   Morphology      A+
Ariel   Pathology       A-
Ariel   Technology      A+
Billy   Etymology       A+
Billy   Psychology      A+
Billy   Apology         A+
Billy   Cosmology       A+
```

Grade Estimation

```
parameter mark(g)       Translation of grades to marks /
        A+ 4.3,  A 4.0, A- 3.7, B+ 3.3, B 3.0, B- 2.7, C+ 2.3,
        C 2.0, C- 1.7, D+ 1.3, D 1, D- 0.7, F 0 /;

parameter       grade(s,c)      Numeric mark;

grade(s,c) = sum(marks(s,c,g), mark(g));


set     r(s,c)          Indication -- registration of student s in course c;
r(s,c) = yes$sum(marks(s,c,g),1);

alias (i,s), (j,c);

NONNEGATIVE
VARIABLE                T(i,j)  Absolute deviation;

VARIABLE                E(j)    Easiness of course j
                        A(i)    Aptitude of student i,
                        Z       Objective (least-squares calibration);

equations objdef, aveeasy, posdev, negdev;

objdef..                Z =g= sum((i,j),T(i,j));

aveeasy..               sum(j, E(j)) =e= 0;

posdev(r(i,j))..        T(i,j) =g= grade(i,j) - (A(i) + E(j));

negdev(r(i,j))..        T(i,j) =g= A(i) + E(j) - grade(i,j);

model grading /all/;
solve grading using LP minimizing Z;
```

Grade Estimation

```
parameter       results Summary of results;
results(s,"Overall","actual") = sum(r(s,c), grade(s,c))/sum(r(s,c), 1);
results(s,"Overall","model") = A.L(s);
results(r(s,c),"actual") = grade(s,c);
results(r(s,c),"model") = A.L(s) + E.L(c);
option results:1;
display results;

----    373 PARAMETER results  Summary of results

                            actual       model

Alex     .Overall              3.4         3.6
Alex     .Demonology           3.7         3.7
Alex     .Ecology              3.0         3.4
Alex     .Epistemology         2.7         2.7
Alex     .Immunology           3.7         4.0
Alex     .Methodology          4.3         4.0
Alex     .Nephrology           3.3         3.6
Alex     .Terminology          4.0         4.0
Alex     .Topology             2.7         3.7
Andy     .Overall              4.1         4.2
Andy     .Apology              4.0         4.0
Andy     .Demonology           4.3         4.3
Andy     .Eulogy               4.3         4.6
Andy     .Pathology            4.3         4.6
Andy     .Philology            3.7         3.6
Andy     .Phrenology           3.7         3.7
Andy     .Technology           4.3         4.3
```

The Box Norm Model

An alternative approach is to use a linear programming model to fit the model based on a $\mathcal{L}_\infty$ (box) norm to measure the goodness of fit. This is representable as the following linear program:

$$\min \max_{ij} |\epsilon_{ij}| = Z$$

subject to:

$$\sum_j e_j = 0$$

$$Z \geq g_{ij} - a_i - e_j$$

$$Z \geq a_i + e_j - g_{ij}$$

Quadratic Programming Formulation

A third approach based on a *quadratic programming model* is the conventional least squares model which employs the $\mathcal{L}_2$ norm:

$$\min \sum_{ij} (a_i + e_j - g_{ij})^2$$

subject to:

$$\sum_{j} e_j = 0$$

Exercise for Wednesday.

Concert Planning

The promoters of a rock concert must perform the tasks shown in the GAMS code below before the concert can be held:

```
set activity /
        A       "Find Site",
        B       "Find Engineers",
        C       "Hire Opening Act",
        D       "Set Radio and TV Ads",
        E       "Set Up Ticket Agents",
        F       "Prepare Electronics",
        G       "Print Advertising",
        H       "Set up Transportation",
        I       "Rehearsals",
        J       "Last-Minute Details"/;
```

Concert Planning (cont.)

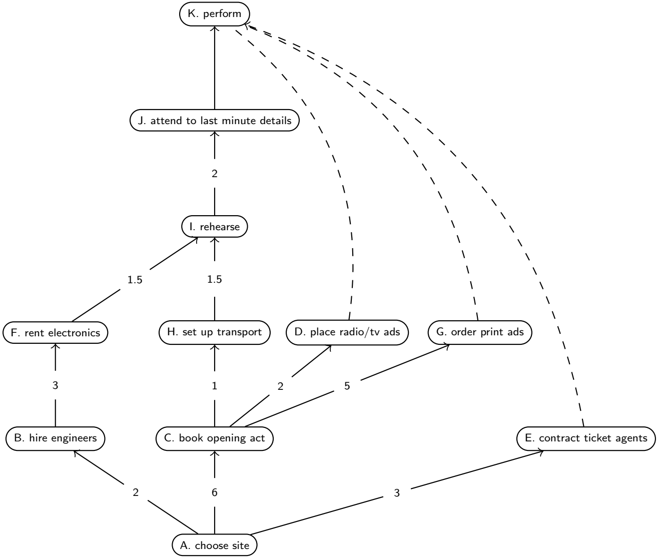
Each activity requires some time:

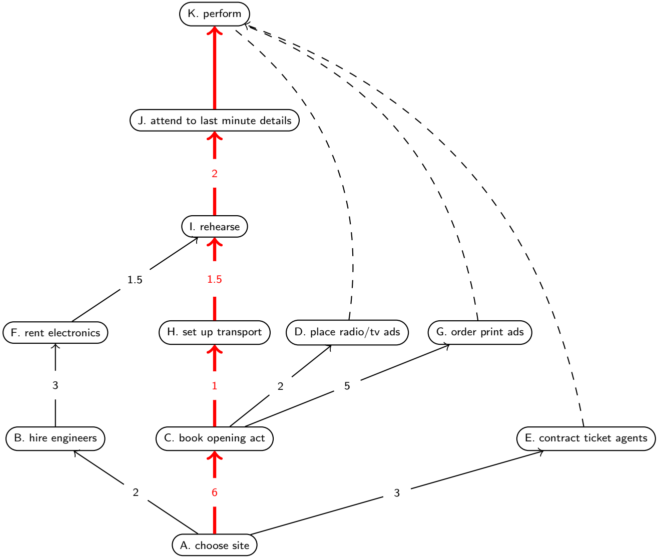
```
parameter duration(activity) "in days" /
        A 3, B 2, C 6, D 2, E 3, F 3, G 5, H 1, I 1.5, J 2 /;
```

Certain activities must be completed before subsequent activities may be initiated. For instance, a concert site must be found before we find engineers, hire an opening act or set up ticket agents, and so forth.

```
alias (activity,i,j);

set prec(i,j) "Precedence order" /
        A.(B,C,E)
        B.F
        C.(D,G,H)
        (F,H).I
        I.J /;
```

Set up a linear program to find the project duration (that is the minimum number of days needed to prepare for the concert).


K. perform
J. attend to last minute details
2
I. rehearse
1.5
1.5
F. rent electronics
H. set up transport
D. place radio/tv ads
G. order print ads
3
1
2
5
B. hire engineers
C. book opening act
E. contract ticket agents
2
6
3
A. choose site


K. perform
J. attend to last minute details
2
I. rehearse
1.5
1.5
F. rent electronics
H. set up transport
D. place radio/tv ads
G. order print ads
3
1
2
5
B. hire engineers
C. book opening act
E. contract ticket agents
2
6
3
A. choose site

Concert Planning Model - Scalar

```
$title  Critical Path Model -- Scalar Format

FREE
VARIABLE
        T       Completion time;

NONNEGATIVE
VARIABLES
        A       "Start time: find site",
        B       "Start time: find engineers",
        C       "Start time: hire opening act",
        D       "Start time: set radio and TV ads",
        E       "Start time: set up ticket agents",
        F       "Start time: prepare electronics",
        G       "Start time: print advertising",
        H       "Start time: set up transportation",
        I       "Start time: rehearsals",
        J       "Start time: last-minute details";
```

Concert Planning Model - Scalar

```
equations

*       Completion times no sooner than any of the tasks:

t_A, t_B, t_C, t_D, t_E, t_F, t_G, t_H, t_I, t_J

*       Task which must be completed before subsequent task:

s_AB, s_AC, s_AE, s_BF, s_CD, s_CG, s_CH, s_FI, s_HI, s_IJ;
```

Concert Planning Model - Scalar

```
t_A.. T =G= A+3; t_B.. T =G= B+2; t_C.. T =G= C+6;
t_D.. T =G= D+2; t_E.. T =G= E+3; t_F.. T =G= F+3;
t_G.. T =G= G+5; t_H.. T =G= H+1; t_I.. T =G= I+1.5;
t_J.. T =G= J+2;

s_AB.. A+3 =L= B; s_AC.. A+3 =L= C; s_AE.. A+3 =L= E;
s_BF.. B+2 =L= F; s_CD.. C+6 =L= D; s_CG.. C+6 =L= G;
s_CH.. C+6 =L= H; s_FI.. F+3 =L= I; s_HI.. H+1 =L= I;
s_IJ.. I+1.5 =L= J;

model cpm /all/;
solve cpm using lp minimizing T;
```

Use of Tuples in GAMS

A *tuple* in GAMS is a multi-dimensional set. It corresponds to a logical (yes/no) data structure. Consider the following code fragment:

```
Set     i /i1*i4/,
        j /j1*j5/,
        ij(i,j) /i1.j1, i2.(j1,j2), (i3,i4).(j3*j5)/;

option ij:0:0:8;

display i, j, ij;
```

which produces the following listing:

```
----      5 SET i
i1,    i2,    i3,    i4

----      5 SET j
j1,    j2,    j3,    j4,    j5

----      5 SET ij
i1.j1,    i2.j1,    i2.j2,    i3.j3,    i3.j4,    i3.j5,    i4.j3,    i4.j4
i4.j5
```

Use of Tuples in GAMS

A *tuple* can be used to restrict assignments, either as a mask over the set domain:

```
parameter a(i,j), b(i,j);

a(i,j) = uniform(0,1);

b(ij(i,j)) = a(i,j);

display a,b;
```

```
----     14 PARAMETER a

              j1          j2          j3          j4          j5
i1         0.172       0.843       0.550       0.301       0.292
i2         0.224       0.350       0.856       0.067       0.500
i3         0.998       0.579       0.991       0.762       0.131
i4         0.640       0.160       0.250       0.669       0.435


----     14 PARAMETER b

              j1          j2          j3          j4          j5
i1         0.172
i2         0.224       0.350
i3                                 0.991       0.762       0.131
i4                                 0.250       0.669       0.435
```

Declaration of Multi-Connected Tuples

Notice that:

```
set prec(i,j) "Precedence order" /
A.(B,C,E), B.F, C.(D,G,H), (F,H).I, I.J /;
```

is equivalent to

```
set prec(i,j) "Precedence order" /
A.B, A.C, A.E, B.F, C.D,C.G,C.H, F.I,H.I, I.J /;
```

Likewise

```
set ij(i,j) /(A,B).(C,D,E)/;
```

is equivalent to:

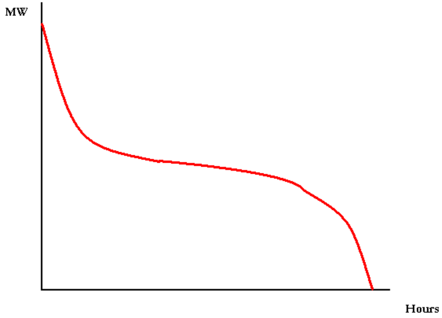
```
set ij(i,j) /A.C,A.D,A.E, B.C,B.D,B.E)/;
```

Concert Planning: An Indexed Formulation

```
set activity / A*J/;    alias (activity,i,j);

parameter duration(activity) "in days" /
  A 3, B 2, C 6, D 2, E 3, F 3, G 5, H 1, I 1.5, J 2 /;

set prec(i,j) "Precedence order" /
  A.(B,C,E), B.F, C.(D,G,H), (F,H).I, I.J /;

FREE VARIABLE        T       Time to completion;

NONNEGATIVE VARIABLE S(i)    Starting time for activity i;

EQUATIONS   ctime(i) Completion time,  ptime(i,j) Sequence;

ctime(i)..          T =g= S(i) + duration(i);

ptime(prec(i,j)).. S(i) + duration(i) =L= S(j);

model cpm /all/;   solve cpm using lp minimizing T;
```

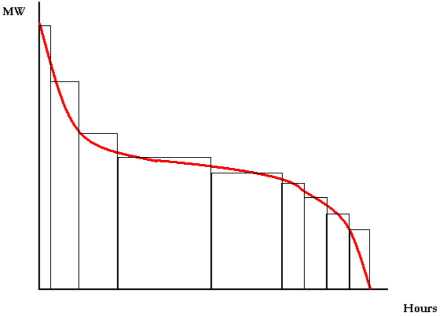
Electricity Models: The Load-Duration Curve

A *load-duration curve* portrays electricity demand over a year in terms of sorted decreasing quantity. Typically constructed on an hourly basis (8760 hours per year):



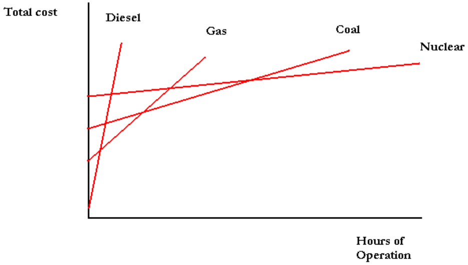
Load Segment Approximation of a Load-Duration Curve

If we want to model electricity sector investment decisions, we need to work with an approximation to the load-duration curve. It does not take too many *load segments* to produce a coherent representation:



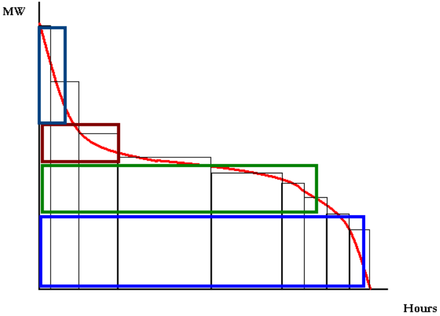
Units Characteristics Depend on Load Factors

- *Peak* generating units typically operate a small number of hours per year and tend to have low capital costs and high variable costs.
- *Base load* generating units typically operate a large number of hours per year and tend to have high capital costs and low variable costs.



Load Dispatch Curve

Investment and dispatch decisions are made jointly: when a utility invests in new generating capacity, it must take into account overall load duration curve and characteristics of existing capacity:



A Canonical Electricity Investment Model

```
$title  Simple Model of Electricity Market Dispatch

*       1. Define sets.

set     s       Load segments (peak to base load) /s1*s9/,

        j       Generating units  /Nuclear, Coal, Gas, Diesel/

        i       Demand categories /
                rsd     Residential,
                com     Commercial,
                ind     Industrial /;
```

A Canonical Electricity Investment Model (cont.)

```
*       2. Read data.

table   load(s,*)           Electricity load

*                           Demand Shares
*                           -------------------
        qref      hours     rsd      com      ind
s1      1.00        310     0.7      0.2      0.1
s2      0.79        750     0.5      0.2      0.3
s3      0.59       1020     0.4      0.3      0.3
s4      0.50       2460     0.4      0.3      0.3
s5      0.44       1910     0.3      0.3      0.4
s6      0.40        580     0.2      0.3      0.5
s7      0.35        580     0.1      0.3      0.6
s8      0.28        600     0.1      0.3      0.6
s9      0.23        540     0.1      0.3      0.6;

table   supply(j,*)         Electricity supply technology
                       mc   cap
Nuclear                 4   0.30
Coal                    6   0.30
Gas                     7   0.20
Diesel                  9   0.30;
```

A Canonical Electricity Investment Model (cont.)

```
parameter       mc(j)   Marginal cost of dispatch,
                cap(j)  Capacity constraint,
                qref(s) Reference demand
                h(s)    Hours;

h(s) = load(s,"hours");
mc(j) = supply(j,"mc");
cap(j) = supply(j,"cap");
qref(s) = load(s,"qref");
```

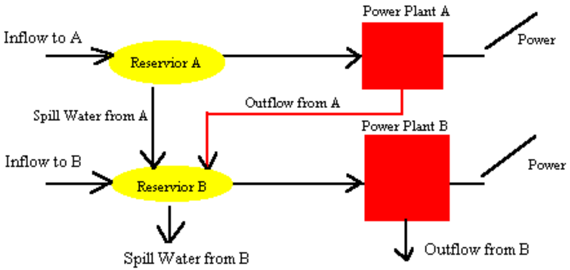
A Canonical Electricity Investment Model (cont.)

```
*       3. Compute reference prices for each of the segments using
*       a linear program.

variables       TOTCOST Objective function (dispatch cost);

nonnegative
variables       Y(j,s)  Dispatch;

equations       costdef, demand;

costdef..       TOTCOST =e= sum((j,s), mc(j)*Y(j,s));

demand(s)..     sum(j, Y(j,s)) =e= qref(s);

model mincost /costdef, demand/;

Y.UP(j,s) = supply(j,"cap");

solve mincost minimizing TOTCOST using LP;
```

A Canonical Electricity Investment Model (cont.)

```
parameter       dispatch(s,j)   Generation by load segment and technology;
dispatch(s,j) = Y.L(j,s);
display dispatch;

----     70 PARAMETER dispatch  Generation by load segment and technology

       Nuclear        Coal         Gas      Diesel
s1       0.300       0.300       0.200       0.200
s2       0.300       0.300       0.190
s3       0.300       0.290
s4       0.300       0.200
s5       0.300       0.140
s6       0.300       0.100
s7       0.300       0.050
s8       0.280
s9       0.230
```

The General Eccentric Power and Lighting Company has a system consisting of two dams and their associated reservoirs and power plants on a river. The important flows of power and water are shown in the following diagram:



Data

In the following table, all quantities measuring water are in units of 1000 cubic meters (TCM). Power is measured in megawatt hours (MWH's).

	A	B	Units
Storage Capacity	2000	1500	TCM
Minimum allowable level	1200	800	TCM
Predicted inflow:			
March	200	40	TCM
April	130	15	TCM
March 1 level	1900	850	TCM
Water-Power Conversion	400	200	MWH/TCM
Power Plant Capacity	60,000	35,000	MWH/month

Power can be sold at CHF 5.00 per MWH for up to 50,000 MWH each month, and excess power above that figure can be sold for CHF 3.50 per MWH. Assume flow rates in and out through the power plants are constant within the month. If the capacity of the reservoir is exceeded, the excess water runs down the spillway and bypasses the power plant. A consequence of these assumptions is that the maximum and minimum water-level constraints need to be satisfied only at the end of the month.

Formulate a linear program to maximize the amount of money General Eccentric receives for the power it sells during the months of March and April, given the constraints.

Variables – Scalar Model

```
$title  Hydro Power Planning Model -- Scalar Format

FREE
VARIABLE
        REVENUE Aggregate earnings on sales in March and April;

NONNEGATIVE
VARIABLES
        PA_MAR  Power production -- plant A in March
        PA_APR  Power production -- plant A in April
        PB_MAR  Power production -- plant B in March
        PB_APR  Power production -- plant B in April

        SA_MAR  Water spilled -- reservior A in March
        SA_APR  Water spilled -- reservior A in April
        SB_MAR  Water spilled -- reservior B in March
        SB_APR  Water spilled -- reservior B in April

        LA_MAR  Level -- reservior A start of March (exogenous)
        LA_APR  Level -- reservior A start of April
        LA_MAY  Level -- reservior A start of May
        LB_MAR  Level -- reservior A start of March (exogenous)
        LB_APR  Level -- reservior A start of April
        LB_MAY  Level -- reservior A start of May

        EH_MAR  High value electricity generation -- March
        EL_MAR  Low value electricity generation -- March
        EH_APR  High value electricity generation -- April
        EL_APR  Low value electricity generation -- April;
```

Bounds – Scalar Model (cont.)

```
*       Levels at the beginning of March are specified exogenously:

LA_MAR.FX = 1900;
LB_MAR.FX =  850;

*       Power can be sold at CHF 5 per MWH up to 50,000 MWH per month:

EH_MAR.UP = 50000;
EH_APR.UP = 50000;

*       Storage capacity:

LA_APR.UP = 2000;
LA_MAY.UP = 2000;
LB_APR.UP = 1500;
LB_MAY.UP = 1500;

*       Minimum reservior levels:

LA_APR.LO = 1200;
LA_MAY.LO = 1200;
LB_APR.LO = 800;
LB_MAY.LO = 800;

*       Power plant capacities:

PA_MAR.UP = 60000;
PA_APR.UP = 60000;
PB_MAR.UP = 35000;
PB_APR.UP = 35000;
```

Equations – Scalar Model (cont.)

```
equations       revenuedef, sales_mar, sales_apr,
                alevel_apr, alevel_may,
                blevel_apr, blevel_may;

*       Unlimited quantities of excess power can be sold for
*       CHF 3.50 per MWH.

*       Revenue therefore equals the sum of high and low price
*       revenue:

revenuedef..    REVENUE =E= 5*(EH_MAR+EH_APR) + 3.5*(EL_MAR+EL_APR);

*       Total sales in March and April equal generation:

sales_mar..     EH_MAR + EL_MAR =e= PA_MAR + PB_MAR;
sales_apr..     EH_APR + EL_APR =e= PA_APR + PB_APR;

*       Level in period t equals level in t-1 plus net inflows:

alevel_apr..    LA_APR =E= LA_MAR + 200 - SA_MAR - PA_MAR / 400;
alevel_may..    LA_MAY =E= LA_APR + 130 - SA_APR - PA_APR / 400;

blevel_apr..    LB_APR =E= LB_MAR + 40 + SA_MAR - SB_MAR - PB_MAR / 200;
blevel_may..    LB_MAY =E= LB_APR + 15 + SA_APR - SB_APR - PB_APR / 200;

model hydro /all/;

solve hydro using lp maximizing revenue;
```

Sets and data: Vector Syntax

```
$title  Hydro Power Planning Model -- Vector Format

set
  r     Reserviors /A, B/,
  t     All time periods /march, april, may/
  tf(t) First period /march/,
  tp(t) Time periods in planning horizon /march, april/,
  i     Types of power /high, low/;

parameter price(i)  Power prices   /high 5.0, low 3.5 /;

Table data summary of relevant data
                A       B
res_cap         2000    1500
minimum         1200    800
march           200     40
april           130     15
level           1900    850
convrate        400     200
pow_cap         60000   35000;

parameter       htr(r)  Heat rate;
htr(r) = data("convrate",r);
```

Variables – Vector Syntax

```
FREE
VARIABLE
        REVENUE   Earnings on sales in March and April;

NONNEGATIVE
VARIABLES
        P(r,t)  Power production
        S(r,t)  Water spilled
        L(r,t)  Reservior level (start of month)
        E(i,tp) Electricity generation;
```

Bounds – Vector Syntax (cont.)

```
*       CHF 5 per MWH up to 50,000 MWH per month:

E.UP("high",tp) = 50000;

*       Storage capacity:

L.UP(r,t) = data("res_cap",r);

*       Minimum reservior levels:

L.LO(r,t) = data("minimum",r);

*       March levels are specified exogenously:

L.FX(r,tf) = data("level",r);

*       Power plant capacities:

P.UP(r,tp) = data("pow_cap",r);
```

Equations – Vector Syntax (cont.)

```
equations       revenuedef, sales, level;

*       Unlimited quantities of excess power
*       can be sold for CHF 3.50 per MWH.

*       Revenue includes high and low price output:

revenuedef..    REVENUE =E= sum((i,tp), price(i)*E(i,tp));

*       Total sales in March and April equal generation:

sales(tp)..     sum(i, E(i,tp)) =e= sum(r, P(r,tp));

*       Accounting for water flows:

level(r,t+1)..  L(r,t+1) =e=

   L(r,t) + data(t,r) + S(r-1,t) - S(r,t) - P(r,t)/htr(r);

model hydro /all/;

solve hydro using lp maximizing revenue;
```