

AMPL Hjälp till mexico-labben

1 Att komma igång

När ni fått filerna är ni redo att köra igång.

De filer som finns är:

```
mex02.mod : modellfilen för mexicolabben
mex02.dat  : datafilen för mexicolabben.
```

Titta gärna på filerna för att förstå hur modellen är uppbyggd.

Ni kan nu starta AMPL med hjälp av kommandot

```
ampl
```

AMPL skall nu starta och ni får upp en prompt som ser ut som

```
ampl:
```

2 AMPL

För att ladda in modellen i AMPL skriver ni `model mex02.mod`; För att även ladda in data skriver ni `data mex02.dat` och för att lösa problemet skriver ni `solve`;

Ni borde nu få utskriften

```
MINOS 5.5: optimal solution found.
40 iterations, objective 21792.34528
```

Om ni blir förvånade över den låga siffran beror denna på att hela problemet är uttryckt i Mton och Mpesos, för att hålla ned storleken på våra konstanter.

Ni kan nu börja undersöka lösningen lite närmare. För att se värdet på en variabel använder man kommandot `display`. För att till exempel se hur mycket råvaror som skickas från gruvorna skriver vi

```
ampl: display raw_trans;
```

(alla kommandon avslutas med `;`) namn på övriga variabler kan grävas fram i modellen.

Vi borde nu få utskriften

```
raw_trans [*,*]
:      Ahmsa   Fundidora   Hylsa   Hylsap   Sicartsa   :=
Cerro_Mer  2.24      0           0       0         0
Coahuila   5.34504    3.648       0       0         1.824
El_Encino  2.38407    6.06593     0       0         0
Laperla    3.47       0           0       0         0
Lastruchas 0          0           0       0         2.85
Penacol    0.381521   0           2.50418 1.65276  0.0263736
;
```

Man kan även få reducerade kostanden för dessa variabler genom att skriva

```
ampl: display raw_trans.rc;
```

På samma sätt kan man få duala variabler genom att skriva till exempel
`ampl: display Raw_Cap.dual;`

vilket ger oss duala värdet för kapacitetsbegränsningarna hos gruvorna.

Slack fås på samma sätt genom att skriva
`ampl: display Raw_Cap.slack;`

Man kan även ha med summationer i de uttryck man visar. För att se total exporterad mängd kan man t.ex skriva

```
ampl: display sum{i in PRODUCER, c in EXPORTS, p in PRODUCT} prod_trans[i,c,p];
```

För att få ut enskilda element kan man även använda index.

```
ampl: display processing['Oven_Red', 'Ahmsa'];
```

ger oss hur mycket vi får ut ur masugnen i Ahmsa.

Om ni gör ändringar i filerna och vill ladda in dem igen så måste ni skriva antingen

```
ampl: reset;
```

, vilket raderar allt, eller

```
ampl: reset data;
```

vilket raderar allt som kommer från datafilen.

Om ni inte gör detta kommer AMPL att klaga när ni läser in filerna igen.

Man kan även ändra konstanter genom kommandot `let`. T.ex

```
ampl: let fixed_cost_raw:=40;
```

ökar den fasta kostnaden för att transportera råmaterial.

3 Mest sannolika missarna

Q: Jag skrev ett kommando men inget hände, när jag sedan skriver in nästa så får jag märkliga fel av typen

```
syntax error  
context: >>>.....
```

A: Du har nog glömt ";" efter förra kommandot. Om inget händer när ett kommando skrivs, titta på prompten. Står det

```
ampl?
```

så väntar ampl fortfarande på slutet på förra kommandot.

Q: Jag få fel av typen

```
invalid subscript min_variabel[j,i] trots att jag borde indexerat rätt.
```

A: Kontrollera ordningen på dina index.