Linear Programming

Harder AMPL example

Dr Thomas Lidbetter
tlidbetter@business.rutgers.edu

Harder example: bits and bobs

- A company produces two products, bits and bobs.
- Bits take 7 hours to make and bobs take 16.25 hours to make.
- Monthly demand:

Demand	June	July
Bits	40	20
Bobs	24	56

Harder example: bits and bobs

- A company produces two products, bits and bobs.
- Bits take 7 hours to make and bobs take 16.25 hours to make.
- Monthly demand:

Demand	June	July
Bits	40	20
Bobs	24	56

- Products can be stored from month to month at a cost of $20/bit, $24/bob
- There are two different work-modes available: normal and overtime.
- Hours available and wage rate for the workmodes:

Work-mode	Hours available	Wage rate ($ per hour)
Normal	1400	8
Overtime	440	12

AMPL implementation

Sets

P	`PRODS`
T	`TIME`
W	`MODES`

Parameters

$d_{p,t}$	`demand`
a_m	`hrs_avail`
r_p	`hrs_req`
c_p	`store_cost`
k_m	`hrly_pay`

Variables

$x_{p,t}$	`produce`
$s_{p,t}$	`store`
$h_{m,t}$	`hrs_paid`

AMPL implementation

Declaring 2-dimensional parameters and variables

```
param demand{PRODS, TIME};
```

Writing the objective function

```
minimize TotalCost:  sum{p in PRODS, t in TIME} store_cost[p] * store[p, t]
                         + sum{m in MODES, t in TIME}hrly_pay[m] * hrs_paid[m, t];
```

AMPL implementation

Writing the constraints

```
SatisfyDemand {t in TIME diff {1}, p in PRODS}:
produce [p, t] + store [p, t-1] - store [p, t] >= demand [p, t];
```

or…

```
SatisfyDemand:  {t in TIME: t > 1, p in PROD}
produce [p, t] + store [p, t-1] - store [p, t] >= demand [p, t];
```

AMPL implementation

Set operations in AMPL

Union:	`A union B`
Intersection:	`A inter B`
Complement:	`A diff B`
Symmetric difference:	`A symdiff B`

AMPL implementation

Entering data for 2-dimensional parameters

```
param demand:        1    2 :=
               bits  40   20
               bobs  24   56;
```