

Modeling and Optimization with OPL

Andreas Popp



3.1 Modeling of
logical expressions

3.2 Decision
dependent
constraints

The Big-M-Method

OPL: modeling of time
periods

Disjunctive Constraints

3.3 OPL: Compact
implementation

3.4 Piecewise
functions

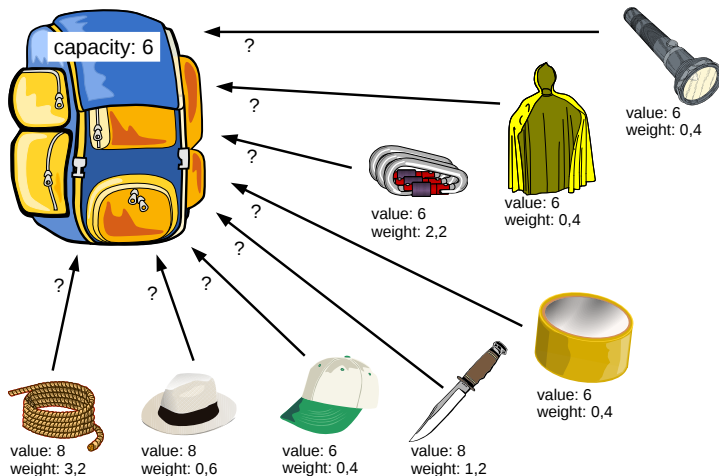
Step functions

Piecewise linear functions

OPL: the `piecewise`
command

3.1 Modeling of logical expressions

Example: Adventure Inc.



3 Methods of
binary
programming

CC-BY-SA
A. Popp

3.1 Modeling of
logical expressions

3.2 Decision
dependent
constraints

The Big-M-Method

OPL: modeling of time
periods

Disjunctive Constraints

3.3 OPL: Compact
implementation

3.4 Piecewise
functions

Step functions

Piecewise linear functions

OPL: the piecewise
command

3.1 Modeling of logical expressions

OPL: modeling of time periods

Step functions

OPL: the piecewise command

Truth table in numerical representation

Example: Let x_1 and x_2 be binary decision variables of a knapsack problem, representing items I_1 and I_2 .

$\neg I_1$: Get the value of I_1 not being packed.

► $1 - x_1$

$l_1 \wedge l_2$: Both l_1 and l_2 must be packed.

► $x_1 + x_2 = 2$

$I_1 \vee I_2$: At least one of the items has to be packed.

► $x_1 + x_2 \geq 1$

$\neg(h_1 \wedge h_2)$: At most one of the items may be packed.

► $x_1 + x_2 < 1$

Logical operators in binary optimization models

3 Methods of binary programming

CC-BY-SA
A. Popp

Example: Let x_1 and x_2 be binary decision variables of a knapsack problem, representing items l_1 and l_2 .

$\neg (I_1 \vee I_2)$: None of the items may be packed.

► $x_1 + x_2 = 0$

$I_1 \vee I_2$: Exactly one of the items must be packed.

► $x_1 + x_2 = 1$

$l_1 \Rightarrow l_2$: If l_1 is packed, l_2 must also be packed.

► $x_1 < x_2$

$l_1 \Leftrightarrow l_2$: The decision is identical for both items.

► $X_1 = X_2$

3.1 Modeling of logical expressions

The Big-M-Method

OPL: modeling of time periods

Piecewise linear functions

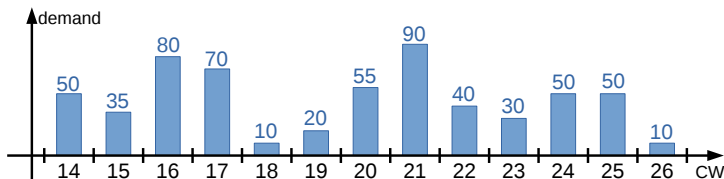
OPL: the piecewise command

3.2 Decision dependent constraints

Example: Lewig Wakuxi

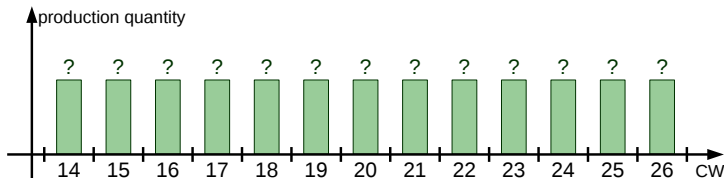
3 Methods of
binary
programming

CC-BY-SA
A. Popp



setup costs per period: 25.000€

inventory costs for one unit from one periode to the next: 100€



3.1 Modeling of
logical expressions

3.2 Decision
dependent
constraints

The Big-M-Method

OPL: modeling of time
periods

Disjunctive Constraints

3.3 OPL: Compact
implementation

3.4 Piecewise
functions

Step functions

Piecewise linear functions

OPL: the piecewise
command

Model: Wagner-Whitin-problem

3 Methods of binary programming

CC-BY-SA
A. Popp

Index sets:

T planning periods $\{t_{min}, \dots, t_{max}\}$

Parameters:

 d_t demand in period $t \in T$ s_t setup costs in period $t \in T$

h_t	inventory costs per item in period $t \in T$
-------	--

 $i_{t_{min}-1}$ initial inventory

M a big number

Decision variables:

x_t	production quantity in period $t \in T$
-------	---

i_t	inventory at the end of period $t \in T$
-------	--

 y_t production decision in period $t \in T$

Model description:

$$\min \sum_{t \in T} s_t \cdot y_t + h_t \cdot i_t$$

$$\text{s.t.} \quad i_t = i_{t-1} + x_t - d_t \quad \forall t \in \mathcal{T} \quad (\text{I})$$

$$x_t \leq M \cdot y_t \quad \forall t \in T \quad (\text{II})$$

$$x_t, i_t \geq 0; \quad y_t \in \{0, 1\} \quad \forall t \in T$$

3.2 Decision dependent constraints

The Big-M-Method

OPL: modeling of time periods

Piecewise linear functions

OPL: the piecewise
command

CC-BY-SA
A. Popp

The Big-M-Method

OPL: modeling of time periods

Disjunctive Constraints

Step functions

OPL: the piecewise command

Constraint of example „Lewig Wakuxi“

$$i_t = i_{t-1} + x_t - d_t \quad \forall t \in T \quad (\text{I})$$

Implementation attempt 1

```
{string} T = {"KW14", "KW15", "KW16", "KW17"};
dvar float+ i[T];
  forall(t in T) i[t] == i[t-1] + x[t] - d[t];
❌ Operator for string - int not available.
```

Constraint of example „Lewig Wakuxi“

$$i_t = i_{t-1} + x_t - d_t \quad \forall t \in T \quad (\text{I})$$

Implementation attempt 2

```
{int} T = {14, 15, 16, 17};
dvar float+ i[T];
  forall(t in T) i[t] == i[t-1] + x[t] - d[t];
❌ Index out of bound for array "i": 13.
```


Constraint of example „Lewig Wakuxi“

$$i_t = i_{t-1} + x_t - d_t \quad \forall t \in T \quad (\text{I})$$

Implementation attempt 4

```
int Tmin = 14;
int Tmax = 17;
range T = Tmin..Tmax;
dvar float+ i[Tmin-1..Tmax];
forall(t in T) i[t] == i[t-1] + x[t] - d[t];
```

OPL: the piecewise command

Decision expressions

3 Methods of
binary
programming

CC-BY-SA
A. Popp

Objective function of the Wagner-Whitin-problem:

```
// objective function
minimize sum(t in T)(s[t]*y[t] + h[t]* i[t]);
```

Structuring with decision expressions:

```
// decision expressions
dexpr float setupCost = sum(t in T)(s[t]*y[t]);
dexpr float inventoryCost = sum(t in T)(h[t]*i[t]);

// objective function
minimize setupCost + inventoryCost;
```

3.1 Modeling of
logical expressions

3.2 Decision
dependent
constraints

The Big-M-Method

OPL: modeling of time
periods

Disjunctive Constraints

3.3 OPL: Compact
implementation

3.4 Piecewise
functions

Step functions

Piecewise linear functions

OPL: the piecewise
command

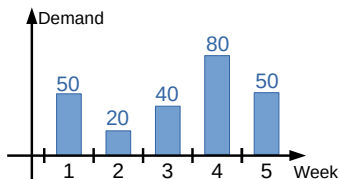
CC-BY-SA
A. Popp

- The Big-M-Method
- OPL: modeling of time periods
- Disjunctive Constraints

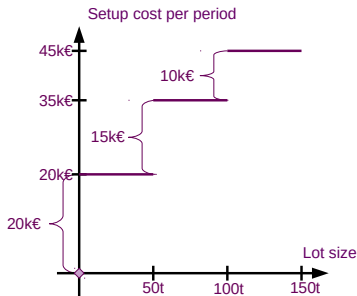
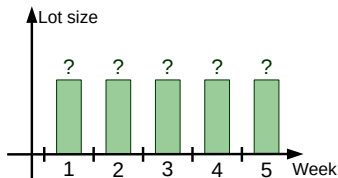
- Step functions
- Piecewise linear functions
- OPL: the `piecewise` command

3.4 Piecewise functions

Example: Lewig Xanxi



Setup costs
Inventory holding cost: 100€



3.1 Modeling of
logical expressions

3.2 Decision
dependent
constraints

The Big-M-Method

OPL: modeling of time
periods

Disjunctive Constraints

3.3 OPL: Compact
implementation

3.4 Piecewise
functions

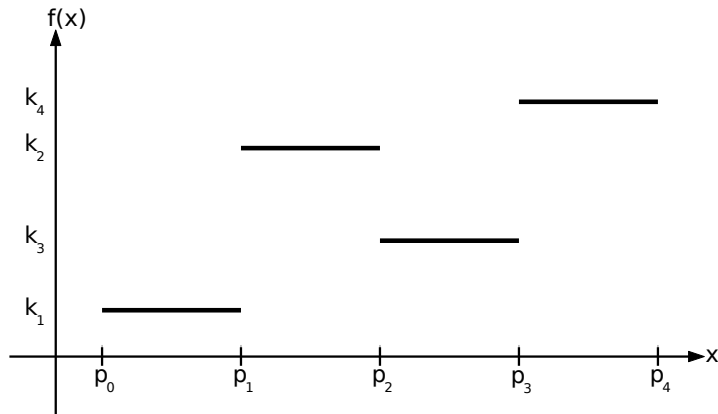
Step functions

Piecewise linear functions

OPL: the piecewise
command

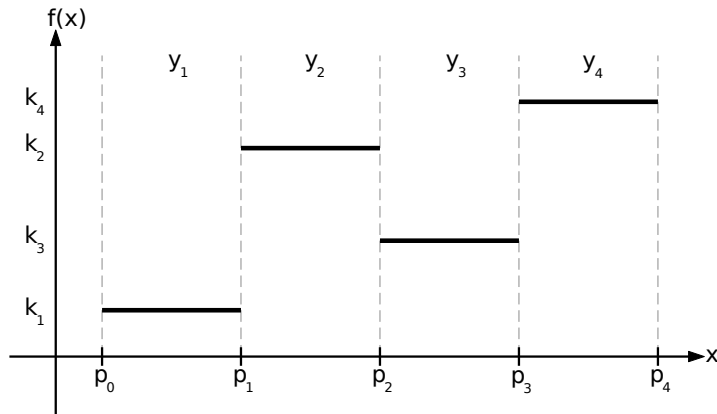
Step functions

Let x be a continuous decision variable:

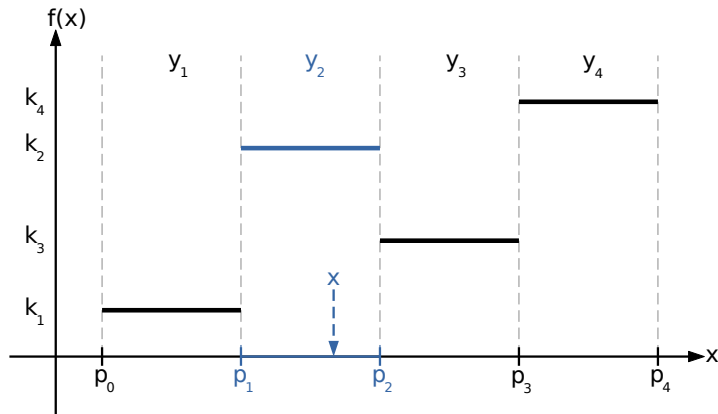


Step functions

Let x be a continuous decision variable:



Let x be a continuous decision variable:



► e.g.: $x = \frac{1}{3} \cdot p_1 + \frac{2}{3} \cdot p_2$

CC-BY-SA
A. Popp

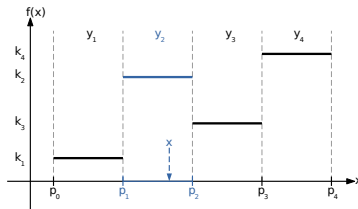
The Big-M-Method

OPL: modeling of time periods

Step functions

OPL: the piecewise command

Decision variable as convex combination of the supporting points



$$x = \sum_{n=0}^N z_n \cdot p_n$$

$$\sum_{n=0}^N z_n = 1$$

$$0 \leq z_n \leq 1 \quad \forall n \in \{1, \dots, N\}$$

3.1 Modeling of logical expressions

3.2 Decision dependent constraints

The Big-M-Method

OPL: modeling of time
periods

Disjunctive Constraints

3.3 OPL: Compact implementation

3.4 Piecewise functions

Step functions

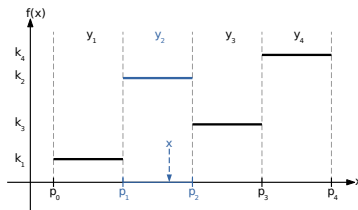
Piecewise linear functions

OPL: the piecewise
command

Choice of the correct interval

3 Methods of
binary
programming

CC-BY-SA
A. Popp



$$\sum_{n=1}^N y_n = 1$$

$$z_0 \leq y_1$$

$$z_n \leq y_n + y_{n+1} \quad \forall n \in \{1, \dots, N-1\}$$

$$z_N \leq y_N$$

3.1 Modeling of
logical expressions

3.2 Decision
dependent
constraints

The Big-M-Method

OPL: modeling of time
periods

Disjunctive Constraints

3.3 OPL: Compact
implementation

3.4 Piecewise
functions

Step functions

Piecewise linear functions

OPL: the piecewise
command

Complete modeling

$$f(x) = \sum_{n=1}^N y_n \cdot k_n$$

$$x = \sum_{n=0}^N z_n \cdot p_n$$

$$\sum_{n=0}^N z_n = 1$$

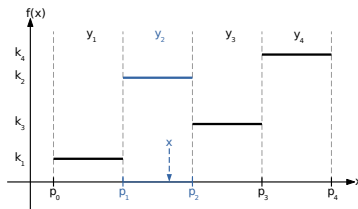
$$0 \leq z_n \leq 1 \quad \forall n \in \{1, \dots, N\}$$

$$\sum_{n=1}^N y_n = 1$$

$$z_0 \leq y_1$$

$$z_n \leq y_n + y_{n+1} \quad \forall n \in \{1, \dots, N-1\}$$

$$z_N \leq y_N$$



3 Methods of
binary
programming

CC-BY-SA
A. Popp

3.1 Modeling of
logical expressions

3.2 Decision
dependent
constraints

The Big-M-Method

OPL: modeling of time
periods

Disjunctive Constraints

3.3 OPL: Compact
implementation

3.4 Piecewise
functions

Step functions

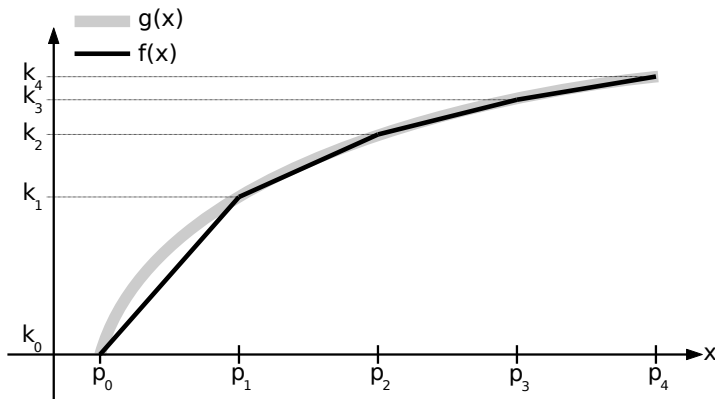
Piecewise linear functions

OPL: the piecewise
command

Piecewise linear functions

3 Methods of
binary
programming

CC-BY-SA
A. Popp



3.1 Modeling of
logical expressions

3.2 Decision
dependent
constraints

The Big-M-Method

OPL: modeling of time
periods

Disjunctive Constraints

3.3 OPL: Compact
implementation

3.4 Piecewise
functions

Step functions

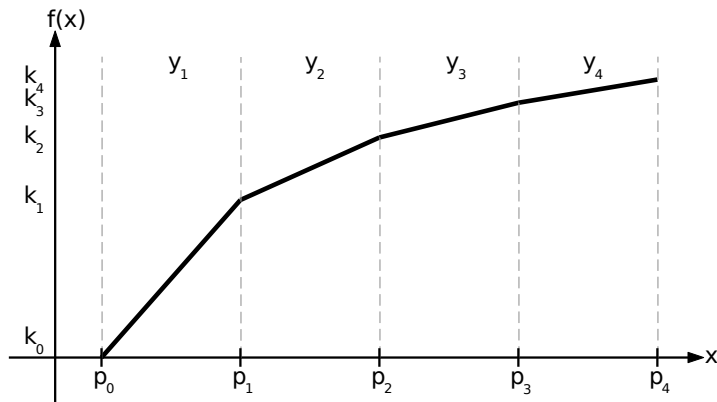
Piecewise linear functions

OPL: the piecewise
command

Piecewise linear functions

3 Methods of
binary
programming

CC-BY-SA
A. Popp



3.1 Modeling of
logical expressions

3.2 Decision
dependent
constraints

The Big-M-Method

OPL: modeling of time
periods

Disjunctive Constraints

3.3 OPL: Compact
implementation

3.4 Piecewise
functions

Step functions

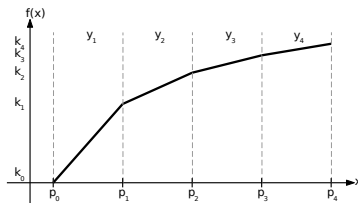
Piecewise linear functions

OPL: the piecewise
command

Function values as convex combination

3 Methods of
binary
programming

CC-BY-SA
A. Popp



$$x = \sum_{n=0}^N z_n \cdot p_n$$

$$f(x) = \sum_{n=0}^N z_n \cdot f(p_n)$$

$$\sum_{n=0}^N z_n = 1$$

$$0 \leq z_n \leq 1 \quad \forall n \in \{1, \dots, N\}$$

3.1 Modeling of
logical expressions

3.2 Decision
dependent
constraints

The Big-M-Method
OPL: modeling of time
periods
Disjunctive Constraints

3.3 OPL: Compact
implementation

3.4 Piecewise
functions

Step functions

Piecewise linear functions

OPL: the piecewise
command

CC-BY-SA
A. Popp

$$f(x) = \sum_{n=0}^N z_n \cdot k_n$$

$$x = \sum_{n=0}^N z_n \cdot p_n$$

$$\sum_{n=0}^N z_n = 1$$

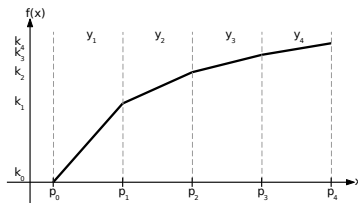
$$0 \leq z_n \leq 1 \quad \forall n \in \{1, \dots, N\}$$

$$\sum_{n=1}^N y_n = 1$$

$$z_0 \leq y_1$$

$$z_n \leq y_n + y_{n+1} \quad \forall n \in \{1, \dots, N-1\}$$

$$z_N \leq y_N$$



The Big-M-Method

OPL: modeling of time periods

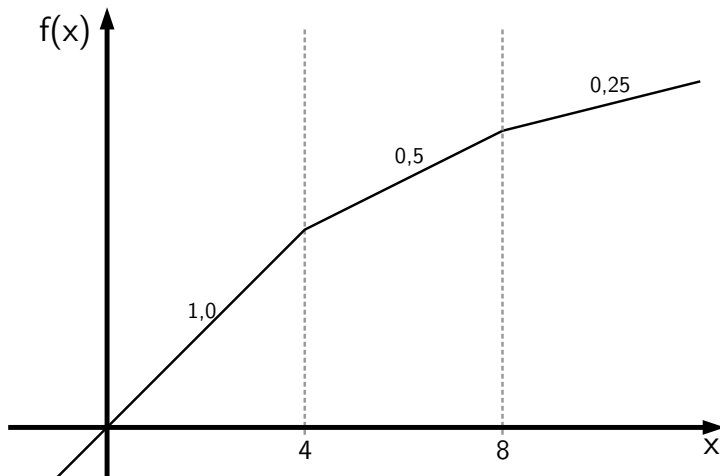
Piecewise linear functions

OPL: the piecewise command

Piecewise linear functions by slope

3 Methods of
binary
programming

CC-BY-SA
A. Popp



3.1 Modeling of
logical expressions

3.2 Decision
dependent
constraints

The Big-M-Method
OPL: modeling of time
periods
Disjunctive Constraints

3.3 OPL: Compact
implementation

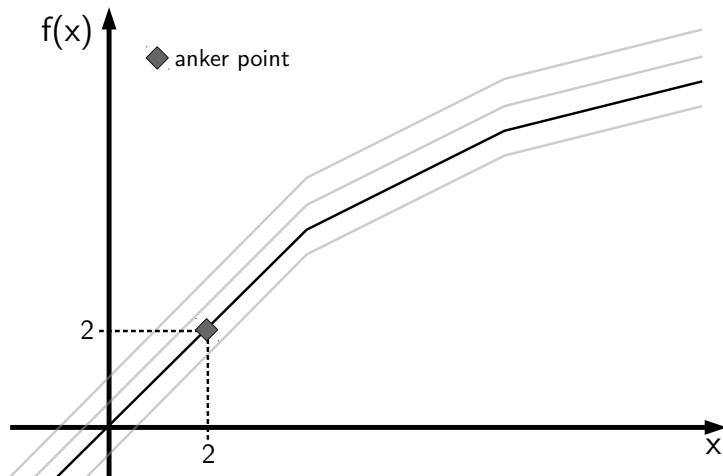
3.4 Piecewise
functions

Step functions
Piecewise linear functions
OPL: the **piecewise**
command

Ankering of piecewise linear functions

3 Methods of
binary
programming

CC-BY-SA
A. Popp



3.1 Modeling of
logical expressions

3.2 Decision
dependent
constraints

The Big-M-Method

OPL: modeling of time
periods

Disjunctive Constraints

3.3 OPL: Compact
implementation

3.4 Piecewise
functions

Step functions

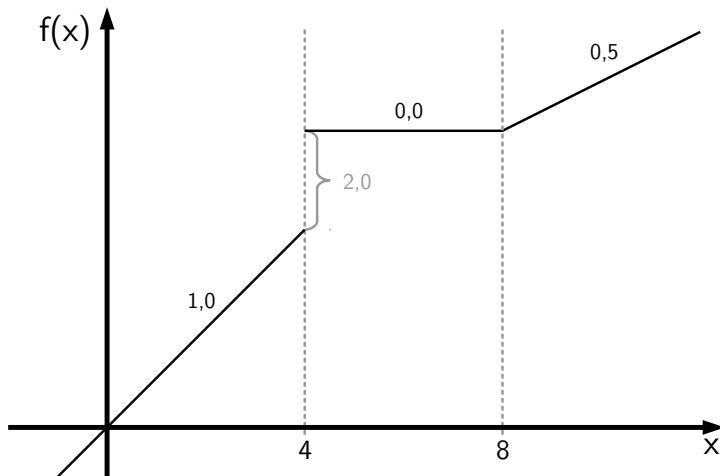
Piecewise linear functions

OPL: the `piecewise`
command

Step functions and general discontinuities

3 Methods of
binary
programming

CC-BY-SA
A. Popp



3.1 Modeling of
logical expressions

3.2 Decision
dependent
constraints

The Big-M-Method

OPL: modeling of time
periods

Disjunctive Constraints

3.3 OPL: Compact
implementation

3.4 Piecewise
functions

Step functions

Piecewise linear functions

OPL: the **piecewise**
command

Step functions and general discontinuities

3 Methods of binary programming

CC-BY-SA
A. Popp

Second slope value at the same supporting point in the piecewise command becomes step value.

Example of figure above

```
int N = 3;
float p[1..N] = [4, 4, 8];
float s[1..N+1] = [1.0, 2.0, 0.0, 0.5];
dvar float+ x;

piecewise(i in 1..N){
    s[i] -> p[i];
    s[N+1]
} x;
```

The Big-M-Method

OPL: modeling of time periods

Piecewise linear functions

OPL: the piecewise command