Comparison of Fortran, Gempack and Sym Notation

1 Algebraic

The value of industry *i*'s output at time *t* is given by the following:

$$ind\_out_{it} = po_{it} \cdot ind\_qty_{it}$$

2 Implementation in Fortran

The time subscript is implicit: sequential periods are loaded and solved one by one (conditional on state and costate variables) by the solution algorithm.

```
integer nind
parameter( NIND = 35 )

real ind(NIND)
real po(NIND)
real ind_qty(NIND)

do ind=1,NIND
     ind_out(ind) = po(ind)*ind_qty(ind)
enddo
```

3 Implementation in Gempack

Time is a special intertemporal set.

```
set nind ( ind1-ind35 );
set (intertemporal) alltime maximum size 8 ( p[0]-p[7] );

variable (all,i,nind) (all,t,alltime) slack13(i,t) ;
variable (all,i,nind) (all,j,alltime) ind_out(i,t) ;
variable (all,i,nind) (all,j,alltime) ind_qty(i,t) ;
variable (all,i,nind) (all,j,alltime) po(i,t) ;

equation output_eq1 (all,i,nind) (all,t,alltime)
     slack13(i,t) = - ind_out(i,t) + po(i,t)*ind_qty(i,t) ;
```

4 Implementation in Sym

Produces output for multiple back-end solvers including Gempack’s TABLO. Time is an explicit set but is implicit in variable declarations. Variables and parameters are declared over sets but don’t require explicit index variables. Equations impose conformability rules and do not usually require explicit indexing or an explicit “equation” at the beginning of the statement.

```
set nind( ind1-ind35 );
set time( p0-p7 );

variable slack13(nind) ;
variable ind_out(nind) ;
variable ind_qty(nind) ;
variable po(nind) ;

slack13 = - ind_out + po*ind_qty ;
```

Could be extended to handle slack variables implicitly:

```
set nind( ind1-ind35 );
set time( p0-p7 );

variable ind_out(nind) ;
variable ind_qty(nind) ;
variable po(nind) ;

ind_out = po*ind_qty ;
```