



A Brief GAMS Tutorial

L. T. Biegler
Chemical Engineering Department
Carnegie Mellon University
Pittsburgh, PA



GAMS Optimization Background

Problem Statement

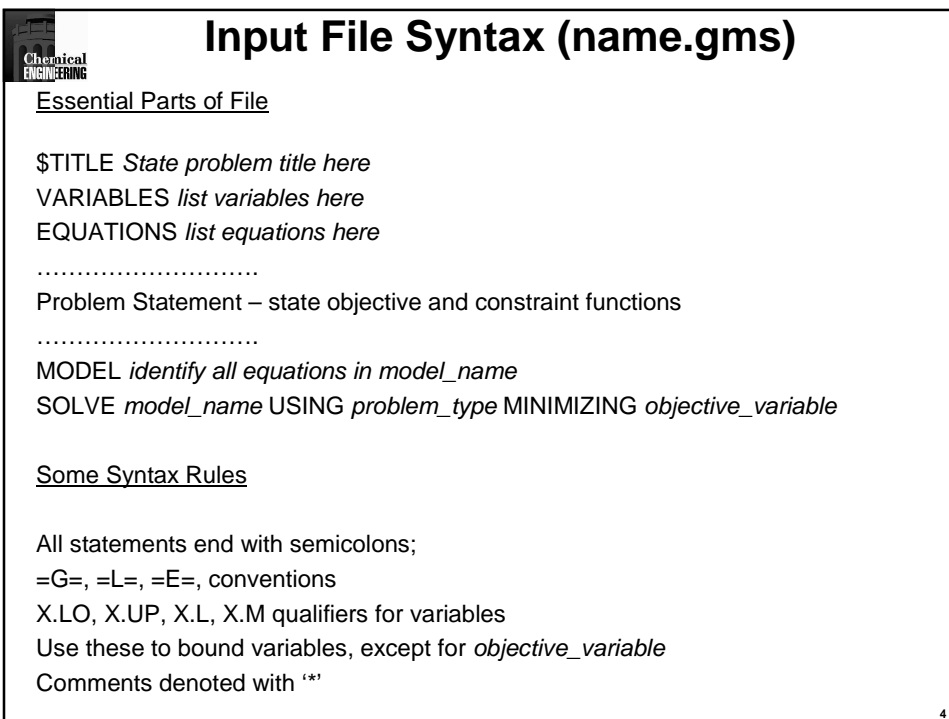
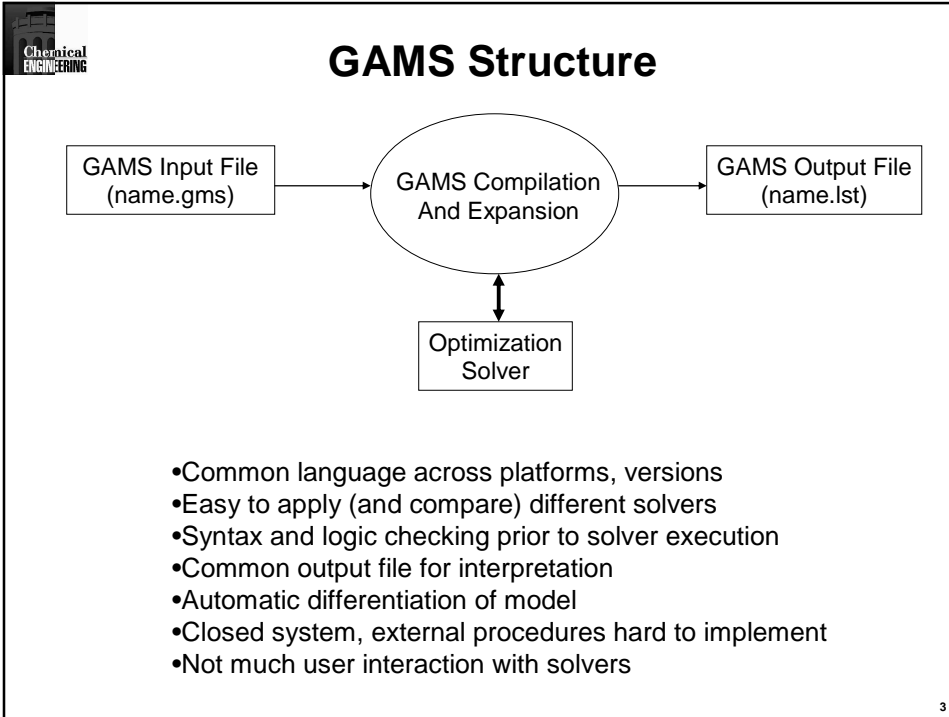
$$\begin{aligned} & \text{Min } F(x, y) \\ & \text{s.t. } h(x, y) = 0 \\ & \quad g(x, y) \leq 0 \\ & \quad x \in \mathcal{R}^{n_x}, y \in \{0, 1\}^{n_y} \end{aligned}$$

Problem Classes

| | |
|--------|------------------------------|
| LP: | CPLEX, XPRESS, ZOOM |
| MILP: | CPLEX, XPRESS, ZOOM |
| NLP: | CONOPT, IPOPT, KNITRO, MINOS |
| MINLP: | DICOPT, BONMIN |

Approach

Leverage powerful existing solvers
Less emphasis on building algorithms
More emphasis on model formulation and refinement





Introductory Example

$$\begin{aligned} & \text{Min } x_1^2 + x_2^2 + x_3^2 \\ & \text{s.t. } x_2 - x_3 \geq 0 \\ & \quad x_1 - x_3 \geq 0 \\ & \quad x_1 - x_2^2 + x_1^2 - 4 = 0 \\ & \quad 0 \leq x_1 \leq 5 \\ & \quad 0 \leq x_2 \leq 3 \\ & \quad 0 \leq x_3 \leq 3 \end{aligned}$$

$$x^0 = [4, 2, 2]$$

Input in test.gms →

See output in test.lst

```
$TITLE Test Problem
$OFFSYMREF
$OFFSYMLIST
* Problem 8.26 in Reklaitis et al (1983)
Variables z;
Positive Variables x1, x2, x3;
Equations con1, con2, con3, obj;

con1.. x2-x3 =G= 0;
con2.. x1-x3 =G= 0;
con3.. x1-x2**2 + x1**2 - 4 =E= 0;
obj.. z =e= sqrt(x1)+sqrt(x2)+sqrt(x3);

* upper bounds
x1.up = 5;
x2.up = 3;
x3.up = 3;

* initial point
x1.l = 4;
x2.l = 2;
x3.l = 2;

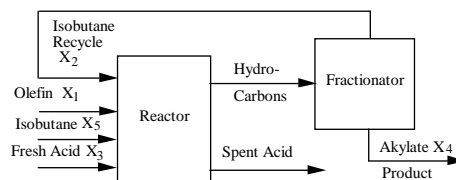
Model test /all/;
Solve test using NLP minimizing z;
```

5



Alkylation Problem

Consider the alkylation process shown below from Bracken & McCormick (1968)



X1 = Olefin feed (barrels per day)
X2 = Isobutane recycle (barrels per day)

X6 = Acid strength (weight percent)
X7 = Motor octane number of alkylate

The alkylation is derived from simple mass balance relationships and regression equations determined from operating data.

The objective function to be maximized is the profit (\$/day)

$$\text{OBJ} = 0.063 * X4 * X7 - 5.04 * X1 - 0.035 * X2 - 10 * X3 - 3.36 * X5$$

6

Chemical ENGINEERING

Alkylation Problem – Input File

```

$ TITLE ALKYLATION PROBLEM FROM GINO USERS
MANUAL
$ OFFSYMXREF
$ OFFSYMLIST
OPTION LIMROW=0;
OPTION LIMCOL=0;

POSITIVE VARIABLES X1,X2,X3,X4,X5,X6,X7,X8,X9,X10;
VARIABLE OBJ;
EQUATIONS E1,E2,E3,E4,E5,E6,E7,E8;

E1..X4=E=X1*(1.12+ .13167*X8-0.0067*X8**2);
E2..X7=E=86.35+1.098*X8-0.038*X8**2+0.325*(X6-89.);
E3..X9=E=35.82-0.222*X10;
E4..X10=E=3*X7-133;
E5..X8*X1=E=X2+X5;
E6..X5=E=1.22*X4-X1;
E7..X6*(X4*X9+1000*X3)=E=98000*X3;

E8..OBJ =E= 0.063*X4*X7-5.04*X1-0.035*X2-10*X3-3.36*X5;

X1.UP = 2000.;
X2.UP = 16000.;
X3.UP = 120.;

X4.UP = 5000.;
X5.UP = 2000.;
X6.LO = 85.;
X6.UP = 93.;
X7.LO = 90;
X7.UP = 95;
X8.LO = 3.;
X8.UP = 12.;
X9.LO = 1.2;
X9.UP = 4.;
X10.LO = 145.;
X10.UP = 162.;
X1.L =1745;
X2.L =12000;
X3.L =110;
X4.L =3048;
X5.L =1974;
X6.L =89.2;
X7.L =92.8;
X8.L =8;
X9.L =3.6;
X10.L =145;

MODEL ALKY/ALL;
SOLVE ALKY USING NLP MAXIMIZING OBJ;

```

7

Chemical ENGINEERING

Refinery Scheduling: Additional Features of GAMS: Sets, Index Notation, External Calcs

```

graph LR
    C1 --> FC[Fuel Chain]
    C2 --> FC
    C3 --> FC
    C4 --> LC[Lube Chain]
    FC --> Gasoline
    FC --> HO[Heating Oil]
    FC --> JF[Jet Fuel]
    FC --> LO[Lube Oil]
    LC --> LO

```

Product Yields for Crudes

| | 1 | 2 | 3 | 4-fuel | 4-lube |
|----------|-----|-----|-----|--------|--------|
| gasoline | 0.6 | 0.5 | 0.3 | 0.4 | 0.4 |
| heat-oil | 0.2 | 0.2 | 0.3 | 0.3 | 0.1 |
| jet-fuel | 0.1 | 0.2 | 0.3 | 0.2 | 0.2 |
| lube-oil | 0.0 | 0.0 | 0.0 | 0.0 | 0.2; |

8



Refinery Scheduling: Problem Data

```
$TITLE Refinery Scheduling
$OFFUPPER
$OFFSYMXREF OFFSYMLIST

*OPTION SOLPRINT = OFF;

* Define index sets
SETS C Crudes /1*3, 4-fuel, 4-lube/
    P Products /gasoline, heat-oil, jet-fuel, lube-oil/

*Define and initialize the problem data
PARAMETER CSUPPLY(C) Crude oil supply kbbl per wk / 1 100.0, 2 100.0, 3 100.0, 4-fuel 200.0,
4-lube 200.0/
CCOST(C) Crude oil costs in $ per bbl /1 15.0, 2 15.0, 3 15.0, 4-fuel 25.0, 4-lube 25.0/
PDEMAND(P) Maximum product demands, kbbl per wk /gasoline 170.0, heat-oil 85.0, jet-fuel
85.0, lube-oil 20.0/
PVALUE(P) Product Values in $ per bbl / gasoline 45.0, heat-oil 30.0, jet-fuel 15.0, lube-oil 60.0/
OCOST(C) Crude operating costs in $ per bbl / 1 5.0, 2 8.5, 3 7.5, 4-fuel 3.0, 4-lube 2.50/
TCOST(C) Total costs: crude plus operating;
TCOST(C) = CCOST(C) + OCOST(C);

TABLE YIELDS(P, C) Yields of products for crudes
    1 2 3 4-fuel 4-lube
gasoline 0.6 0.5 0.3 0.4 0.4
heat-oil 0.2 0.2 0.3 0.3 0.1
jet-fuel 0.1 0.2 0.3 0.2 0.2
lube-oil 0.0 0.0 0.0 0.0 0.2;
```

9



Refinery Scheduling: Problem Statement

```
* Define the optimization variables

VARIABLES X(C) Crude oils used in kbbl per week
    Q(P) Amounts of products produced in kbbl
    X4 Total amount of crude 4 used in kbbl
    PROFIT Total profit from product sales in k$;
POSITIVE VARIABLES X, X4, Q;

* Define constraints and objective function

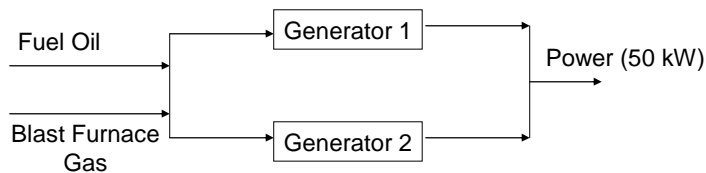
EQUATIONS OBJFUN Objective function to be maximized
    CRUDE4 Total crude 4 usage
    PRODUCTION(P) Amounts of products produced;
OBJFUN.. PROFIT =E= SUM(P, Q(P)*PVALUE(P)) - SUM(C, TCOST(C)*X(C));
PRODUCTION(P).. Q(P) =E= SUM(C, YIELDS(P,C)*X(C));
CRUDE4.. X4 =E= X("4-fuel") + X("4-lube");

* Define upper and lower bounds
* Upper bounds on amounts of product produced from their maximum demands
Q.UP(P) = PDEMAND(P);
* Upper bounds on crude oil usages from their supplies
X.UP(C) = CSUPPLY(C);
X4.UP = CSUPPLY("4-fuel");
* Define model and solve
MODEL SCHEDULE /ALL/;
SOLVE SCHEDULE USING LP MAXIMIZING PROFIT;

DISPLAY X.L, Q.L, PROFIT.L;
```

10

Power Generation via Fuel Oil



Minimize Fuel Oil Consumption

$$\text{Fuel (oil or BFG)} = A_0 + A_1 \cdot \text{Power} + A_2 \cdot (\text{Power})^2$$

Coefficients in the fuel consumption equations

| | A_0 | A_1 | A_2 |
|------------------------|--------|--------|----------|
| gen ₁ (oil) | 1.4609 | .15186 | .00145 |
| gen ₁ (gas) | 1.5742 | .16310 | .001358 |
| gen ₂ (oil) | 0.8008 | .20310 | .000916 |
| gen ₂ (gas) | 0.7266 | .22560 | .000778; |

11

Power Generation via Fuel Oil: Problem Data

```

$TITLE Power Generation via Fuel Oil
$OFFUPPER
$OFFSYM XREF OFFSYMLIST

*OPTION SOLPRINT = OFF;


* Define index sets
SETS G Power Generators /gen1*gen2/
    F Fuels /oil, gas/
    K Constants in Fuel Consumption Equations /0*2/;

* Define and Input the Problem Data
TABLE A(G,F,K) Coefficients in the fuel consumption equations
      0      1      2
gen1.oil  1.4609  .15186  .00145
gen1.gas  1.5742  .16310  .001358
gen2.oil  0.8008  .20310  .000916
gen2.gas  0.7266  .22560  .000778;

PARAMETER PMAX(G) Maximum power outputs of generators /
    GEN1 30.0, GEN2 25.0/;
PARAMETER PMIN(G) Minimum power outputs of generators /
    GEN1 18.0, GEN2 14.0/;
SCALAR GASSUP Maximum supply of BFG in units per h /10.0/
PREQ Total power output required in MW /50.0/;

```

12



Power Generation via Fuel Oil: Problem Statement

```

* Define optimization variables
VARIABLES P(G) Total power output of generators in MW
          X(G, F) Power outputs of generators from specific fuels
          Z(F) Total Amounts of fuel purchased
          OILPUR Total amount of fuel oil purchased;
POSITIVE VARIABLES P, X, Z;

* Define Objective Function and Constraints
EQUATIONS TPOWER Required power must be generated
          PWR(G) Power generated by individual generators
          OILUSE Amount of oil purchased to be minimized
          FUELUSE(F) Fuel usage must not exceed purchase;
TPOWER.. SUM(G, P(G)) =G= PREQ;
PWR(G).. P(G) =E= SUM(F, X(G,F));
FUELUSE(F).. Z(F) =G= SUM((K,G), a(G,F,K)*X(G,F)**(ORD(K)-1));
OILUSE.. OILPUR =E= Z("OIL");


* Impose Bounds and Initialize Optimization Variables
* Upper and lower bounds on P from the operating ranges
P.UP(G) = PMAX(G);
P.LO(G) = PMIN(G);
* Upper bound on BFG consumption from GASSUP
Z.UP("gas") = GASSUP;
* Specify initial values for power outputs
P.L(G) = .5*(PMAX(G)+PMIN(G));

* Define model and solve
MODEL FUELOIL /all/;
SOLVE FUELOIL USING NLP MINIMIZING OILPUR;

DISPLAY X.L, P.L, Z.L, OILPUR.L;

```

13



Suggested GAMS Exercises

- Refinery Scheduling – make product demands lower bound requirements
 - Maximize profit
 - Minimize operating cost
- Fuel Oil – Restrict fuel oil supply to 10 ton/h, purchase BFG.
 - Minimize BFG purchased
 - Minimize BFG and Fuel Oil purchased
- Alkylation – Add relaxation variables between bounds and reformulate problem

14



Dynamic Optimization Problem

$$\begin{aligned} \min \quad & \psi(z(t), y(t), u(t), p, t_f) \\ \text{s.t.} \quad & \frac{dz(t)}{dt} = F(z(t), y(t), u(t), t, p) \\ & G(z(t), y(t), u(t), t, p) = 0 \\ & z^o = z(0) \\ & z^l \leq z(t) \leq z^u \\ & y^l \leq y(t) \leq y^u \\ & u^l \leq u(t) \leq u^u \\ & p^l \leq p \leq p^u \end{aligned}$$

t, time
z, differential variables
y, algebraic variables

t_f, final time
u, control variables
p, time independent parameters

Carnegie Mellon



DAE Models in Engineering

Differential Equations

- Conservation Laws (Mass, Energy, Momentum)

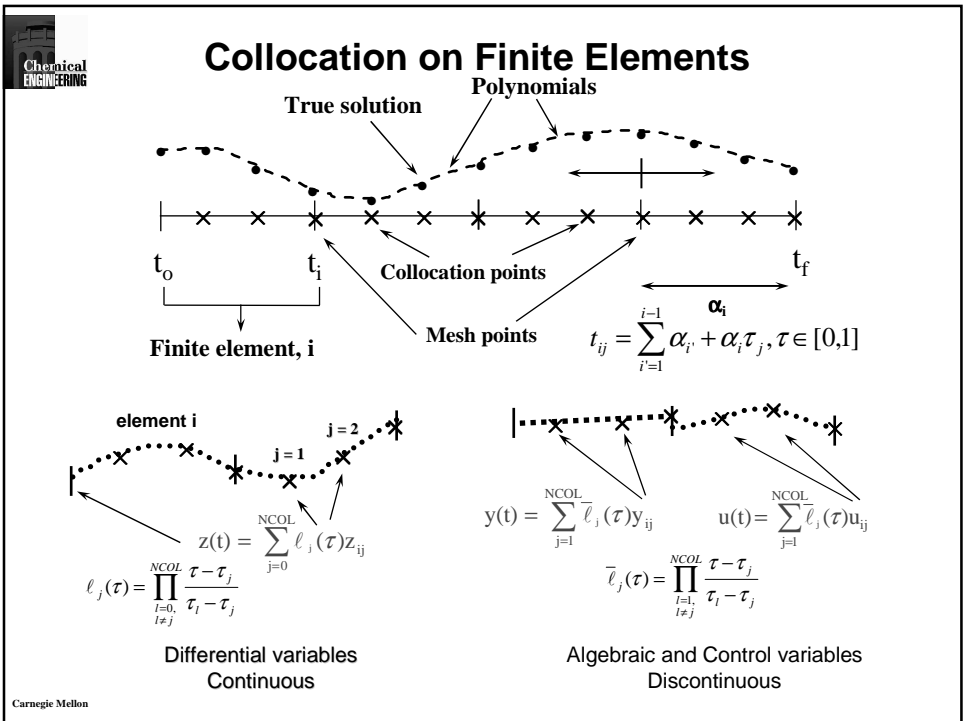
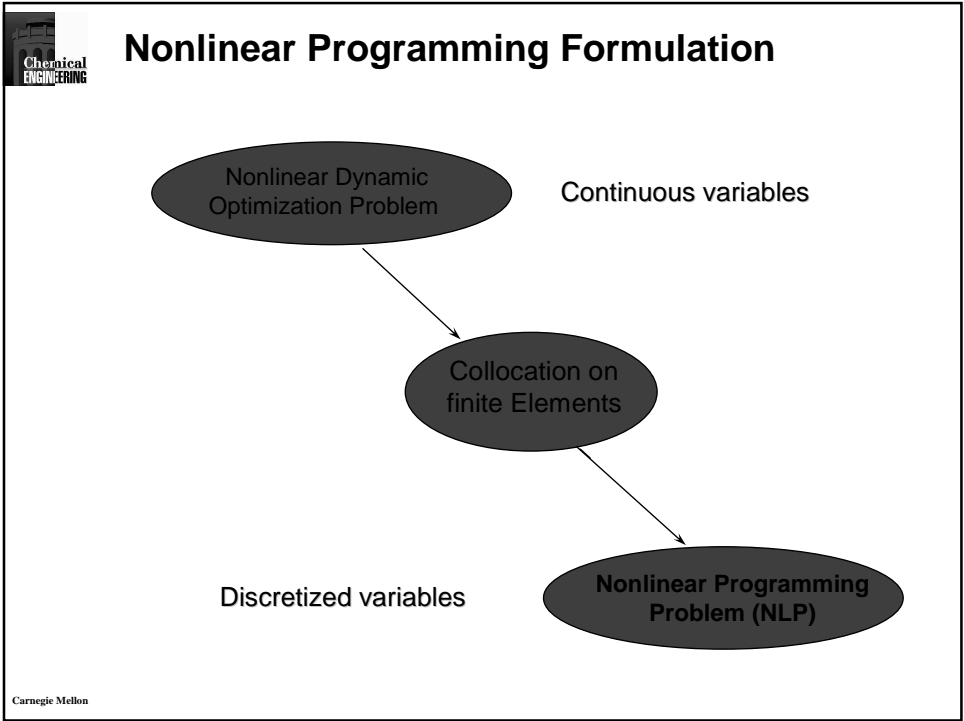
Algebraic Equations


- Constitutive Equations, Equilibrium (physical properties, hydraulics, rate laws)

Characteristics

- Large-scale models – not easily scaled
- Sparse but no regular structure
- Direct solvers widely used
- Coarse-grained decomposition of linear algebra

Carnegie Mellon





Nonlinear Programming Problem

$$\min \psi(z_{i,j}, y_{i,j}, u_{i,j}, p, t_f)$$

$$\text{s.t. } \sum_{k=0}^{NCOL} \ell_k(\tau_j) z_{ik} = \alpha_i F(z_{ij}, y_{ij}, u_{ij}, p)$$

$$G(z_{i,j}, y_{i,j}, u_{i,j}, p) = 0$$

$$z_{i+1,0} = \sum_{j=0}^{NCOL} \ell_j(1) z_{ij}$$

$$z_1^o = z(0), z_f = z_{i+1}^o$$

$$z_i^l \leq z_{i,j} \leq z_i^u$$

$$y_{i,j}^l \leq y_{i,j} \leq y_{i,j}^u$$

$$u_{i,j}^l \leq u_{i,j} \leq u_{i,j}^u$$

$$p^l \leq p \leq p^u$$


$$i = 1, \dots, NFE; j = 1, \dots, NCOL$$

$$\min_{x \in \mathcal{X}^n} f(x)$$

$$\text{s.t. } c(x) = 0$$

$$x^L \leq x \leq x^u$$

Carnegie Mellon



Example: Car Problem

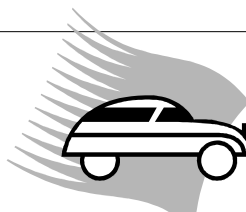
$$\text{Min } t_f$$

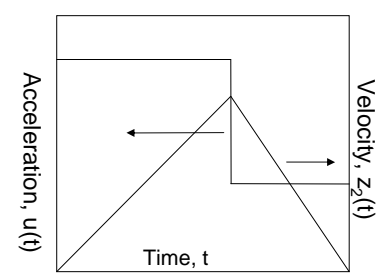
$$\text{s.t. } z_1' = z_2$$

$$z_2' = u$$

$z_2 \leq z_{\max}$

$$-2 \leq u \leq 1$$

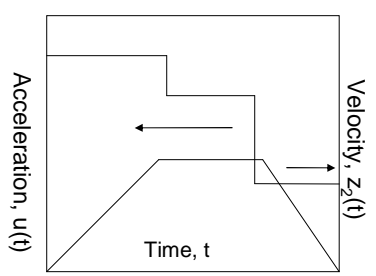




Acceleration, $u(t)$

Velocity, $z_2(t)$

Time, t




Acceleration, $u(t)$

Velocity, $z_2(t)$

Time, t


Carnegie Mellon



Dynamic Optimization: Methods of Solution

- GAMS – requires complex set notation and conditional statements, general formulation with all collocation equations
- AMPL – similar to GAMS, more elegant set notation and conditional statements, general formulation with all collocation equations
- DynoPC – requires only statement of model, data-driven program in Windows, uses IPOPT and ADOL-C

Carnegie Mellon



Dynamic Optimization: GAMS Define Sets and Conditional Sets

```

• TITLE The Car Problem: Minimum Time for a Fixed
Distance
• SOFFUPPER
• SOFFSYMXREF OFFSYMLIST
• SOFFDIGIT

• * This program is the GAMS formulation of the car problem
• *****
• SETS K equation # (max 10) /K1*K10/
I finite elements # (max 20) /I1*I20/
J collocation coeff. # /J1*J5/
COL # possible coll pt (max 3) /C1*C3/

• ALIAS (K,KP)
(JJP,JJS);
• *****
• SCALARS NK actual # of equations /2/
NFE actual # of FE used /2/
NCOL actual # coll. pt used /2/;
• SCALAR NCOF equal to ncol+1;
NCOF = ncol+1;
• SCALAR NCOT equal to ncol+2;
NCOT = ncol+2;
• *****
• ABORT $ (nk GT 10) "Error in defining NK (max 10)", nk
;
• ABORT $ (nfe GT 20) "Error in defining NFE (max 20)",
nfe ;
• ABORT $ (ncol GT 3) "Error in defining NCOL (max 3)",
*****
• SET SXCOL(k,i,jp) actual dim of coll. coeff. (XCOL);
SXCOL(k,i,jp) = YES $ ( (ORD(k) LE nk) $ (ORD(i) LE
nfe)
$ (ORD(jp) LE ncof) );
• SET SU(i,j) actual dim of control variable;
SU(i,j) = YES $ ( (ORD(i) LE nfe) $ (ORD(j) GT 1)
$ (ORD(j) LE ncof) );
• SET SPHIPR(j,jp) actual dim of PHIPR;
SPHIPR(j,jp) = YES $ ( (ORD(j) GT 1) $ (ORD(j) LE
ncot)
$ (ORD(jp) LE ncof) );
• SET SDPHI(jp) actual dim of dominator of PHI;
SDPHI(jp) = YES $ ( (ORD(jp) LE ncof) );
• SET SRES(i,j) actual dim of residual eq;
SRES(i,j) = YES $ ( (ORD(i) LE nfe) $ (ORD(j) GT 1)
$ (ORD(j) LE ncof) );
• SET SERR(i,j) actual dim of error eq;
SERR(i,j) = YES $ ( (ORD(i) LE nfe)
$ (ORD(j) EQ ncol) );
• SET SALF(i) actual dim of alpha;
SALF(i) = YES $ (ORD(i) LE nfe);
• SET SUPRO(i) dim of contol profile;
SUPRO(i) = YES $ (ORD(i) LE nfe);
• SET SXEND(k,i) end condition for state variables;
SXEND(k,i) = YES $ ( (ORD(k) LE nk) $ (ORD(i) EQ
nfe) );
• SET SCONT(k,i) actual dim of continuity eq;
SCONT(k,i) = YES $ ( (ORD(k) LE nk) $ (ORD(i) GT
1)
$ (ORD(i) LE nfe) );
• *****

```

Carnegie Mellon



Dynamic Optimization: GAMS Define Problem Data

```

*
* PARAMETERS TAU(jp) tau at specified ncol
* PHIPR(j,jp) 1-st deriv of phi
* DPHI(jp)   dominator of phi ;
*
* -----
*
* TABLE GENTAU(col,jp) the roots of Lagrange polyn.
*
*      J2      J3
* C1      .5      1.
* C2      .211324865405187      .788675134594813
* C3      .1127016653792585      .5
*
*      +      J4      J5
* C2      1.
* C3      .8872983346207415      1. ;
*
* -----
*
* Assign tau according to the specified NCOL
*
* TAU(jp) $ ( ORD(jp) LE ncol) =
* gentau('C1',jp) $(ncol EQ 1) +
* gentau('C2',jp) $(ncol EQ 2) +
* gentau('C3',jp) $(ncol EQ 3) ;
*
* -----
*
* Calculate DPHI (needed for calculating PHIPR)
*
* DPHI(JP) $ SDPHI(JP) = PROD(J $ ( (ORD(J) LE ncol) $
* (ORD(J) NE ORD(JP)) ),
* (TAU(JP) - TAU(J)) ) ;
*
* -----
*
* Calculate PHIPR (1-st derivative of PHI)
*
* PHIPR(J,JP) $ SPHIPR(J,JP)
* = SUM (JS $ ( (ORD(JS) LE ncol)
* $ (ORD(JS) NE ORD(JP)) ),
* PROD(JJ $ ( (ORD(JJ) LE ncol)
* $ (ORD(JJ) NE ORD(JP))
* $ (ORD(JJ) NE ORD(JS)) ),
* (TAU(J) - TAU(JJ)) ) / DPHI(JP) ;
*
* *****

```

Carnegie Mellon




Dynamic Optimization: GAMS Define Variables and Equations

```

*
* *****
*
* VARIABLES XCOL(k,i,jp) collocation coefficients
* XEND(k,i) state variable at the end condition
* ALPHA(i) time at i-th finite element
* U(i,j) control variables
* OBJ objective function ;
*
* *****
*
* EQUATIONS ERES1(i,j) residual equations for eq 1
* ERES2(i,j) residual equations for eq 2
* ECONT(k,i) continuity equations
* EXEND(k,i) end conditions
* EERR1L(i,j) lower bounds for error control eq 1
* EERR1U(i,j) upper bounds for error control eq
1
* EERR2L(i,j) lower bounds for error control eq 2
* EERR2U(i,j) upper bounds for error control eq
2
* EULO(i) lower bounds for control profile
* EUUP(i) upper bounds for control profile
* EOBJ objective function ;
*
* -----
*
* residual equations for eq 1
*
* ERES1(i,j) $ SRES(i,j) ..
*
* SUM(jp $ (ORD(jp) LE ncol), XCOL('K1',i,jp)*PHIPR(j,jp)) -
* ALPHA(i) *
* ( XCOL('K2',i,j) ) ) =E= 0 ;
*
* residual equations for eq 2
*
* ERES2(i,j) $ SRES(i,j) ..
*
* SUM(jp $ (ORD(jp) LE ncol), XCOL('K2',i,jp)*PHIPR(j,jp)) -
* ALPHA(i) *
* ( U(i,j) ) ) =E= 0 ;
*
* continuity equations
*
* ECONT(k,i) $ SCONT(k,i) ..
*
* XCOL(k,i,'J1') =E= SUM(j $ (ORD(j) LE ncol), XCOL(k,i-1,j)*
* PROD(jp $ (ORD(jp) NE ORD(j) AND ORD(jp) LE
* ncol),
* (( (1.0 - tau(jp))/
* (tau(j) - tau(jp)) ) ) ) ;
*
* end condition equations
*
* EXEND(k,i) $ SXEND(k,i) ..
*
* XEND(k,i) =E= SUM(j $ (ORD(j) LE ncol), XCOL(k,i,j)*
* PROD(jp $ (ORD(jp) NE ORD(j) AND ORD(jp) LE
* ncol),
* (( (1.0 - tau(jp))/
* (tau(j) - tau(jp)) ) ) ) ;

```

Carnegie Mellon




Dynamic Optimization: GAMS

Define Error Equations (optional)

- * error equations for eq 1 - lower bounds
- EERR1L(i,j) \$ SERR(i,j) ..
- SUM(jp \$ (ORD(jp) LE ncof), XCOL('K1'.i,jp)*PHIPR(j,jp)) -
- ALPHA(i) *SUM(jp \$ (ORD(jp) LE ncof), XCOL('K2'.i,jp)*
- PROD(js \$ (ORD(js) NE ORD(jp) AND ORD(js) LE ncof),
- ((1.0 - tau(js))/(tau(jp) - tau(js))))
- =G= - 1.0E-2 ;
- * upper bounds
- EERR1U(i,j) \$ SERR(i,j) ..
- SUM(jp \$ (ORD(jp) LE ncof), XCOL('K1'.i,jp)*PHIPR(j,jp)) -
- ALPHA(i) *SUM(jp \$ (ORD(jp) LE ncof), XCOL('K2'.i,jp)*
- PROD(js \$ (ORD(js) NE ORD(jp) AND ORD(js) LE ncof),
- ((1.0 - tau(js))/(tau(jp) - tau(js))))
- =L= 1.0E-2 ;
- * error equations for eq 2 - lower bounds
- EERR2L(i,j) \$ SERR(i,j) ..
- SUM(jp \$ (ORD(jp) LE ncof), XCOL('K2'.i,jp)*PHIPR(j,jp)) -
- ALPHA(i) *SUM(jp \$ (ORD(jp) GT 1 AND ORD(jp) LE ncof), U(i,jp)*
- PROD(js \$ (ORD(js) NE ORD(jp) AND ORD(js) GT 1
- AND ORD(js) LE ncof), ((1.0 - tau(js))/(tau(jp) - tau(js))))
- =G= - 1.0E-2 ;
- * upper bounds
- EERR2U(i,j) \$ SERR(i,j) ..
- SUM(jp \$ (ORD(jp) LE ncof), XCOL('K2'.i,jp)*PHIPR(j,jp)) -
- ALPHA(i) *SUM(jp \$ (ORD(jp) GT 1 AND ORD(jp) LE ncof),
- U(i,jp)*
- PROD(js \$ (ORD(js) NE ORD(jp) AND ORD(js) GT 1
- AND ORD(js) LE ncof), ((1.0 - tau(js))/(tau(jp) - tau(js))))
- =L= 1.0E-2 ;
- * lower bounds for control profile
- EULO(i) \$ SUPRO(i) ..
- SUM(jp \$ (ORD(jp) GT 1 AND ORD(jp) LE ncof), U(i,jp)*
- PROD(js \$ (ORD(js) NE ORD(jp) AND ORD(js) GT 1
- AND ORD(js) LE ncof),
- ((0 - tau(js))/(tau(jp) - tau(js))))
- =G= - 2 ;
- * upper bounds for control profile
- EUUP(i) \$ SUPRO(i) ..
- SUM(jp \$ (ORD(jp) GT 1 AND ORD(jp) LE ncof), U(i,jp)*
- PROD(js \$ (ORD(js) NE ORD(jp) AND ORD(js) GT 1
- AND ORD(js) LE ncof),
- ((1.0 - tau(js))/(tau(jp) - tau(js))))
- =L= 1 ;
- * the objective function
- E OBJ ..
- OBJ =E= SUM(i \$ SALF(i), ALPHA(i)) ;

Carnegie Mellon



Dynamic Optimization: GAMS

Define Bounds, Initialization, Model...

- * Bounds on variables
- XCOL.LO('K1'.i,jp) \$ SXCOL('K1'.i,jp) = 0 ;
- XCOL.UP('K1'.i,jp) \$ SXCOL('K1'.i,jp) = 300 ;
- XCOL.UP('K2'.i,jp) \$ SXCOL('K2'.i,jp) = 150 ;
- U.LO(i,j) \$ SU(i,j) = - 2.0 ;
- U.UP(i,j) \$ SU(i,j) = 1.0 ;
- ALPHA.LO(i) \$ SALF(i) = 1 ;
- ALPHA.UP(i) \$ SALF(i) = 50 ;
- * initial conditions
- XCOL.FX('K1'.i,'I1'.i) = 0 ;
- XCOL.FX('K2'.i,'I1'.i) = 0 ;
- * final conditions
- XEND.FX('K1'.i) \$ SXEND('K1'.i) = 300 ;
- XEND.FX('K2'.i) \$ SXEND('K2'.i) = 0 ;
- * starting points
- XCOL.L('K1'.i,jp) \$ SXCOL('K1'.i,jp) = 100 ;
- XCOL.L('K1'.i,jp) \$ SXCOL('K2'.i,jp) = 5 ;
- U.L(i,j) \$ SU(i,j) = .1 ;
- ALPHA.L(i) = 10 ;
- *****
- * MODEL PROBLEM1/ALL/ ;
- OPTION NLP = MINOS5 ;
- OPTION LIMCOL = 0 ;
- OPTION LIMROW = 50 ;

Example illustrates:

- GAMS structure/language for complex problem formulations
- AMPL language is much simpler
- DynoPC makes problem setup much easier

Similar input files for

- Catalyst mixing (CATMIX)
- Parameter estimation (PARAM)

Suggested Exercises

- Run all three files
- Change NCOL and NFE
- Modify bounds on controls and states
- Compare with finite elements as parameters, not variables

Carnegie Mellon SOLVE PROBLEM1 USING NLP MINIMIZING OBJ ;