6. Unconstrained Optimization

Unconstrained optimization is the process of searching from a point on a nonlinear curve or surface to find the nearest optimum point of a desired type: maximum, minimum, or saddle point. The function being optimized must be nonlinear or else no optimum point exists. (In linear programming, which involves optimization of linear functions, the optimum is not defined by the functions, which have no optima, but by the application of constraints to restrict them.) A typical unconstrained optimization problem may be stated mathematically as

find $x_1, ..., x_n$

which minimizes $f = f(x_1, ..., x_n)$

The objective function *f is represented by a set of formulas which are, in principle, reducible to a single equation.*

In FC, this unconstrained optimization problem is expressed by a statement of the form:

FIND $x_1, ..., x_n$; **IN** *modelcall*; **BY** *solver*{(*controller*)}; **TO MINIMIZE** *f*

where $x_1, ..., x_n$ are the names of the variables whose values are to be determined and *f* is an objective variable to be minimized. *Modelcall* identifies the model whose execution computes *f* for a given set of values of $x_1, ..., x_n$. The solution process involves successive approximations of the unknowns $x_1, ..., x_n$ by iterative execution of the model until a local optimum value for *f* is found. Because of this, an initial estimate for the values of $x_1, ..., x_n$ must be made before the calculus process is invoked.

6.1. Solver HERA

The nominal solver for unconstrained optimization is called HERA. HERA is an advanced version of Newton's second-order gradient method with special logic to recognize and avoid unwanted extrema during its search process.

The general form of the FIND statement for HERA is as follows:

FIND *unknowns*; **IN** *modelcall*; **BY HERA** {(*controller*)};
{**REPORTING** *auxiliaries;*} {{**WITH**|**AND**} **BOUND**{**S**} *limits;*}
{{**WITH**|**AND**} **SCALE**{**S**} *factors;*} {{**WITH**|**AND**} **FLAG** *signal;*}
TO MAXIMIZE|**MINIMIZE** |**EXTREMIZE** *objective*

where the optional clauses may appear in any order. The common symbols, *unknowns*, *controller*, and *auxiliaries*, are exactly as described for the general FIND statement in Section 2.2.2 The *modelcall* symbol is the model name optionally followed by an

argument list as in the right-hand-side of a CALL statement. This form is not general, but is limited to certain solvers, including AJAX, MARS, HERA and THOR. The three types of parameters, BOUNDS, SCALES, and the FLAG, have special significance to HERA, as described in the solution process below. The final objective clause identifies the function to be optimized and the desired type of extremum.

When HERA is invoked by execution of the FIND statement, it activates derivative evaluation to compute gradients and Hessians with respect to the unknowns. The gradient and Hessian of the objective are used by HERA to select new values of the unknowns during each iteration. The iterative computations of the model continue until the desired optimum is found, until the maximum number of allowed iterations have been performed, or until the model is aborted via an ABORT statement. If the EXTREMIZE option is chosen, the solver searches for the nearest extremum, regardless of whether it is a maximum, minimum, or saddlepoint (maximum in some variables, minimum in others).

Example

Find (x,y) which minimizes

$$F = -3803.84 - 138.08x - 232.92y + 123.08x^2 + 203.64y^2 + 182.25xy$$

using the starting point (1,0.5)

```
PROBLEM .MINF
  X=1 : Y=.5  !INITIAL ESTIMATES
  FIND X,Y; IN FUN; BY HERA; TO MINIMIZE F
END
MODEL   FUN
  F=-3803.84-138.08*X-232.92*Y+123.08*X*X+203.64*Y*Y+182.25*X*Y
END
```

The Solution Process - The Newton procedure for unconstrained optimization employs the following recurrence relation (for the *i*th iteration)

$$\bar{x}_{i+1} = \bar{x}_i - \left[\frac{\partial^2 f}{\partial x^2}\right]_i^{-1} \left[\frac{\partial f}{\partial x}\right]_i$$

in which $\bar{x}$ represents an n-dimensional *unknowns* vector, while $[\partial^2 f/\partial x^2]$ is the Hessian matrix and $[\partial f/\partial x]$ the gradient vector of the *objective*. In the usual interpretation of this process, the recurrence relation is used to locate the zeros of the gradient vector, thus satisfying the necessary conditions for a local extremum.

Another interpretation, one that is easier to visualize, is that Newton's method approximates the objective function at each step by a quadratic function computed by a second order Taylor series. The step change in $\bar{x}$ is exactly the step required to move to the nearest local extremum of the approximating function.

Newton's method has an important disadvantage in that is does not discriminate one

type of extremum from another. HERA was designed to eliminate this disadvantage by computing the step direction and magnitude on the basis of the discrimination procedure described below.

First, the Hessian matrix is diagonalized

$$\begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix} \rightarrow \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & & 0 \\ & \ddots & \\ 0 & & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

by rotating the coordinate system from the independent variables of optimization, $x_1,...,x_n$ to the axes of symmetry $y_1,...,y_n$ of the Taylor approximating function. Graphically this is illustrated in Figure 6-1.

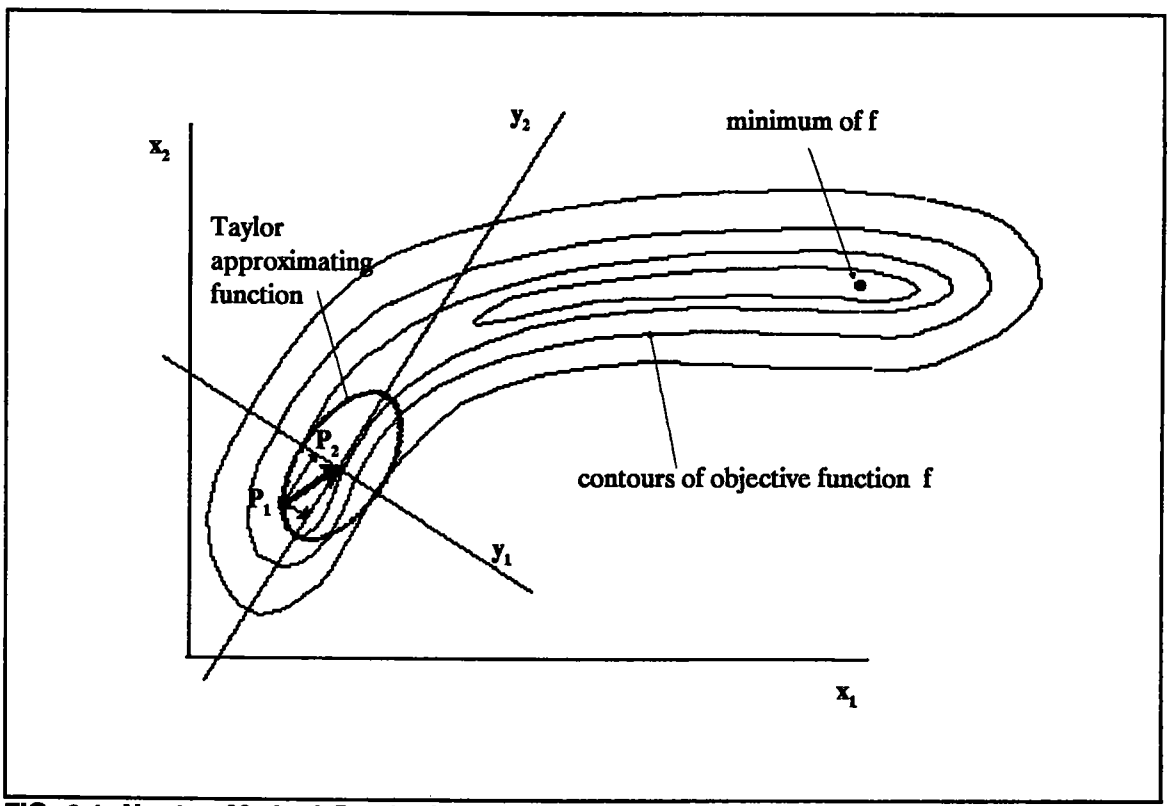

FIG. 6-1 Newton Method Quadratic Approximation

The new axes are the eigenvectors of the Hessian matrix. The approximation of f at the point P_1 is elliptical if the second derivatives

$$\frac{\partial^2 f}{\partial y_1^2} \quad \cdots \quad \frac{\partial^2 f}{\partial y_n^2}$$

(the eigenvalues of the Hessian matrix) all have the same sign. If the eigenvalues are all positive, then the direction into the ellipse is downhill, and the second-order gradient step proceeds from P_1 to P_2, the local minimum of the approximating function. As

illustrated in the figure, this is the result obtained by Newton's method. HERA achieves the same result in a slightly different way, by vector addition of its component steps in the eigenvector direction. Discrimination would become active if HERA had been asked to seek a maximum rather than a minimum. Then the step taken from must P_1 must be an *anti-Newton* step, opposite to the one Newton's method would have selected. HERA selects the direction for each component step on the basis of the signs of the eigenvalues: a positive eigenvalue denotes a step toward a minimum and a negative eigenvalue denotes a step toward a maximum. In the given example, all components must be reversed at P_1 in order to seek a maximum of f. If all of the eigenvalues were not of the same sign (in the neighborhood of a saddle point), HERA would only reverse some of the component directions, so that the resultant step vector will avoid the unwanted saddlepoint.

Step Size Control - Under nominal control conditions (control variable ADJUST = 0), HERA will compute the step component along each eigenvector according to Newton's criterion, i.e.,

$$\Delta y_i = \left(\frac{\partial f}{\partial y_i}\right) \Big/ \left(\frac{\partial^2 f}{\partial y_i^2}\right)$$

The step direction will be *Newton* or *anti-Newton* according to the sign. The step sizes computed in this manner will be effective if the curvature of the objective function is reasonably approximated by a quadratic Taylor series from the point of approximation (P_1) to the optimum point of the quadratic (P_2). However, when an objective function is locally more nonlinear than quadratic, this step size may be too large. As a result, HERA might diverge from the desired optimum point. This may be prevented by the application of bounds from each step component. With proper application of bounds, most unconstrained problems may be solved by HERA.

Bounding - The control of step bounding is specified by the control variable ADJUST, as follows

ADJUST = 0	*No bounding*: Newton step size (Nominal) taken in either Newton or anti-Newton direction
ADJUST = 1	*Variable bounding*: bounds are adjusted from iteration to iteration according to the change in the objective function.
ADJUST = 2	*Fixed bounding*: bounds remain the same throughout HERA execution

Bounds are specified for the independent variables of optimization $x_p,...,x_n$ but are rotated and applied along the step component directions $y_p,...,y_n$. In cases where bounding is active (ADJUST = 1,2), whenever the Newton step size for a component step is less than the applied bound, then the step is unbounded, i.e., the Newton step is used for that component.

Bounds may be specified in either of two ways:

(a) The control variable FRACT may be used to compute bounds for all of the unknowns according to the initial estimates for x_i:

$$b_i = |\text{FRACT}^* x_i|$$

Whenever x_i is initially zero, b_i is set to 1.0.

Warning: If the initial value of x_i is very small and its natural order of magnitude (per solution) is not also very small, then the FRACT option must not be used. If FRACT were used, the variable would be unreasonably constrained.

(b) The WITH BOUNDS clause in the FIND statement may be used to specify an individual bound for each unknown:

WITH BOUNDS $b_1,..., b_n$

where the parameters b_i are variables whose values are to be used as bounds for x_i. As usual for lists of variables in a FIND statement, the list may either be a group of scalar variables or a single vector of the appropriate length. There must be a b_i for each x_i; but if any b_i is zero, then its value is computed by means of FRACT (per above).

Regardless of which approach is used to specify bounds, the values are computed only once, during the initiation stage of the optimization. Thus, the values for variables which specify bounds must be assigned before the FIND statement is executed, and changing them thereafter will have no effect.

Scaling - Scaling may be used to equilibrate the $x_1,..., x_n$ (unrotated axes) for better numerical behavior whenever the magnitude of the *x's* are greatly different. Scale factors may be input as parameters of the FIND statement using the clause

{**WITH** |**AND**} **SCALES** $s_i,..., s_n$;

where s_i are variables whose values are to be used as scale factors for x_i. When this clause is present, a scale factor must be provided for every unknown. As with bounds, the scale factors may be a list of scalar variables or a single vector of an appropriate length. The scale factors are computed only once, during the initiation stage of the optimization. Thus the variables s_i must be assigned values before the FIND statement is executed, and changing them thereafter has no effect on scaling.

If no scales are input, but bounding is applied through the ADJUST = 1 or ADJUST = 2 options, then automatic scaling is applied by setting the scale factors equal to the input or calculated bounds. This is valid only for the usual case where the bounds are proportional to the independent variables. If bounds are input and some are set very high (or low) because the associated independent variable is known to be linear (or highly nonlinear) then automatic scaling should not be used.

Whichever approach is adopted (input or automatic), the scale factors are used effectively as multipliers for the independent variables and associated derivatives during HERA execution. The model computations are unaffected by the scaling. (It is perfectly valid, of course, to scale the model itself by choosing an appropriate system of physical dimensions.)

Small Eigenvalues - Whenever one of the eigenvalues is very small with respect to the largest eigenvalue, either of two interpretations is possible:

(a) The objective function is linear along the associated eigenvector, in which case a large step should be taken in this component direction.

(b) The Hessian matrix is singular at the current point, in which case no step should be taken in this component direction. Step changes in the other components should remove the singularity.

A choice between these alternatives is provided by the control variable STEP.

STEP = 1 selects alternative (a), and the step size is specified by the bound on the variable. In the unbound case (ADJUST = 0) this bound is specified by the value of FRACT times the initial value of the particular independent variable.

STEP = 0 selects alternative (b).

Solution Signal - The FLAG parameter *signal* is a scalar variable which the solver will use to signal the condition of problem solution. After the optimization has terminated, *signal* will have one of the following values:

Signal value	Meaning
0	Convergence was achieved by satisfaction of the active convergence criteria.
-	The model executed an ABORT statement.
+	The maximum number of iterations was performed without achieving convergence.

HERA Controls - The control variables for HERA are as follows:

Variable	Value	Option	Preset Value
ADJUST	0	No bounding	0
	1	Variable bounding	
	2	Fixed bounding	
SUMMARY	1	Print iteration summary	1
	0	No iteration summary	
DETAIL	0	No detailed iteration print	0
	n	Detailed print every nth iteration plus first and last	
STEP	0	No movement along eigenvector corresponding to near zero eigenvalue	1
	1	standard movement to + bound limit	
REMAX		Maximum number of allowed iterations	20
FRACT		Fractional bound to be applied to all independent variables	0.5
CONVERGE	1	Satisfy objective convergence or unknowns convergence	1
	2	Satisfy objective convergence and unknowns convergence	
DELTA		Tolerance for unknowns	10^{-7}
PROGRESS		Relative improvement for objective convergence	10^{-5}
DETOUT	0	Detailed report to PRINTER	+1
	+1	Detailed report to SCROLL	
	-1	Detailed report to Console	
SUMOUT	0	Summary report to PRINTER	-1
	+1	Summary report to SCROLL	
	-1	Summary report to Console	
BREAKIN	0	No interactive breakpoints	0
	n	Breakpoint after every nth iteration	

Convergence Criteria - Convergence criteria for HERA include combinations of the convergence tests *unknowns convergence* and *objective convergence* as specified by the CONVERGE control variable.

The *unknowns convergence* test is satisfied if

$$\max\left(\left|\Delta x_1/x_1\right|, \ldots, \left|\Delta x_n/x_n\right|\right) < \textbf{DELTA}$$

The *objective convergence* test is satisfied if

$$\left|\Delta f/f\right| < \textbf{IMPROVE} \text{ and } \Delta^2 f \text{ is negative}$$

Optimization Summary Report - The optimization summary report is the nominal print report issued from HERA. An example of this report as printed from the problem DIFCRV (in Section 6.2.2) is given in Figure 6-2.

```
---- HERA SUMMARY, INVOKED AT DIFCRVG[11] FOR MODEL CURFIT ----

   CONVERGENCE CONDITION AFTER  6 ITERATIONS
      UNKNOWNS CONVERGED
      OBJECTIVE CRITERION UNSATISFIED
      ALL SPECIFIED CRITERIA SATISFIED

 LOOP NUMBER .........   [INITIAL]          1              2
 UNKNOWNS
   P1                  0.100000E-01  0.274270E-01  0.429687E-01
   P2                  0.500000E-01  0.379046E-01  0.234666E-01
 OBJECTIVE
   ERROR               0.160504E+01  0.775805E+00  0.294929E+00

 LOOP NUMBER .........   [INITIAL]          3              4
 UNKNOWNS
   P1                  0.100000E-01  0.551459E-01  0.682729E-01
   P2                  0.500000E-01  0.609659E-02  0.324918E-02
OBJECTIVE
   ERROR               0.160504E+01  0.505781E-01  0.511270E-02
 LOOP NUMBER .........   [INITIAL]          5              6
 UNKNOWNS
   P1                  0.100000E-01  0.747395E-01  0.760619E-01
   P2                  0.500000E-01  0.370729E-02  0.388164E-02
OBJECTIVE
   ERROR               0.160504E+01  0.274014E-03  0.136268E-03

----END OF LOOP SUMMARY
```

FIG. 6-2 HERA Summary Report

The REPORTING phrase is used to include other variables in the summary report. The statement

FIND THRUST(1), THRUST(2), TBURN(2), TBURN(3);
IN STAGES; BY HERA; REPORTING DLVTOT,TBTOT;
TO MINIMIZE WEIGHT

resulted in the following summary report shown in Figure 6-3

Detailed Iteration Report - The detailed iteration report is issued from HERA whenever the control DETAIL is nonzero. An example of a detailed report (also taken from problem DIFCRV) is shown in Figure 6-4. This report includes the current values of the independent variables, the Hessian of the objective function, the eigenvalues and eigenvectors of the Hessian, and the change in the independent variables computed by HERA. The changes in the independent variables are flagged by "|" if the step was bounded and by "!" if the step was anti-Newton.

```
---- HERA SUMMARY, INVOKED AT ROCKET[15] FOR MODEL STAGES ----

   CONVERGENCE CONDITION AFTER  3 ITERATIONS
      UNKNOWNS NOT CONVERGED
      OBJECTIVE CRITERION SATISFIED
      ALL SPECIFIED CRITERIA SATISFIED

LOOP NUMBER .........    [INITIAL]          1              2
UNKNOWNS
  THRUST   (      1)    0.350000E+03   0.313511E+03   0.327167E+03
  THRUST   (      2)    0.150000E+04   0.127961E+04   0.127827E+04
  TBURN    (      2)    0.100000E+03   0.126287E+03   0.132646E+03
  TBURN    (      3)    0.180000E+03   0.153080E+03   0.151388E+03
OBJECTIVE
  WEIGHT                0.381451E+04   0.378066E+04   0.377866E+04
OTHER VARIABLES
  DLVTOT                0.280000E+05   0.280000E+05   0.280000E+05
  TBTOT                 0.400000E+03   0.400000E+03   0.400000E+03

LOOP NUMBER .........    [INITIAL]          3
UNKNOWNS
  THRUST   (      1)    0.350000E+03   0.328280E+03
  THRUST   (      2)    0.150000E+04   0.127746E+04
  TBURN    (      2)    0.100000E+03   0.133078E+03
  TBURN    (      3)    0.180000E+03   0.151068E+03
OBJECTIVE
  WEIGHT                0.381451E+04   0.377865E+04
OTHER VARIABLES
  DLVTOT                0.280000E+05   0.280000E+05
  TBTOT                 0.400000E+03   0.400000E+03

---END OF LOOP SUMMARY
```

FIG. 6-3 Summary Report with REPORTING variables

6.2. Using Constrained Optimization Solvers

Unconstrained optimization problems may be solved by some of the techniques employed for constrained optimization. In fact, such problems simply represent the limiting case in which constraints are omitted or have no effect upon the problem.

The solvers JOVE, ZEUS and JUPITER described in the following chapter may be used to solve unconstrained optimization problems, while THOR cannot. One disadvantage incurred in using either JOVE, ZEUS or JUPITER is that, in comparison with HERA, far less information is automatically provided concerning the progress to a solution. In fact, these solvers will solve an unconstrained optimization in a single iteration. This is because each of their iterations actually consists of a complete optimization problem. Where such interim information is unnecessary, any of these solvers is suitable. Actually, even this limitation is not entirely valid, since the information which a solver outputs in an iteration report may always be obtained by ordinary FC statements within the model itself.

```
---- HERA ITERATION 1 INVOKED AT DIFCRVG[11] FOR MODEL CURFIT ----

   OBJECTIVE [F] 0.775805E+00 WITH ITERATIVE IMPROVEMENT -0.829234E+00

   INDEPENDENT VARIABLES [X]
     0.274270E-01   0.379046E-01

   HESSIAN MATRIX [D2F/DXDX]
                (   1)          (   2)
   (   1)    0.119909E+04   0.451528E+02
   (   2)    0.451528E+02  -0.257099E+02

   GRADIENT VECTOR [DF/DX]
    -0.274791E+02   0.123231E+02

   EIGENVALUES OF HESSIAN MATRIX
     0.270170E+00  -0.615874E-02

   MATRIX OF EIGENVECTORS
                (   1)          (   2)
   (   1)    0.999323E+00  -0.367904E-01
   (   2)    0.367904E-01   0.999323E+00

   DELTA-X [|=BOUNDED,!=ANTI-NEWTON]
     0.155417E-01| -0.144380E-01!

   DELTA-X / X
     0.566657E+00  -0.380904E+00

   CONVERGENCE CONDITION AFTER  1 ITERATIONS
      UNKNOWNS NOT CONVERGED
      OBJECTIVE CRITERION UNSATISFIED
      SPECIFIED CRITERIA UNSATISFIED

---- END HERA ITERATION 1
```

FIG. 6-4 Detailed Iteration Report for HERA

6.3. Optimization Hierarchies

This section explains the structuring of calculus processes using unconstrained optimization as the outer process for the purpose of addressing higher order applications such as constrained optimization, optimal control, process identification, eigenvalue problems of differential equations, and inverse problems of partial differential equations.

6.3.1. Equality Constraints via Nested Equation Solving

A common mathematical programming problem is

find $x_1, ..., x_n$ which minimizes $f(x_1, ..., x_n)$

subject to the constraints

$$g_1(x_1, ..., x_n) = 0$$

$$\vdots$$

$$g_m(x_1, ..., x_m) = 0$$

In this type of problem, the implicit equations representing the constraints must be solved concurrently with the optimization. One way of solving this type of problem in FC is to nest the solution of implicit equations inside an unconstrained optimization as illustrated below:

```
   :
FIND x1 ,..., xn-m; IN MINF; BY HERA; TO MINIMIZE f
   :
MODEL MINF
      :
   FIND xn-m+1 ,..., xn; IN CONS; BY AJAX; TO MATCH g1 ,..., gm
      :
   f=f(x1 ,..., xn)
END
MODEL CONS
      :
   g1 = g1(x1 ,..., xn)
      :
   gm = gm(x1 ,..., xn)
END
```

This problem has the structure illustrated in Figure 6-5.

When equality constraints are present in an optimization problem, the number of degrees of freedom for optimization is reduced by the number of equality constraints[1]. In the nested problem formulation this is visible, because some of the independent variables, namely $x_{n-m+1}, ..., x_n$ are reduced in status from optimization variables to the unknowns of the implicit equations. Their values are specified when

1 In unconstrained optimization, the number of independent variables n is the number of *degrees of freedom* of optimization, meaning that any variation of the n variables is permitted in the search for an optimum.

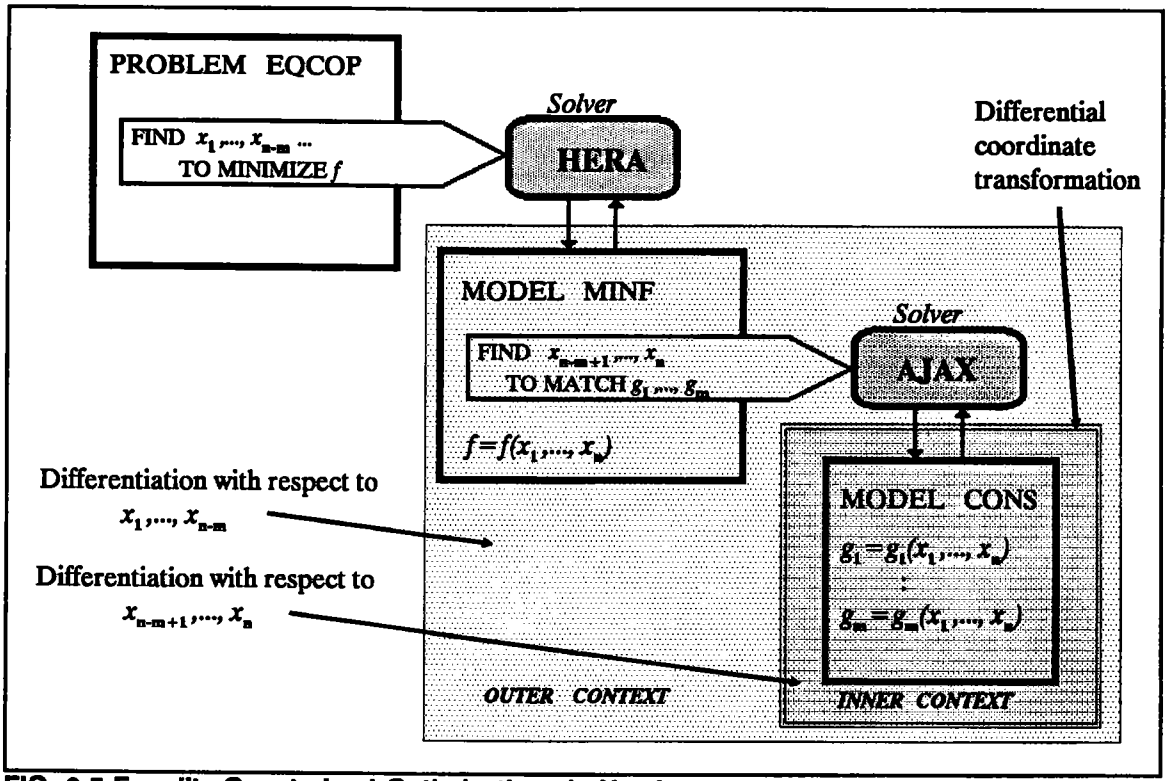

FIG. 6-5 Equality Constrained Optimization via Nesting

the constraints are matched, thus they are dependent variables of the optimization problem. It is usually immaterial which of the independent variables are chosen to be reduced from optimization unknowns to constraint unknowns. The only requirement is that the constraints be dependent on the ones that are chosen.

Differential Coordinate Transformation - The nesting of implicit solution processes is facilitated by a coordinate transformation which first deactivates the differentiation of the outer context, stores its existing derivatives and independent variables ($x_1,\ldots,x_{n-m}$) in an inactive partials environment in the Bucket (See Section 1.2), then activates differentiation with respect to $x_{n-m+1},\ldots,x_n$ of the inner context and executes it. Upon completion of the inner context execution (including convergence to an inner solution), The inactive derivatives are retrieved and used with the active derivatives of the inner context to transform all of the derivatives of inner context with respect to the inner independents $x_{n-m+1},\ldots,x_n$ into equivalent derivatives with respect to the outer independents $x_1,\ldots,x_{n-m}$. This operation converts the inner independent variables into dependent variables of the outer process with correct partial derivatives with respect to the outer independents. The net effect is as if inner computations were *explicit* rather than implicit. This process requires more than one differentiation pass through the inner context following convergence of the inner process, each having different independent variables and/or different order of differentiation.

APPLICATION 6-1: Three-stage Rocket Constrained Design Optimization

(Optimization with Nested Constraint Matching)

Problem Statement

A three-stage liquid rocket vehicle is to be designed to lift a given payload into orbit from the surface of Mars. The propellants have been specified, thus the rocket performance level (specific impulse) is specified. The total velocity increment, and the total burn time are also specified. Each stage is governed by the following equations:

Specific impulse: $I_{sp} = (I_{sp})_{vacuum}\,(1\text{-}X_{IP})$

Propellant weight: $W_{prop} = T\,t_{burn}/I_{sp}$

Stage weight: $W_{stage} = 0.234T + W_{prop} + 1.255W_{prop}^{0.704} + 4$

Structural factor: $S_{FAC} = W_{prop}/W_{stage}$

Total vehicle weight: $W = \sum_{i=1}^{3} W_{stage\,i}$

Mass ratio: $MRi = W/(W\text{-}W_{prop})$

Stage velocity increment: $\Delta V_i = g_c I_{sp} \ln(MR_i)$

Total velocity change: $\Delta V_{tot} = \sum_{i=1}^{3} \Delta V_i$

Total burn time: $t_{Btot} = \sum_{i=1}^{3} t_{burn\,i}$

where

X_{ip} = atmospheric pressure impulse expansion loss

T = thrust (lbf) of stage engine

t_B = burn time of stage engine

It is desired to fix the design conditions for the three-rocket engines in order to minimize total vehicle weight. This involves the determination of six unknowns, the thrust level and burn time for each of the three-stage engines.

Problem Analysis

Since the total velocity increment and total burn time are fixed, we specify the constants DELVIP (ΔV input) and TBIP (t_B input) to define the equality constraints.

$g_1 = \Delta V_{total} - \Delta V_{input}$

$g_2 = t_{Btotal} - t_{B\,input}$

These constraints reduce the optimization degrees of freedom from six to four. Thus, in solving the constraint equations we determine two of the six unknowns. In this case, the choice of unknowns is arbitrary since the constraints are dependent on total vehicle quantities which are, in turn, dependent on all six unknowns. Choosing T_3 and t_{B1} as the implicit equation unknowns, the problem may be stated as

FIND T_1, T_2, t_{B2}, t_{B3} TO MINIMIZE W

and

FIND T_3, t_{B1} TO MATCH g_1, g_2

The program listing is given in Figure 6-6 . The output begins in Figure 6-7, consisting

```
    PROBLEM ROCKET(20000,2000,2000)
      COMMON/ROC/SPI(3),SPIVAC(3),TBURN(3),THRUST(3),XIP(3),WPROP(3),
     *       RATIO(3),WSTAGE(3),STRFAC(3),DELV(3),DLVTOT,TBTOT,WEIGHT
      THRUST(1)=350 : THRUST(2)=1500 : THRUST(3)=4100
      TBURN(1)=110  : TBURN(2)=100   : TBURN(3)=180
      XIP(1)=5D-3   : XIP(2)=0       : XIP(3)=0
      SPIVAC(1)=315 : SPIVAC(2)=315  : SPIVAC(3)=315
      FIND THRUST(1),THRUST(2),TBURN(2),TBURN(3); IN STAGES;
     *   BY HERA; REPORTING DLVTOT,TBTOT; TO MINIMIZE WEIGHT
    END
    MODEL STAGE
  COMMON/ROC/SPI(3),SPIVAC(3),TBURN(3),THRUST(3),XIP(3),WPROP(3),
     *       RATIO(3),WSTAGE(3),STRFAC(3),DELV(3),DLVTOT,TBTOT,WEIGHT
      DIMENSION G(2)
      FIND THRUST(3),TBURN(1); IN EQNS(G); BY AJAX; TO MATCH G
    END

    MODEL EQNS(G)
  COMMON/ROC/SPI(3),SPIVAC(3),TBURN(3),THRUST(3),XIP(3),WPROP(3),
     *       RATIO(3),WSTAGE(3),STRFAC(3),DELV(3),DLVTOT,TBTOT,WEIGHT
      DIMENSION G(2)
      DATA GC,WPAYLD,DELVIP,TBIP/32.174,50,2.8E4,400/
      DLVTOT=0 : TBTOT=0
      WEIGHT=WPAYLD
      DO 10 I=1,3
        SPI(I)=SPIVAC(I)*(1-XIP(I))
        WPROP(I)=THRUST(I)*TBURN(I)/SPI(I)
        WSTAGE(I)=0.0234*THRUST(I)+WPROP(I)+1.255*WPROP(I)**0.704+4
        STRFAC(I)=WPROP(I)/WSTAGE(I)
        WEIGHT=WEIGHT+WSTAGE(I)
        RATIO(I)=WEIGHT/(WEIGHT-WPROP(I))
        DELV(I)=GC*SPI(I)*LOG(RATIO(I))
        DLVTOT=DLVTOT+DELV(I)
        TBTOT=TBTOT+TBURN(I)
10    CONTINUE
      G(1)=DLVTOT-DELVIP  ! Total delta V constraint
      G(2)=TBTOT-TBIP     ! Total burn time constraint
    END
```

FIG. 6-6 Rocket Problem Program Listing

of a series of four AJAX summary reports summarizing the inner solution process (each one corresponding to a single iteration of HERA). The outer solution process is summarized by the HERA summary report given in Figure 6-3.

```
--- AJAX SUMMARY, INVOKED AT STAGES[23] FOR MODEL EQNS ----

   CONVERGENCE CONDITION AFTER  3 ITERATIONS
      UNKNOWNS CONVERGED
      CONSTRAINTS SATISFIED
      ALL SPECIFIED CRITERIA SATISFIED

 LOOP NUMBER .........   [INITIAL]          1              2
 UNKNOWNS
   THRUST   (     3)    0.410000E+04  0.443571E+04  0.445450E+04
   TBURN    (     1)    0.110000E+03  0.120000E+03  0.120000E+03
 CONSTRAINTS
   G        (     1)   -0.490723E+03 -0.213077E+02 -0.422461E-01
   G        (     2)   -0.100000E+02  0.000000E+00  0.000000E+00

 LOOP NUMBER .........   [INITIAL]          3
 UNKNOWNS
   THRUST   (     3)    0.410000E+04  0.445454E+04
   TBURN    (     1)    0.110000E+03  0.120000E+03
 CONSTRAINTS
   G        (     1)   -0.490723E+03 -0.166612E-06
   G        (     2)   -0.100000E+02  0.000000E+00

---END OF LOOP SUMMARY

--- AJAX SUMMARY, INVOKED AT STAGES[23] FOR MODEL EQNS ----

   CONVERGENCE CONDITION AFTER  3 ITERATIONS
      UNKNOWNS CONVERGED
     CONSTRAINTS UNSATISFIED
      ALL SPECIFIED CRITERIA SATISFIED

 LOOP NUMBER .........   [INITIAL]          1              2
 UNKNOWNS
   THRUST   (     3)    0.445454E+04  0.506731E+04  0.510717E+04
   TBURN    (     1)    0.120000E+03  0.120632E+03  0.120632E+03
 CONSTRAINTS
   G        (     1)   -0.675363E+03 -0.387587E+02 -0.142002E+00
   G        (     2)   -0.632439E+00  0.000000E+00  0.000000E+00

 LOOP NUMBER .........   [INITIAL]          3
 UNKNOWNS
   THRUST   (     3)    0.445454E+04  0.510731E+04
   TBURN    (     1)    0.120000E+03  0.120632E+03
CONSTRAINTS
   G        (     1)   -0.675363E+03 -0.191770E-05
   G        (     2)   -0.632439E+00  0.000000E+00

---END OF LOOP SUMMARY

--- AJAX SUMMARY, INVOKED AT STAGES[23] FOR MODEL EQNS ----

   CONVERGENCE CONDITION AFTER  2 ITERATIONS
      UNKNOWNS CONVERGED
      CONSTRAINTS UNSATISFIED
     ALL SPECIFIED CRITERIA SATISFIED

 LOOP NUMBER .........   [INITIAL]          1              2
 UNKNOWNS
   THRUST   (     3)    0.510731E+04  0.510287E+04  0.510421E+04
   TBURN    (     1)    0.120632E+03  0.115965E+03  0.115965E+03
 CONSTRAINTS
   G        (     1)    0.369694E+02 -0.129275E+01 -0.157237E-03
   G        (     2)    0.466703E+01  0.000000E+00  0.000000E+00

---END OF LOOP SUMMARY

--- AJAX SUMMARY, INVOKED AT STAGES[23] FOR MODEL EQNS ----

   CONVERGENCE CONDITION AFTER  2 ITERATIONS
      UNKNOWNS CONVERGED
      CONSTRAINTS SATISFIED
      ALL SPECIFIED CRITERIA SATISFIED

 LOOP NUMBER .........   [INITIAL]          1              2
 UNKNOWNS
   THRUST   (     3)    0.510421E+04  0.511094E+04  0.511094E+04
   TBURN    (     1)    0.115965E+03  0.115854E+03  0.115854E+03
 CONSTRAINTS
   G        (     1)   -0.563950E+01 -0.470836E-02 -0.208092E-08
   G        (     2)    0.111206E+00  0.000000E+00  0.000000E+00
```

FIG. 6-7 Output Listing (Continued In Fig. 6-3)

6.3.2. Optimizing Differential Equations Models

In optimization hierarchies where the inner problem is an initial value problem of ordinary differential equations, the underlying procees of differential arithmetic is used to differentiate the integration prcess as it is executing, propagating partial derivatives of the output of integation (along the integral curves) with respect to variables which are input to the integration process, i.e. initial conditions or parameters of the differential equations. Various kinds of inverse problems of differential equations can be solved with this kind of problem structure. The following applications are typical.

APPLICATION 6-2 Chemical Kinetics Process Identification

(Multipoint Boundary Value, Model-Fitting Optimization)

A chemical reaction process

$$A+B \underset{P_2}{\overset{P_1}{\rightleftharpoons}} C+D$$

can be represented by a system of first order differential equations defining the rate of production of the chemical species A, B, C, and D.

$$dC/dt = P_1AB$$

$$dA/dt = -P_1AB-(0.01+P_2)A = -dC/dt-(0.01+P_2)A$$

$$dB/dt = -P_1AB-0.05BD = -dC/dt-0.05BD$$

$$dD/dt = (0.01+P_2)A-0.05BD = dB/dt-dA/dt$$

where P_1 and P_2 are unknown reaction rate parameters. In an experiment involving the chemical reaction, the concentrations A and C were measured at 10 minute intervals for one hour, resulting in the following data:

Time	t_m	0	10	20	30	40	50	60
	A_m	1	.483	.281	.191	.134	.097	.065
	C_m	0	.419	.563	.629	.666	.689	.708

A program is required to identify the process by selecting values of the parameters P_1 and P_2 for which the integrated concentrations A and C match their respective measured values in a least-squares sense. Specifically, the cumulative error function

$$e(P_1, P_2) = \sum_{i=1}^{6} [A_m(t_i)-A(P_1, P_2, t_i)]^2 + [C_m(t_i) - C(P_1, P_2, t_i)]^2$$

is to be minimized with respect to the parameters P_1 and P_2 where the concentrations $A(P_1, P_2, t_i)$, $C(P_1, P_2, t_i)$, $i=1,...,6$ are determined by integrating the differential equations given initial conditions

$A_0 = 1$ $B_0 = 1.03$ $C_0 = 0$ $D_0 = 0$.

The output of the program will include the values of P_1 and P2.

The program is given in Figure 6-8, and output is shown in figure 6-9.

```
   PROBLEM DIFCRV(5000,1000,1000) ! Chemical Kinetics Process Ident
      COMMON /PARAMS/ P1,P2
      DIMENSION TM(6),CM(6),AM(6)
      DATA TM/10,20,30,40,50,60/
      DATA CM/0.419,0.563,0.629,0.666,0.689,0.708/
      DATA AM/0.483,0.281,0.191,0.134,0.097,0.065/
      DATA NM,A0,B0,C0,D0,DT/6,1.0,1.03,0.0,0.0,2.0/
      DATA B1,B2/0.015,0.015/
      P1=0.01    : P2=0.05 !Parameter starting guesses
      FIND P1,P2; IN CURFIT(TM,CM,AM,A0,B0,C0,D0,DT,NM,ERROR);
   *     BY HERA(SET); WITH BOUNDS B1,B2; TO MINIMIZE ERROR
    END

    CONTROLLER SET(HERA)
      DELTA=1E-2
      DETAIL=1
      ADJUST=2
    END

    MODEL CURFIT(TM,CM,AM,A0,B0,C0,D0,DT,NM,ERROR)
      COMMON /TRAJ/DADT,A,DBDT,B,DCDT,C,DDDT,D,T
      DIMENSION TM(*),CM(*),AM(*)
      A=A0 : B=B0 : C=C0 : D=D0 : T=0     ! Initial conditions
      INITIATE ISIS; FOR DIFEQS;
   *     EQUATIONS DADT/A,DBDT/B,DCDT/C,DDDT/D;
   *     OF T; STEP DT; TO TF
      ERROR=0
      DO 10 I=1,NM
        TF=TM(I)
        INTEGRATE DIFEQS; BY ISIS
        ERROR=ERROR+(AM(I)-A)**2+(CM(I)-C)**2   ! Cumulative error
10    CONTINUE
      TERMINATE DIFEQS
    END

     MODEL DIFEQS
      COMMON /PARAMS/P1,P2 /TRAJ/DADT,A,DBDT,B,DCDT,C,DDDT,D,T
      DCDT=P1*A*B
      DADT=-(DCDT+(.01+P2)*A
      DBDT=-(DCDT+.05*B*D)
      DDDT=DBDT-DADT
    END
```

FIG. 6-8 Program for Process Identification

Bounding is used in this problem to limit the optimization search stepping. Fixed bounding is employed as specified by the control variable ADJUST = 2 in the controller SET.

The optimization model CURFIT contains logic to initialize and integrate the differen-

```
---- HERA SUMMARY, INVOKED AT DIFCRV[11] FOR MODEL CURFIT ----

  CONVERGENCE CONDITION AFTER  6 ITERATIONS
      UNKNOWNS CONVERGED
      OBJECTIVE CRITERION UNSATISFIED
      ALL SPECIFIED CRITERIA SATISFIED

  LOOP NUMBER .........    [INITIAL]          1              2
 UNKNOWNS
    P1                    0.100000E-01  0.274270E-01  0.429687E-01
    P2                    0.500000E-01  0.379046E-01  0.234666E-01
 OBJECTIVE
     ERROR                 0.160504E+01  0.775805E+00 0.294929E+00

  LOOP NUMBER .........    [INITIAL]          3              4
 UNKNOWNS
    P1                    0.100000E-01  0.551459E-01  0.682729E-01
    P2                    0.500000E-01  0.609659E-02  0.324918E-02
 OBJECTIVE
    ERROR                 0.160504E+01  0.505781E-01  0.511270E-02

  LOOP NUMBER .........    [INITIAL]          5              6
 UNKNOWNS
    P1                    0.100000E-01  0.747395E-01  0.760619E-01
    P2
            0.500000E-01  0.370729E-02  0.388164E-02
 OBJECTIVE
    ERROR                 0.160504E+01  0.274014E-03  0.136268E-03

---END OF LOOP SUMMARY
 ELAPSED TIME =    32.41 SECONDS
-------------------------------------------------------------------------

  --- SCROLL TABLE OF CONTENTS
   0.   ---- HERA ITERATION 0 INVOKED AT DIFCRV[11] FOR MODEL CURFIT ----
   1.   ---- HERA ITERATION 1 INVOKED AT DIFCRV[11] FOR MODEL CURFIT ----
   2.   ---- HERA ITERATION 2 INVOKED AT DIFCRV[11] FOR MODEL CURFIT ----
   3.   ---- HERA ITERATION 3 INVOKED AT DIFCRV[11] FOR MODEL CURFIT ----
   4.   ---- HERA ITERATION 4 INVOKED AT DIFCRV[11] FOR MODEL CURFIT ----
   5.   ---- HERA ITERATION 5 INVOKED AT DIFCRV[11] FOR MODEL CURFIT ----
   6.   ---- HERA ITERATION 6 INVOKED AT DIFCRV[11] FOR MODEL CURFIT ----

==== SCROLL SECTION     0

---- HERA ITERATION 0 INVOKED AT DIFCRV[11] FOR MODEL CURFIT ----

    OBJECTIVE [F]  0.160504E+01

    INDEPENDENT VARIABLES [X]
      0.100000E-01   0.500000E-01

    HESSIAN MATRIX [D2F/DXDX]
                  (   1)          (   2)
    (   1)     0.253315E+04   0.475965E+03
    (   2)     0.475965E+03  -0.164538E+02

    GRADIENT VECTOR [DF/DX]
     -0.557362E+02   0.796051E+01

    EIGENVALUES OF HESSIAN MATRIX
      0.589299E+00  -0.230423E-01

    MATRIX OF EIGENVECTORS
                  (   1)          (   2)
    (   1)     0.984081E+00  -0.177719E+00
    (   2)     0.177719E+00   0.984081E+00

    DELTA-X [|=BOUNDED,!=ANTI-NEWTON]
      0.174270E-01| -0.120954E-01!

    DELTA-X / X
      0.174270E+01  -0.241909E+00

    CONVERGENCE CONDITION AFTER  0 ITERATIONS
       UNKNOWNS NOT CONVERGED
       OBJECTIVE CRITERION UNSATISFIED
       SPECIFIED CRITERIA UNSATISFIED

---- END HERA ITERATION 0
```

FIG. 6-9 Process Identification Output Listing Files

tial equations while accumulating the objective function ERROR at time points corresponding to measured values.

Since, in this problem, an integration process is nested inside an optimization process, partial derivatives are propagated through the integration, requiring that the propagating solver ISIS be used for integration.

The output consists of two files, shown in the upper and lower portions of Figure 6-9. First the HERA summary report was printed on the console (nominal output corresponding to SUMOUT=-1). Since DETAIL=1 was specified in the controller block .SET, detailed reports were written on the SCROLL files for each HERA iteration. These files were appended to the console file. The first and second of these files, containing the table of contents and the first iteration report(ITERATION 0 detailing the initial state of the optimization search) are shown in the lower portion of Figure 6-9.

APPLICATION 6-3 Optimal Design And Control

(Optimization of a Two Point Boundary Value Model)

A basic problem in modern control theory is that of minimizing the scalar functional

$$J(y) = \int_0^T g(x,y)\, dt$$

with respect to the vector function y(t), known as the control function, where x(t) is the state function being controlled. The functions x(t) and y(t) are connected by the state differential equation

$$\frac{dx}{dt} = h(x,y), \quad x(0) = C \ .$$

Frequently, the problem is a more general one of minimizing the functional

$$J(y,\alpha) = \int_0^T g(x,y,\alpha)\, dt \ + \phi(\alpha)$$

with respect to y(t) where the state equation is now

$$\frac{dx}{dt} = h(x,y,\alpha), \quad x(0) = C \ .$$

These entail questions of design since α is a design parameter. In most cases, optimal control problems are transformed into two point boundary value problems using principles of the calculus of variations. For example, consider the functional

$$J(y,\alpha) = \frac{1}{2}\int_0^1 (x^2+y^2)\, dt \ + \frac{\alpha^2}{2}$$