7. Constrained Optimization

In the vast majority of practical applications of optimization, there are limitations on the acceptable solutions. Not all values of the unknowns are allowable or even feasible. In a bridge design, beam members cannot exceed certain bearing loads. In budget planning, depreciation rates must follow fixed schedules. There are practical limitations on sources of supply, container size, temperature gradients, etc.

Limitations of this type are expressed as constraints on the optimization variables, and a typical constrained optimization problem may be expressed mathematically as

find $x_1, ..., x_n$ *in the range* $l_i \le x_i \le u_i$ *for* $i = 1,...,n$

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

subject to the constraint functions

$g_j(x_1, ..., x_n) \ge 0 \quad (j = 1, ..., l)$

$h_k(x_1, ..., x_n) = 0 \quad (k = 1, ..., m)$

In contrast with unconstrained optimization problems, the objective function f is represented by a set of formulas which depend upon some, though not necessarily all, of the unknowns $x_1, ..., x_n$. Those unknowns which do not appear in the objective must enter into the constraints in some way or else they are not really unknowns but rather simply auxiliary design variables.

The constant constraints l_i and u_i together with the inequality functions g_j prescribe a feasible region in which solutions are acceptable and are phrased, in a standard fashion, as positive constraints. Essentially, these constraints are imposed to assure that any solution which is reached is physically reasonable and compatible with the problem being analyzed.

The equality constraints h_k serve a different purpose. They express relationships between the unknowns which cannot be simply solved by explicit equations. Constraints of this type might include physical laws, such as conservation of energy, or empirical relationships. Properly defined equality constraints, therefore, are expressed as implicit functions; and, as noted in the previous section, they could be treated separately as a part of a combined calculus process.

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

FIND $x_1, ..., x_n$; **IN** *model*; **BY** *solver*;
WITH LOWER $l_1, ..., l_n$; **AND UPPER** $u_1, ..., u_n$;
HOLDING $g_1, ..., g_l$; **MATCHING** $h_1, ..., h_m$;
TO MINIMIZE f

where $x_1, ..., x_n$ are the names of the unknowns whose values are to be determined and f is an objective variable to be minimized. The *model* identifies a MODEL procedure whose execution computes both f and all of the constraints, g and h, for a given set of values of $x_1, ..., x_n$. In general, there may be any number of constraints and they may take any functional form.

The solution process, like that of unconstrained optimization, involves successive approximation of the unknowns $x_1, ..., x_n$ by iterative execution of the *model* until a local optimum value of f is found. In this case, however, the solution is held within the feasible region by one of a variety of techniques. As usual, an initial estimate must be made for the values of the unknowns before the calculus process is invoked. If this estimate is infeasible, the solution process first attempts to find a feasible starting point, typically employing the same general technique as it will use thereafter.

Specifying Constraints - Constraints have two forms in FC. Constant constraints such as

$$x \geq 0$$

$$y \geq 0$$

$$a \leq x \leq b$$

are specified at the beginning of the optimization process, and may not be changed while it is in progress.

Constraint functions such as

$$e^x + e^y = 1$$

$$xy \leq 100$$

are dependent on the values of the unknowns. Thus they are calculated in the model. Where there are explicit equality relationships between model variables, they do not properly represent constraints. Thus, in a problem with two variables x and y where

$$x + y = 1$$

only one may be correctly viewed as an unknown. Of course, there is no reason to eliminate one of the variables from the model, particularly since it might represent a design parameter of interest. Rather, it should not be specified as an unknowns, and the equation should be treated as an auxiliary calculation rather than a constraint.

Functions - In generating the model functions which represent constraints, standard conditions must be met. All equality constraints are matched to zero, and thus the fourth constraint above must be calculated by

$$h = 1 - e^x - e^y = 0$$

Similarly, all inequality constraints are expressed as positive functions, whence the fifth

constraint is expressed as

$$g = 100 - xy \geq 0$$

For the sake of simplicity, all of these examples involve few variables and can be expressed concisely. There are, however, no limits on the complexity of the relationships; they may require whole series of calculations, even involving calculus processes. The only requirement is that the values finally computed meet the standard mathematical conditions. Indeed, the constraints need not be computed directly from the unknowns at all, but may be expressed as functions of any variables which are dependent upon one or more unknowns.

Care must be exercised to assure that constraints describe a feasible problem, particularly when they are complex and when there are many auxiliary variables. Furthermore, all constraints must depend, directly or indirectly, on at least one unknown.

Constants - Constant constraints may obviously be expressed as "trivial" functions. However, it is far better to express them as limits prior to the start of optimization, since all solvers work more efficiently and exactly in this manner. Either upper or lower limits (or both) may be specified. However, when either type of limit is specified a value must be specified for every unknown.

If the upper and lower limits on any unknown overlap, they are ignored. Thus it is possible to limit some variables and not others by setting the upper and lower limits the same (or overlapping) for those variables which are to remain unlimited.

Algorithm levels - Due to the influence of first-order unconstrained optimization methods and the broad use of penalty-function stratagies, most constrained optimization algorithms are hierarchic combinations of three algorithmic levels. The highest level is usually referred to as the *strategy level*. This is the level at which problems are specified using the FIND statement. Each strategy iteration is a complete solution of a transformed problem, usually an unconstrained optimization problem. Between strategy iterations, the transformed problem is altered according to the strategy algorithm so that the solution to each successive transformed problem is a closer approximation to the constrained optimization problem posed in the FIND statement. The second iterative level generally corresponds to unconstrained optimization (although in some cases constraints are directly addressed at this level as well). It is therefore referred to as the *optimizer level*. Each optimizer iteration selects a search vector in the coordinate system of the unknowns. Thus the model calls at this level inviably generate gradient vectors or Hessian matrices for use in computing the direction vector. Since many optimizers perform a one-dimensional search along the direction vector, seeking either a local extremum or a constraint boundary, the third iterative level is called the *vector or one-dimensional search level*. In most cases, this does not involve differentiation, thus model calls at this level involve only ordinary arithmetic.

7.1. Solver JOVE

The solver JOVE is a sequential unconstrained optimization technique[1]. In this method, a new objective function is constructed from the original one by adding penalty functions involving the constraints. Then this augmented objective is optimized by an unconstrained optimization technique.

The general form of the FIND statement for JOVE is:

FIND *unknowns*; **IN** *model*; **BY JOVE** {(*controller*)};
{**WITH**|**AND**} **LOWER** *floor*;} {**WITH**|**AND**} **UPPER** *ceiling*;}
{**REPORTING** *auxiliaries*;} {**WITH FLAG** *signal*;}
{**HOLDING** *inequalities*;} {**MATCHING** *equalities*;}
TO MINIMIZE|**MAXIMIZE** *objective*

where the optional clauses may appear in any order. The common elements, *unknowns*, *model*, *controller*, and *auxiliaries*, are exactly as described for the general FIND statement in Section 2.2.2. The elements *floor* and *ceiling* are lists containing constant constraints. If present, these lists must correspond in size to the *unknowns* list. The FLAG parameter *signal* permits JOVE to signal the condition of problem solution, as explained in Section 2.2.5. The constraints if any, are specified in the HOLDING and MATCHING clauses. Each prescribes a list of variables whose values represent the constraints. The final objective clause identifies the function to be optimized and the desired type of extremum.

When JOVE is invoked by execution of the FIND statement, it activates derivative evaluation to compute gradients and Hessians with respect to the *unknowns*.

Example

Find x,y which minimizes $f = (x-2)^2 + (y-1)^2$

Subject to $g(x,y) = 1 - x^2/4 - y^2 \geq 0$

$$h(x,y) = e^{xy} - x - 2 = 0$$

starting from the feasible point (-1,0)

```
PROBLEM SIMPLE
 COMMON/EQS/X,Y,G,H,F
 X=-1 : Y=0  ! Initial estimates
 FIND X,Y; IN MODF; BY JOVE;
   HOLDING G; AND MATCHING H;
   TO MINIMIZE F
END
```

1 The theoretical basis for this technique is thoroughly treated in *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, Fiacco, A.V. and McCormick, G.P. (Wiley 1968).

```
MODEL MODF
 COMMON/EQS/X,Y,G,H,F
  G=1-X**2/4 - Y**2
  H=EXP(X*Y) -X -2
  F = (X-2)**2 + (Y-1)**2
END
```

The Solution Process - The essential idea of a penalty function technique is to transform the constrained optimization problem into a sequence of unconstrained problems, for which a variety of methods are very effective. The constraints are incorporated into the objective in appropriate functions and then they are effectively deleted from the problem. The augmented objective is optimized in a sequence of steps in which the contributions of the penalty functions are progressively diminished. In the limit, their contributions become negligible and the optimum of the augmented objective is also that of the original one.

The modified objective constructed by JOVE is

$$P = f - \rho \Sigma \ln g + \Sigma h^2/\rho$$

where f, g, and h are the original objective, the inequality constraints, and the equality constraints, respectively. The parameter ρ is a weighting factor which is decreased for each unconstrained suboptimization until $P \rightarrow f$ and the problem is solved. The method employed in each of the unconstrained optimization steps is to select a direction by a generalized Newton procedure and then to locate an optimum in that direction by a modified Fibonacci search. Upon option, when the problem is believed to be ill-conditioned, the gradient is used in selecting a direction whenever the Hessian matrix of the augmented objective is indefinite (where it has eigenvalues of mixed signs).

JOVE provides acceleration controls which, for well-behaved functions, can significantly speed convergence. Essentially, the technique may be adjusted to begin extrapolating extrema rather than actually calculating them once sufficient steps have been taken.

JOVE Controls - The control variable for JOVE are as follows:[2]

2 In these descriptions, the term "iteration" refers to a complete unconstrained optimization.

Variable	Value	Option	Preset Value
DETAIL	0	No detail iteration point	1
	n	Detailed print every nth iteration plus first and last	
SUMMARY	1	Print iteration summary	1
	0	No iteration summary	
REMAX		Maximum number of allowed iterations	5
STEPSLIM		Limit on the number of steps per iteration Since an iteration amounts to an entire unconstrained optimization, this control is analogous to REMAX for solver HERA. However, if solving a constrained problem, exceeding LIMSTEPS does not terminate the process, it merely halts the current iteration.	100
STEPOUT	n	Provides snapshot prints of the step-by-step solution progress at every nth step of all iterations (about 4-9 lines of critical problem data).	5
	-n	Constraint values are added to the snapshot prints.	
	0	No snapshot prints are generated.	
EVALMAX		Limit on the total number of model evaluations. The optimization process is halted when this limit is exceeded and the result is flagged in the same manner as if maximum iterations were reached. The actual number of evaluations may exceed EVALMAX slightly because of the need to terminate the solution process correctly.	1000
ABORT[3]	0	Delay a model abort until the current iteration is complete.	1
	1	Execute a model abort after the current step in which an ABORT statement was executed.	
SEARCH	1	Use generalized Newton-Raphson method	1
	2	As above, but use the gradient when the Hessian is indefinite.	
SETRHO	1	Set the initial value of ρ to 1	1
	2	Compute an initial value of ρ which minimizes the magnitude of the gradient at the initial point.	
	3	Compute an initial value of ρ which minimizes $(\partial P/\partial x)(\Delta \bar{x})$ at the initial point.	
	n	$(0 < n < 1)$ set value of ρ to n (necessary for restarting a problem from a previously computed point - the appropriate value for ρ is printed in the detailed report.)	

3 This control is only effective when a model abort is requested by an ABORT statement, see Section 2.2.2.

Variable	Value	Option	Preset Value
RATIO		Factor used to reduce ρ between successive unconstrained optimizations: must be >2	16
SPEEDUP	1	Do not accelerate by performing extrapolation.	1
	2	Perform first order extrapolation.	
	3	Perform second order extrapolation.	
ESTIMATE	1	Determine final problem convergence on the basis of the current (zero order) solution.	1
	2	Base convergence upon either the zero order solution or on a first order extrapolation.	
	3	Base convergence upon either the zero order solution or on a second order extrapolation.	
CONVERGE	1	Assume that the problem finally converged when the scaled objective functions of the primal and dual solutions differ by less than ACCURACY.	1
	2	Converge when the penalty contribution of the inequality constraints is less than ACCURACY.	
	3	Converge when the scaled first order estimate of the optimum objective differs from the dual objective by less than ACCURACY.	
ACCURACY		Final convergence tolerance	10^{-4}
PROGRESS		Iteration improvement limit; an unconstrained optimization is terminated if the objective fails to improve by at least this amount.	0
SUBTEST	1	Terminate an iteration (an unconstrained optimization) when $(\partial P/\partial x_I) <$ ZERO for all x_I.	1
	2	Terminate an iteration when $(\partial P/\partial x_I)(\Delta x_I)$ is less than 20% of the improvement in the augmented objective for all x_I.	
	3	Terminate an iteration when $\lvert\partial P/\partial x_I\rvert <$ ZERO for all x_I.	
ZERO		Iteration convergence tolerance	10^{-6}
BREAKIN	0	No interactive breakpoints	0
	n	Breakpoint after each nth iteration	
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	

This rather disconcertingly long list of controls basically provides capabilities for tuning JOVE for difficult problems. For many problems, and for virtually all simple ones, the preset values should be effective. When the need arises to reset controls, the following considerations should be helpful.

When the circumstances permit, computing an estimate for ρ is desirable, rather than picking the arbitrary value of 1. However, this is only possible if there are no equality constraints. Furthermore, the best approximation (SETRHO = 3) is further limited by the requirement that the initial point must be close to some inequality constraint boundary (if any exist).

The preset RATIO works well for convex problems. A slower reduction of ρ may be required for nonconvex problems.

Either of CONVERGE = 1 or 3 are theoretically consistent with the solution technique. The second option ignores the contribution of equality constraints, which may be appropriate whenever such constraints are weakly imposed. Only the preset test is allowed whenever convergence estimation is activated (ESTIMATE $\neq$ 1).

The iteration improvement limit is inactive in the preset condition. When JOVE is being used to obtain a rough estimate at which to begin a refined optimization, the value of PROGRESS might be increased.

The value of ACCURACY essentially bounds the accuracy of the final solution: 10^{-4} assures four figure accuracy. If the unconstrained subproblems are being solved to a high precision (ZERO = 10^{-6}), the final solution will normally have an accuracy substantially better than the upper bound.

Optimization Summary Report - A standard optimization summary report is nominally generated by JOVE. In format and content, it is identical to that produced by HERA (see Figure 6-2). It must be remembered that an iteration, for JOVE, is a complete unconstrained optimization. Thus, when this solver is used for unconstrained optimization, it will converge in a single iteration.

If the initial estimate of the unknowns is not within the feasible region, JOVE searches for a feasible point as a first step. this circumstance is flagged in the first iteration print, if one is requested. In any case, the summary of the first iteration presents the results for the first optimization which began at a feasible point.

Detailed Iteration Report - A detailed iteration report is issued by JOVE whenever the control DETAIL is nonzero. This report actually summarizes an unconstrained suboptimization. This report lists the current value of ρ and the evaluated penalty function gradient vector ∇P. The solution is presented for the original objective, the unknowns, the constraints, and the augmented objective function (the primal penalized objective). The contributions of each class of constraints to the penalty function is also displayed.

```
--- JOVE SUMMARY, INVOKED AT SIMPLEJ[5] FOR MODEL MODF ----

   CONVERGENCE CONDITION AFTER 10 ITERATIONS
      OBJECTIVE CRITERION SATISFIED
     ALL SPECIFIED CRITERIA SATISFIED

LOOP NUMBER .........   [INITIAL]          1              2
UNKNOWNS
   X                  -0.100000E+01  0.115943E+01  0.119551E+01
   Y                   0.000000E+00  0.658668E+00  0.800625E+00
OBJECTIVE
   F                   0.100000E+02  0.823059E+00  0.686951E+00
INEQUALITY CONSTRAINTS
   G                   0.750000E+00  0.230085E+00  0.168779E-02
EQUALITY CONSTRAINTS
   H                   0.000000E+00 -0.101327E+01 -0.591231E+00

LOOP NUMBER .........   [INITIAL]          3              4
UNKNOWNS
   X                  -0.100000E+01  0.117343E+01  0.117181E+01
   Y                   0.000000E+00  0.809790E+00  0.810379E+00
OBJECTIVE
   F                   0.100000E+02  0.719403E+00  0.721853E+00
INEQUALITY CONSTRAINTS
   G                   0.750000E+00  0.692142E-05  0.270073E-07
EQUALITY CONSTRAINTS
   H                   0.000000E+00 -0.587123E+00 -0.587106E+00

LOOP NUMBER .........   [INITIAL]          5              6
UNKNOWNS
   X                  -0.100000E+01  0.117171E+01  0.117170E+01
   Y                   0.000000E+00  0.810416E+00  0.810419E+00
OBJECTIVE
   F                   0.100000E+02  0.722009E+00  0.722018E+00
INEQUALITY CONSTRAINTS
   G                   0.750000E+00  0.106743E-09  0.563216E-12
EQUALITY CONSTRAINTS
   H                   0.000000E+00 -0.587106E+00 -0.587106E+00

LOOP NUMBER .........   [INITIAL]          7              8
UNKNOWNS
   X                  -0.100000E+01  0.117170E+01  0.117170E+01
   Y                   0.000000E+00  0.810419E+00  0.810419E+00
OBJECTIVE
   F                   0.100000E+02  0.722018E+00  0.722018E+00
INEQUALITY CONSTRAINTS
   G                   0.750000E+00  0.364753E-11  0.849432E-12
EQUALITY CONSTRAINTS
   H                   0.000000E+00 -0.587106E+00 -0.587106E+00

LOOP NUMBER .........   [INITIAL]          9              10
UNKNOWNS
   X                  -0.100000E+01  0.117170E+01 -0.655608E+00
   Y                   0.000000E+00  0.810419E+00 -0.451401E+00
OBJECTIVE
   F                   0.100000E+02  0.722018E+00  0.915882E+01
INEQUALITY CONSTRAINTS
   G                   0.750000E+00  0.302314E-12  0.688782E+00
EQUALITY CONSTRAINTS
   H                   0.000000E+00 -0.587106E+00 -0.226111E-08

---END OF LOOP SUMMARY
```

FIG. 7-1 JOVE Optimization Summary Report

The optimization technique signifies which SEARCH has been chosen for unconstrained optimization.

A sample iteration print generated while solving the SIMPLE problem described earlier is shown in Figure 7-2.

```
---- JOVE ITERATION 3 INVOKED AT SIMPLEJ[5] FOR MODEL MODF ----

  OPTIMIZATION TECHNIQUE ... GENERALIZED NEWTON-RAPHSON

  SUB-OPTIMIZATION INITIAL CONDITIONS AT RHO = 0.390625E-02

     PENALTY FUNCTION GRADIENT VECTOR
      0.156729E+01    0.431828E+01

     GRADIENT VECTOR NORM = 0.459390E+01

  TERMINAL CONDITIONS AFTER 25 STEPS

    OBJECTIVE FUNCTION = 0.719403E+00

    PENALIZED OBJECTIVES ... PRIMAL = 0.890125E+02 DUAL = 0.177209E+03

    CUMULATIVE MODEL EVALUATIONS = 844

    INDEPENDENT VARIABLES
     0.117343E+01    0.809790E+00

    INEQUALITY CONSTRAINTS, WITH PENALTY CONTRIBUTION  0.464097E-01
     0.692142E-05

    EQUALITY CONSTRAINTS, WITH PENALTY CONTRIBUTION  0.882466E+02
    -0.587123E+00

   CONVERGENCE CONDITION AFTER  3 ITERATIONS
      OBJECTIVE CRITERION UNSATISFIED
      SPECIFIED CRITERIA UNSATISFIED

---END JOVE ITERATION 3
```

FIG. 7-2 JOVE Detailed Iteration Report

7.2. Solver ZEUS

1st Order Only

ZEUS applies the same general approach as JOVE for the solution of constrained optimization problems. However, its method for performing the unconstrained suboptimizations is a variation of the Davidon-Fletcher-Powell (DFP) technique. This is a first-order technique, employing only gradients, in contrast with the second-order technique of JOVE. By way of comparison, ZEUS will often prove more reliable for problems in which the Hessian matrix of the augmented objective is ill-conditioned.

The general form of the FIND statement for ZEUS is :

FIND *unknowns*; **IN** *model*; **BY ZEUS** {(*controller*)};
{**WITH**|**AND**} **LOWER** *floor*;} {**WITH**|**AND**} **UPPER** *ceiling*;}
{**REPORTING** *auxiliaries*;} {**WITH FLAG** *signal*;}
{**HOLDING** *inequalities*;} {**MATCHING** *equalities*;}
TO MINIMIZE|**MAXIMIZE** *objective*

Except for the BY solver clause, which invokes ZEUS, the form, meaning, and usage of this statement is identical to that described for JOVE.

When ZEUS is invoked by execution of the FIND statement, it activates derivative evaluation to compute gradients with respect to the unknowns.

The Solution Process and ZEUS Control - As noted above, the solution process is identical to that of JOVE, except that a Davidon-Fletcher-Powell (DFP) technique is used to select a direction for performing each of the sequence of unconstrained suboptimization. The ZEUS controls are also the same as those for JOVE, except that the SEARCH switch has no significance.

7.3. Solver THOR

1st order only

THOR applies a "sectionally linearized" linear programming technique to the solution of constrained optimization problems. Where the constraints and objective are linear, a modified simplex algorithm is applied. Where they are nonlinear, a sequence of linear programming problems is generated in which the model is locally linearized. This is accomplished in such a way that the solutions of the linearized subprograms converge to the true solution of the nonlinear problem. There is no requirement to identify which circumstance holds, as THOR automatically detects which technique is appropriate.

For true linear programming problems, THOR is the preferred solver. As they become progressively more nonlinear, either of JOVE, ZEUS, or JUPITER may be expected to give better performance.

The FIND statement for THOR takes the following form:

FIND *unknowns*; **IN** *model;* **BY THOR** {*(controller)*};
{**WITH**|**AND**} **LOWER** *floor;*} {**WITH**|**AND**} **UPPER** *ceiling;*}
{**REPORTING** *auxiliaries;*} {**WITH FLAG** *signal;*}
{**HOLDING** *inequalities;*} {**MATCHING** *equalities;*}
{**WITH BOUNDS** *limits;*} **TO MINIMIZE**|**MAXIMIZE** *objective*

Except for the BY *solver* clause, which invokes THOR, and the use of BOUNDS, the form, meaning, and usage of this statement are identical to that of JOVE. (The BOUNDS parameters are described below.)

When THOR is invoked by executing a FIND statement, it activates derivative evaluation to compute gradients with respect to the *unknowns*.

Unlike JOVE and ZEUS, THOR cannot be used to solve unconstrained optimization problems. This limitation is not particularly serious, however, since it is almost always possible, and usually necessary, to establish some inequality constraints to assure a

physically reasonable solution.

The Solution Process - THOR's fundamental optimization technique is the revised simplex method. Constraints are incorporated by means of slack variables and the problem is linearized in the vicinity of the current approximation point. As each subproblem is solved, the problem is re-linearized and the allowable changes in the unknowns are bounded to minimize nonlinearity error. For highly nonlinear problems, this may limit these changes in a way which impedes progress to a solution. Although this may result in many more iterations, the overall efficiency of the technique provides a compensating factor.

Equality constraints are accommodated in a special manner by THOR. Rather than incorporating them directly, each is decomposed into two inequality constraints, i.e.,

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

becomes

$$h_1(x_1, ..., x_n) + \delta \geq 0 \quad \text{and} \quad \delta - h_2(x_1, ..., x_n) \geq 0$$

This approach, which is required by the general solution process, is less effective than the penalty function methods of JOVE and ZEUS, and far less effective than using AJAX to satisfy equality constraints[4].

Many problems permit an error bound or range over which a constraint may be satisfied. For example, a temperature may be constrained to 70 $\pm 5^{\circ}$F. For such cases, it is better to impose two inequality constraints than to force the solution to satisfy the equality constraint 70 $\pm \delta^{\circ}$ F, since δ, as chosen by THOR, may be a physically unreasonable limitation.

Bounding - Since the problem is only locally linearized, it is necessary to limit the calculated changes in the unknowns as the successive approximations are performed. An upper limit is provided by the BOUNDS parameters. This is a list of variables whose values, at the time at which the FIND statement is executed, prescribe the maximum allowable changes in each unknown during any iteration. The specification of BOUNDS may take the same forms as described for the analogous parameters of HERA (see Section 6.1). In the case of THOR, however, the use of the FRACT control is discouraged because the bounding is far more critical for this technique. Reasonable values for bounds are approximately 10 percent of the total expected change in the associated unknowns.

---

4 When equality constraints are required, it is essential (for THOR) to provide an initial starting point which satisfies these constraints. If it is difficult to arrive at approximate values for the independent variables, because of complex relationships, then the technique described in Section 6.2.1 may be used.

THOR also applies "adaptive move limits" which actually compute the nonlinearity error and restrict the changes in the unknowns to values which guarantee no error greater than a prescribed value (given by the control ERRMAX). This operation may be disabled by resetting the control variable ADAPT, but this usually is poor practice. If nonlinearity error is so large that the adaptive limits are unacceptably small, an alternate solver should be used or a different initial estimate should be given. When adaptive limits are being used, the upper limits on the changes, as specified by the BOUNDS, are automatically doubled in order to permit the maximum change consistent with the objective function.

Convergence - Detection of an optimum may occur in either of two ways, depending upon whether or not a true local extremum exists within the feasible region. If the real optimum lies outside of this region, the solution to the problem lies on the constraint boundary at a point closest to the infeasible exterior optimum. In this sense, "closest" means "as measured along the gradient." This condition is detected by the fact that the computed change (step size) for one or more unknowns was less than its current limit. On the other hand, if the optimum is accessible, the solution process uses the measured changes in the objective to progressively reduce the step size limits until they fall below a prescribed criterion (UNKNOWN) or until the objective improvement is acceptably small (PROGRESS).

The method is as follows. Changes in the objective function are classified as "small" or "large" according to whether they are less than or greater than a discriminator PROGTEST. When the number of consecutive small moves equals a control called STEPLIM, the step size limits are halved; if the number of consecutive large moves equals STEPLIM, the limits are doubled (up to a maximum initially set by BOUNDS). There are two alternate convergence criteria which may be applied. The objective convergence criterion measures the relative change in the objective against the measure PROGRESS. If the change is smaller than PROGRESS, then the criterion in satisfied. The *unknowns convergence* criterion measures the current step size limits against the originally specified bounds for each unknowns. If all are less than UNKNOWN, then the criterion is satisfied. Either or both criteria may be activated by the control variable CONVERGE.

THOR Controls - The control variables for THOR are as follows[5]:

---

5 An iteration is defined to be a complete linear programming subproblem.

Variable	Value	Option	Preset Value
ERRMAX		Maximum permitted nonlinearity error, measured by the current values of $\lvert \partial^2 f/\partial x_I^2 \rvert$ for all x_I	0.05
ADAPT	1	Apply adaptive control to hold nonlinearity error below ERRMAX.	1
	0	Disable adaptive control.	
UNKNOWN		Unknowns convergence criterion; an interior optimum has been reached when $\lvert \Delta x_I/b_I \rvert \leq$ UNKNOWN. for all x_I, where b_I are the initial step size bounds.	0.05
PROGRESS		Objective convergence criterion; an interior optimum has been reached when $\lvert \Delta f/f \rvert \leq$ PROGRESS.	0.01
PROGTEST		Relative discriminator for determining if a change in the objective is large or small; measured against $\lvert \Delta f/f \rvert$	0.01
STEPLIM		Number of consecutive large or small moves required before changing the step size limits	3
FRACT		Fractional bound to be applied to all independent variables	0.5
CONVERGE	1	Satisfy objective or unknowns convergence.	1
	2	Satisfy objective and unknowns convergence.	
REMAX		Maximum number of allowed iterations	40
DETAIL	0	No detailed iteration print.	0
	n	Detailed print every nth iteration plus first and last	
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	Breakpoints after every nth iteration	
SUMMARY	1	Print iteration summary	1
	0	No iteration summary	
RESET	1	Decrease the step size and restore the previous iteration values whenever Δf reverses sign.	0
	0	Decrease the step size but continue from the current values whenever Δf reverses sign.	

For the majority of fairly linear problems, the preset controls should prove acceptable. For cases in which this is not true, an examination of the iteration and summary prints usually suggests how controls may be reset for greater effectiveness. In general, better convergence properties are achieved with adaptive error control and with STEPLIM =3 or 4. For PROGTEST much greater than 0.05, spurious convergence at a suboptimal point may occur. As a guideline for these controls:

Large PROGTEST Small STEPLIM	Fast convergence but some danger of selecting a suboptimal solution
Small PROGTEST Small STEPLIM	Quick approach to the vicinity of the optimum but slow convergence thereafter
Small PROGTEST Large STEPLIM	Stable but relatively slow convergence

The accuracy of the solution is directly related to the control UNKNOWN. The smaller its value, the more accurate the converged objective. There is a price paid for increased accuracy, however, by an increase in the number of iterations required. As a rough guide, a second order of magnitude improvement in accuracy will increase the number of iterations by 30 percent.

The control switch RESET is mainly useful when the initial estimate is known to be close to the desired optimum, for example when the last of a sequence of progressive optimizations is being performed. Turning this switch on will prevent any serious overshoot.

Optimization Summary Report - A standard optimization report is nominally generated by THOR. In format and content, it is identical to that produced by JOVE (see Figure 7-1). It should be recalled that each iteration is a complete linear programming problem. Thus, when the objective and the constraints are linear, it will converge in a single iteration.

Detailed Iteration Report - A detailed iteration report is issued by THOR whenever DETAIL is nonzero. It displays the values of the objective, the unknowns, and the constraints and gives information which indicates how step size limits are currently progressing. When adaptive limits are introduced or backtracking occurs under the RESET option, a suitable notation is printed.

7.4. Solver JUPITER

JUPITER[6] may be applied to solve either constrained or unconstrained optimization problems, and trial usage has shown it to be generally superior in speed, accuracy, and memory requirements to any of the previously described solvers. For unconstrained problems, the solution technique is the Davidon-Fletcher-Powell (DFP) variable-metric method. This is a first order method, i.e., one requiring only first partial derivatives in the model.

The general form of the FIND statement for JUPITER is

FIND *unknowns*; **IN** *model*; **BY JUPITER**{(*controller*)};
{{**WITH**|**AND**} **LOWER** floor;}{{**WITH**|**AND**} **UPPER** *ceiling*;}
{**REPORTING** *auxiliaries*;} {**WITH FLAG** *signal*;}
{{**AND**} **HOLDING** *inequalities*;}{{**AND**} **MATCHING** *equalities*;}
TO MINIMIZE|**MAXIMIZE** *objective*

Except for the BY solver clause, which invokes JUPITER, the form, meaning, and usage of this statement are identical to that of JOVE (Section 7.1).

The Solution Process - Constrained problems are transformed into a sequence of unconstrained problems by a penalty function technique. A modified objective function,

$$P(\bar{x},t) = \min\{0, [t-f(\bar{x})]\}^2 + \sum_{i=1}^{l} h_i^2(\bar{x}) + \sum_{i=1}^{m+2n} \min[0, g_i(\bar{x})]^2 \qquad (1)$$

called the Moving Exterior Truncations (MET) penalty function. is formulated from the objective function $f(\bar{x})$ incorporating the m inequality constraints $g_i(\bar{x})$ including constant constraints, and the l equality constraints $h_i(\bar{x})$. In the MET function, t is a truncation level. The basic procedure for utilizing the MET penalty function is to minimize equation (1) for a monotonic increasing sequence of truncation levels $\{t_k\}$ converging to f^*, the value of the objective function at the constrained minimum, so that

$$t_{k-1} < t_k \leq f^*$$

where the subscript k denotes the kth unconstrained minimization. In order to find the initial trucation level FIGURE, which is not known a priori, the parametric quadratic loss penalty function

$$P_L(\bar{x},\rho) = f(\bar{x}) + \frac{1}{\rho}\sum_{i=1}^{l} h_i^2(\bar{x}) + \frac{1}{\rho}\sum_{i=1}^{m+2n} \min[0, g_i(\bar{x})]^2 \qquad (2)$$

6 This solver is based upon an algorithm called COMET, contributed by Professor D. M. Himmelblau, and documented by R.L. Staha, Dept. of Chemical Engineering, the University of Texas at Austin.

is minimized on the first stage, where ρ is the weighting parameter. Because a point minimizing equation (2) also minimizes equation (1) for an appropriate value of t, the initial truncation level is found from

$$t_1 = f\ [\bar{x}^\circ(\rho_1)] - \rho/2$$

where $\bar{x}^\circ(\rho_1)$ denotes the point minimizing equation (2) for the parameter value ρ_1.

The criteria which must be satisfied before final convergence to the problem solution are the following:

$$f\,[\bar{x}^\circ(t_k)] - t_k \le \varepsilon_T$$

$$|h_i\,[\bar{x}^\circ(t_k)]| \le \varepsilon_T, \quad i=1, \ldots l$$

$$g_i\,[\bar{x}^\circ(t_k)] \le -\varepsilon_T, \quad i=1, \ldots m+2n$$

where ε_T is a small positive number and $\bar{x}^\circ(t_k)$ denotes the point minimizing the MET penalty function for the truncation level t_k.

JUPITER Controls - JUPITER has the following control variables:

Variable	Value	Option	Preset Value
ABORT	0	Delay a model abort until the current iteration is complete.	1
	1	Execute a model abort after the current step in which an ABORT statement was executed.	
DETAIL	0	No detailed iteration print	1
	n	Detailed print every nth iteration plus first and last	
SUMMARY	1	Print iteration summary	1
	0	No iteration summary	
REMAX		Maximum number of allowed iterations	20
STEPSLIM		Limit on the number of steps per iteration Since an iteration amounts to an entire unconstrained optimization, this control is analogous to REMAX for solver HERA. However, if solving a constrained problem, exceeding STEPSLIM does not terminate the process, it merely halts the current iteration.	250

Variable	Value	Option	Preset Value
STEPOUT	n	Provides snapshot prints of the step- by-step solution progress at every nth step of all iterations (about 4-9 lines of critical problem date).	-5
	-n	Constraint values are added to the snapshot prints.	
	0	No snapshot prints are generated.	
EVALMAX		Limit on the total number of model evaluations The optimization process is halted when this limit is exceeded and the result is flagged in the same manner as if maximum iterations were reached. The actual number of evaluations may exceed EVALMAX slightly because of the need to terminate the solution process correctly.	1000
ZERO		Iteration convergence tolerance	10^{-6}
BREAKIN	0	No interactive breakpoints	0
	n	Breakpoint after each nth iteration	
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	
ACCURACY		Relative tolerance for constraint satisfaction at convergence. This variable must be greater than control variable ZERO.	0.001
RHO		Weighting factor for the penalty function during the initial iteration	0.02