NASA Contractor Report 177985

ADS - A FORTRAN PROGRAM FOR AUTOMATED
DESIGN SYNTHESIS - VERSION 1.10

(NASA-CR-177985) ADS: A FORTRAN PROGRAM FOR AUTOMATED DESIGN SYNTHESIS: VERSION 1.10 Final Report (California Univ.) 46 p HC A03/MF A01 CSCL 09B N86-11894
Unclas
G3/61 27578

G. N. Vanderplaats

UNIVERSITY OF CALIFORNIA
Santa Barbara, California

Grant NAG1-567
September 1985

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

ABSTRACT

A general-purpose optimization program for engineering design is described. ADS (Automated Design Synthesis) Version 1.10 is a FORTRAN program for solution of nonlinear constrained optimization problems. The program is segmented into three levels, being strategy, optimizer, and one-dimensional search. At each level, several options are available so that a total of over 100 possible combinations can be created. Examples of available strategies are sequential unconstrained minimization, the Augmented Lagrange Multiplier method and sequential quadratic programming. Available optimizers include variable metric methods and the method of feasible directions as examples. A modified method of feasible directions, similar to the generalized reduced gradient method is included also. One-dimensional search options include the Golden Section method, polynomial interpolation, and combinations of these.

ADS version 1.10 contains several enhancements. These include general program organization, addition of equality constraints to all options in the program, and addition of a new convex linearization strategy.

Emphasis is placed on ease of use of the program. All information is transferred via a single parameter list. Default values are provided for all internal program parameters such as convergence criteria, and the user is given a simple means to over-ride these, if desired.

The program is demonstrated with a simple structural design example.

Acknowledgement: A portion of the program enhancements described herein were funded by the Optimization Users' Group, sponsered by EDO, Inc., Santa Barbara, CA.

CONTENTS

PRECEDING PAGE BLANK NOT FILMED

PAGE ii INTENTIONALLY BLANK

FIGURES

TABLES

1.0 INTRODUCTION

ADS is a general purpose numerical optimization program containing a wide variety of algorithms. The problem solved is:

Minimize F(X)

Subject to;

$$G_j(X) \leq 0 \qquad j=1,m$$

$$H_k(X) = 0 \qquad k=1,l$$

$$XL_i \leq X_i \leq XU_i \qquad i=1,n$$

The solution of this general problem is separated into three basic levels:

1. STRATEGY - For example, Sequential Unconstrained Minimization or Sequential Linear Programming.

2. OPTIMIZER - For example, Variable Metric methods for unconstrained minimization or the Method of Feasible Directions for constrained minimization.

3. ONE-DIMENSIONAL SEARCH - For example, Golden Section or Polynomial Interpolation.

Additionally, we may consider another component to be problem formulation. It is assumed that the engineer makes every effort to formulate the problem in a form amenable to efficient solution by numerical optimization. This aspect is perhaps the most important ingredient to the efficient use of the ADS program for solution of problems of practical significance.

By choosing the Strategy, Optimizer and One-Dimensional Search, the user is given considerable flexibility in creating an optimization program which works well for a given class of design problems.

This manual describes the use of the ADS program and the available program options. Section 1.1 describes the enhancements and modifications to the ADS program subsequent to Version 1.00 (ref. 1). Section 2 identifies the available optimization strategies, optimizers and one-dimensional search algorithms. Section 3 defines the program organization, and Section 4 gives user instructions. Section 5 presents several simple examples to aid the user in becoming familiar with the ADS program. Section 6 gives a simple main program that is useful for general design applications.

1.1 Enhancements and Modifications to Version 1.00

Since the release of Version 1.00 in May of 1984, several modifications and enhancements have been made to the program. Many of these are minor and are transparent to the casual user. These include various formatting changes, internal logic enhancements to improve program flow, and a few actual bugs in the FORTRAN. Because of the robustness of the basic program, where program bugs exist, their correction often is detected only in special test cases. Examples of this are enhancement of the automatic scaling of unconstrained problems, correction of an error in using the absolute convergence criteria and correction of polynomial one-dimensional search when a constraint is being followed. Other enhancements include checking to insure the initial design does not violate any side constraints, and checking to be sure the combinations of strategy, optimizer and one-dimensional search are valid.

Enhancements to the program, beyond the original capability, include addition of equality constraint capability throughout the program and addition of a new strategy.

Equality constraints are now available in all options of the program, whereas in Version 1.00 they were only available when using penalty function strategies. Specifically, equality constraints have been added to optimizers 4 and 5. Here, two approaches were investigated. The first was to formally treat them in a mathematical sense. This requires considerable program logic and usually insures rather precise following of the constraints, but at some efficiency cost. The second approach, and that used here, was to treat equality constraints via a linear penalty function and an equivalent inequality constraint. The basic concept is to first change the sign on the constraint, if necessary, so that the scalar product of the gradient of the constraint with the gradient of the objective function is negative. The constraint is then converted to a non-positive inequality constraint and a linear penalty is added to the objective. The penalty, together with the conversion to an inequality constraint have the effect of driving the original equality constraint to zero at the optimum, but without demanding precise accuracy, with its corresponding inefficiency. This is in keeping with the general phylosophy of ADS of finding a near optimum design quickly.

A new strategy (ISTRAT=9), called Sequential Convex Programming, developed by Fleury and Briabant (ref. 2), has been added to ADS. The basic concept of this strategy is that a linear approximation to the objective and constraint functions is first created, just as in sequential linear programming. However, during the approximate optimization sub-problem, either direct or reciprocal variables are used, depending on the sign of the corresponding components of the gradients. This creates a conservative convex approximation to the optimization problem. In reference 2, the method was applied to structural optimization problems in which all design variables were positive. It was shown that move limits were not required during the sub-problem and that the method converged quickly to the optimum. When incorporating the algorithm into ADS, move limits were included, but they are less stringent than for sequential linear programming. This is based on the experience that the design space can become ill-conditioned in some general applications. Also, reciprocal variables are only used if the design variable is positive. It should be emphasized here that this algorithm as well as its implementation into ADS is new and enhancements can be expected. Initial experience, especially with structural optimization problems, has shown the algorithm to be a powerful one. In Version 1.10 of ADS, this algorithm is used in conjunction with a general optimizer. Reference 2 uses a dual algorithm, for which this method is well suited. It is expected that this strategy will be modified to take advantage of duality as applied to separable problems such as this in the future.

2.0 PROGRAM OPTIONS

In this section, the options available in the ADS program are identified. At each of the three solution levels, several options are available to the user.

2.1 Strategy

Table 1 lists the strategies available. The parameter ISTRAT will be sent to the ADS program to identify the strategy the user wants. The ISTRAT=0 option would indicate that control should transfer directly to the optimizer. This would be the case, for example, when using the Method of Feasible Directions to solve constrained optimization problems because the optimizer works directly with the constrained problem. On the other hand, if the constrained optimization problem is to be solved by creating a sequence of unconstrained minimizations, with penalty functions to deal with constraints, one of the appropriate strategies would be used.

TABLE 1: STRATEGY OPTIONS

ISTRAT	STRATEGY TO BE USED
0	None. Go directly to the optimizer.
1	Sequential unconstrained minimization using the exterior penalty function method (refs. 3, 4).
2	Sequential unconstrained minimization using the linear extended interior penalty function method (refs. 5-7).
3	Sequential unconstrained minimization using the quadratic extended interior penalty function method (refs. 8, 9).
4	Sequential unconstrained minimization using the cubic extended interior penalty function method (ref. 10).
5	Augmented Lagrange Multiplier method (refs. 11-15).
6	Sequential Linear Programming (refs. 16, 17).
7	Method of Centers (method of inscribed hyperspheres) (ref. 18).
8	Sequential Quadratic Programming (refs. 13, 19, 20).
9	Sequential Convex Programming (ref. 2).

2.2 Optimizer

Table 2 lists the optimizers available. IOPT is the parameter used to indicate the optimizer desired.

TABLE 2: OPTIMIZER OPTIONS

IOPT	OPTIMIZER TO BE USED
0	None. Go directly to the one-dimensional search. This option should be used only for program development.
1	Fletcher-Reeves algorithm for unconstrained minimization (refs. 21).
2	Davidon-Fletcher-Powell (DFP) variable metric method for unconstrained minimization (refs. 22, 23).
3	Broydon-Fletcher-Goldfarb-Shanno (BFGS) variable metric method for unconstrained minimization (refs. 24-27).
4	Method of Feasible Directions (MFD) for constrained minimization (refs. 28, 29).
5	Modified Method of Feasible Directions for constrained minimization (ref. 30).

In choosing the optimizer (as well as strategy and one-dimensional search) it is assumed that the user is knowledgeable enough to choose an algorithm consistent with the problem at hand. For example, a variable metric optimizer would not be used to solve constrained problems unless a strategy is used to create the equivalent unconstrained minimization task via some form of penalty function.

2.3 One-Dimensional Search

Table 3 lists the one-dimensional search options available for unconstrained and constrained problems. Here IONED identifies the algorithm to be used.

TABLE 3: ONE-DIMENSIONAL SEARCH OPTIONS

IONED	ONE-DIMENSIONAL SEARCH OPTION (refs. 3, 31, 32)
1	Find the minimum of an unconstrained function using the Golden Section method.
2	Find the minimum of an unconstrained function using the Golden Section method followed by polynomial interpolation.
3	Find the minimum of an unconstrained function by first finding bounds and then using polynomial interpolation.
4	Find the minimum of an unconstrained function by polynomial interpolation/extrapolation without first finding bounds on the solution.
5	Find the minimum of an constrained function using the Golden Section method.
6	Find the minimum of an constrained function using the Golden Section method followed by polynomial interpolation.
7	Find the minimum of an constrained function by first finding bounds and then using polynomial interpolation.
8	Find the minimum of an constrained function by polynomial interpolation/extrapolation without first finding bounds on the solution.

2.4 Allowable Combinations of Algorithms

Not all combinations of strategy, optimizer and one-dimensional search are meaningful. For example, constrained one-dimensional search is not meaningful when minimizing unconstrained functions.

Table 4 identifies the combinations of algorithms which are available in the ADS program. In this table, an X is used to denote an acceptable combination of strategy, optimizer and one-dimensional search. An example is shown by the heavy line on the table which indicates that constrained optimization is to be performed by the Augmented Lagrange Multiplier Method (ISTRAT=5), using the BFGS optimizer (IOPT=3) and polynomial interpolation with bounds for the one-dimensional search (IONED=3). From the table, it is clear that a large number of possible combinations of algorithms are available.

TABLE 4: PROGRAM OPTIONS

	OPTIMIZER				
STRATEGY	1	2	3	4	5
0	X	X	X	X	X
1	X	X	X	0	0
2	X	X	X	0	0
3	X	X	X	0	0
4	X	X	X	0	0
5	X	X	X	0	0
6	0	0	0	X	X
7	0	0	0	X	X
8	0	0	0	X	X
9	0	0	0	X	X
ONE-D SEARCH					
1	X	X	X	0	0
2	X	X	X	0	0
3	X	X	X	0	0
4	X	X	X	0	0
5	0	0	0	X	X
6	0	0	0	X	X
7	0	0	0	X	X
8	0	0	0	X	X

Appendix A contains an annotated version of Table 4 for convenient reference once the user is familiar with ADS.

To conserve computer storage, it may be desirable to use only those subroutines in the ADS system needed for a given combination of ISTRAT, IOPT and IONED. Appendix C provides the information necessary for this. Appendix D lists the subroutines with a very brief description of each.

3.0 PROGRAM FLOW LOGIC

ADS is called by a user-supplied calling program. ADS does not call any user-supplied subroutines. Instead, ADS returns control to the calling program when function or gradient information is needed. The required information is evaluated and ADS is called again. This provides considerable flexibility in program organization and restart capabilities.

ADS can be used in four principal modes:

1. Default Control parameters and finite difference gradients.

2. Over-ride default parameters, use finite difference gradients.

3. Default control parameters and user-supplied gradients.

4. Over-ride default parameters and user-supplied gradient.

The first mode is the simplest "black box" approach. In the second mode, the user over-rides the default parameters to "fine tune" the program for efficiency. In modes 3 and 4, the user supplies all needed gradient information to the program.

Figure 1 is the program flow diagram for the simplest use of ADS. The user begins by defining the basic control parameters and arrays (to be described in Section 4). The gradient computation parameter, IGRAD, is set to zero to indicate that finite difference gradients will be used. The information parameter, INFO, is initialized to zero and ADS is called for optimization. Whenever the values of the objective, OBJ, and constraints, G(I), I=1,NCON, are required, control is returned to the user with INFO=1. The functions are then evaluated and ADS is called again. When INFO=0 is returned to the user, the optimization is complete.

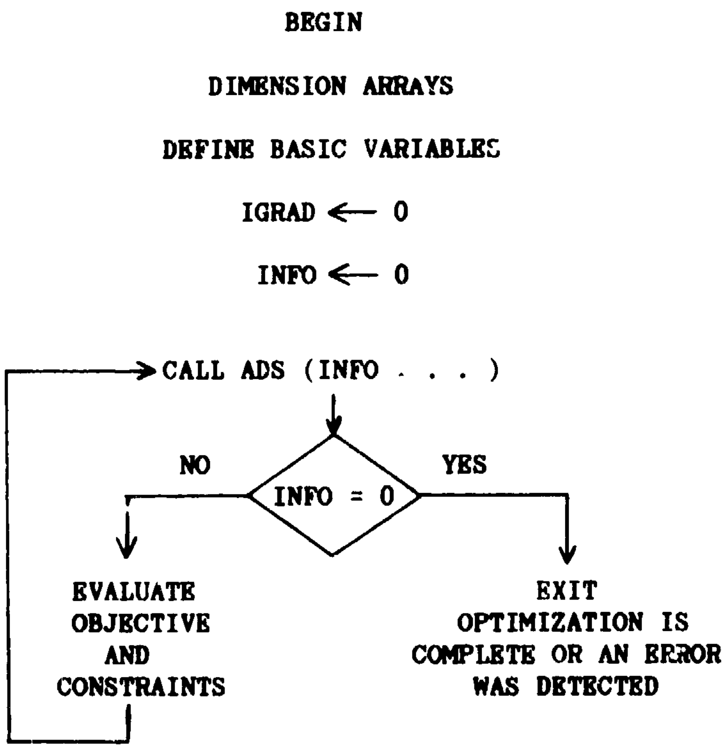

Figure 1: Simplified Program Usage; All Default Parameters and Finite Difference Gradients

Figure 2 is the program flow diagram for the case where the user wishes to over-ride one or more internal parameters, such as convergence criteria or maximum number of iterations. Here, after initialization of basic parameters and arrays, the information parameter, INFO, is set to -2. ADS is then called to initialize all internal parameters and allocate storage space for internal arrays. Control is then returned to the user, at which point these parameters, for example convergence criteria, can be over-ridden if desired. At this point, the information parameter, INFO, will have a value of -1 and should not be changed. ADS is then called again and the optimization proceeds. Section 4.3 provides a list of internal parameters which may be modified, along with their locations in the work arrays WK and IWK.

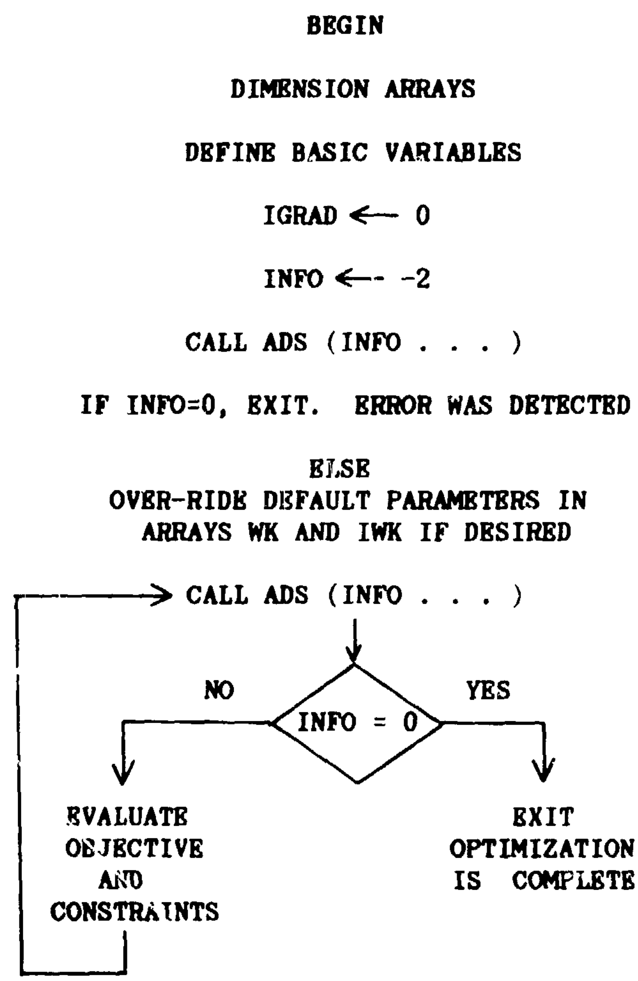

Figure 2: Program Flow Logic; Over-ride Default Parameters, Finite Difference Gradients

Figure 3 is the flow diagram for the case where the user wishes to provide gradient information to ADS, rather than having ADS calculate this information using finite difference methods. In Figure 3, it is also assumed that the user will over-ride some internal parameters, so the difference between Figures 2 and 3 is that IGRAD is now set to 1 and the user will now provide gradients during optimization. If the user does not wish to over-ride any default parameters, INFO is initialized to zero and the first call to ADS is omitted (as in Figure 1). Now, when control is returned to the user, the information parameter will have a value of 1 or 2 (if INFO=0, the optimization is complete, as before). If INFO=1, the objective and constraint functions are evaluated and ADS is called again, just as in Figure 2. If INFO=2, the gradient, DF, of the objective function is evaluated as well as the gradients of NGT constraints defined by vector IC.

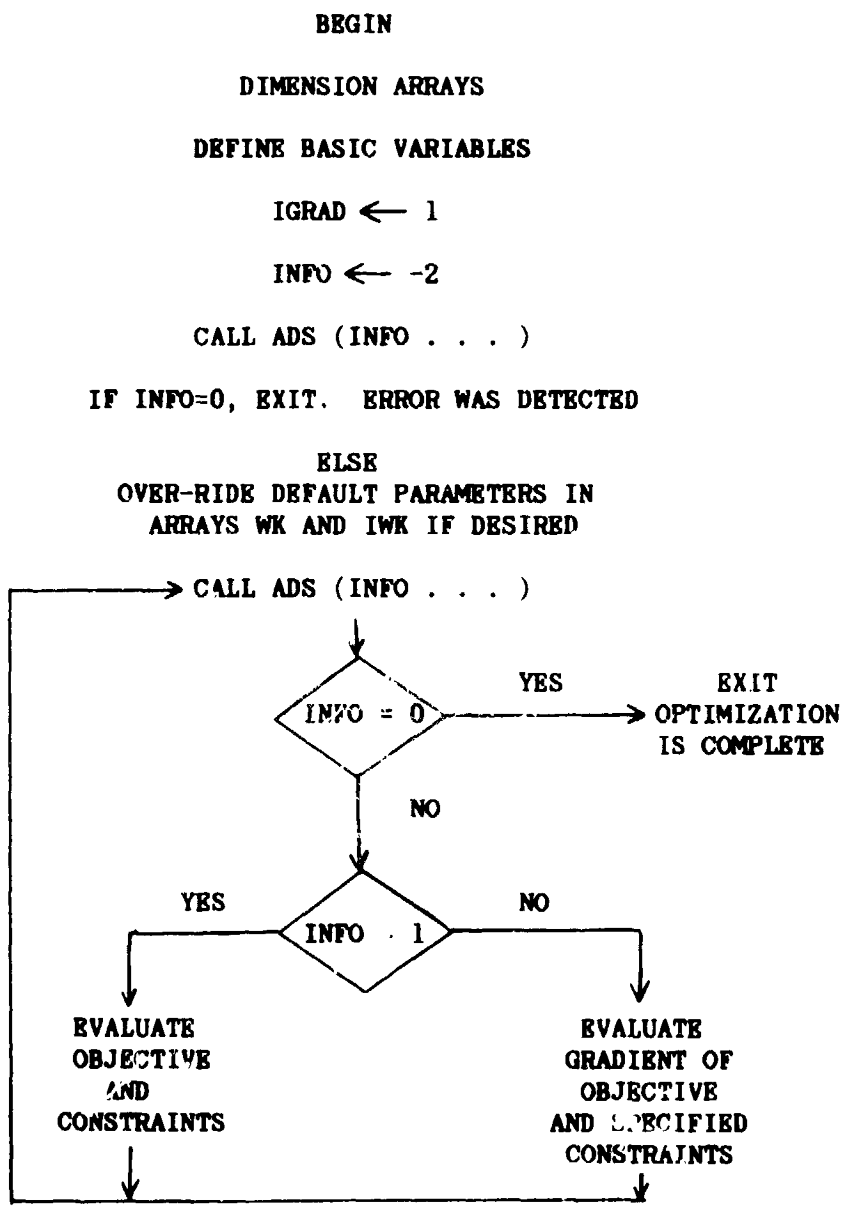

Figure 3: Program Flow Logic; Over-ride Default Parameters and Provide Gradients

4.0 USER INSTRUCTIONS

In this section, the use of the ADS program is outlined. The FORTRAN Call statement to ADS is given first, and then the parameters in the calling statement are defined. Section 4.3 identifies parameters that the user may wish to over-ride to make more effective use of ADS. Arrays are designated by boldface print.

4.1 Calling Statement

ADS is invoked by the following FORTRAN calling statement in the user's program:

```
CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X,
* VLB,VUB,OBJ,G,IDG,NGT,IC,DF,A,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
```

4.2 Definitions of Parameters in the ADS Calling Statement

Table 5 lists the parameters in the calling statement to ADS. Where arrays are defined, the required dimension size is given as the array argument.

TABLE 5: PARAMETERS IN THE ADS ARGUMENT LIST

PARAMETER	DEFINITION
INFO	Information parameter. On the first call to ADS, INFO=0 or -2. INFO=0 is used if the user does not wish to over-ride internal parameters and INFO=-2 is used if internal parameters are to be changed. When control returns form ADS to the calling program, INFO will have a value of 0, 1, or 2. If INFO=0, the optimization is complete. If INFO=1, the user must evaluate the objective, OBJ, and constraint functions, G(I), I=1,NCON, and call ADS again. If INFO=2, the user must evaluat the gradient of the objective and the NGT constraints identified by the vector IC, and call ADS again. If the gradient calculation control, IGRAD=0, INFO=2 will never be returned from ADS, and all gradient information is calculated by finite difference within ADS.
ISTRAT	Optimization strategy to be used. Available options are identified in Tables 1 and 4.
IOPT	Optimizer to be used. Available options are identified in Tables 2 and 4.
IONED	One-dimensional search algorithm to be used. Available options are identified in Tables 3 and 4.

TABLE 5 CONTINUED: PARAMETERS IN THE ADS ARGUMENT LIST

PARAMETER	DEFINITION
IPRINT	A four-digit print control. IPRINT=IJKL where I, J, K and L heve the following definitions: I ADS system print control. 0 - No print. 1 - Print initial and final information. 2 - Same as 1 plus parameter values and storage needs. 3 - Same as 2 plus scaling information calculated by ADS. J Strategy print control. 0 - No print. 1 - Print initial and final optimization information. 2 - Same as 1 plus OBJ and X at each iteration. 3 - Same as 2 plus G at each iteration. 4 - Same as 3 plus intermediate information. 5 - Same as 4 plus gradients of constraints. K Optimizer print control. 0 - No print. 1 - Print initial and final optimization information. 2 - Same as 1 plus OBJ and X at each iteration. 3 - Same as 2 plus constraints at each iteration. 4 - Same as 3 plus intermediate optimization and one-dimensional search information. 5 - Same as 4 plus gradients of constraints. L One-Dimensional search print control, (debug only). 0 - No print. 1 - One-dimensional search debug information. 2 - More of the same. Example: IPRINT=3120 corresponds to I=3, J=1, K=2 and L=0. NOTE: IPRINT can be changed at any time control is returned to the user.
IGRAD	Gradient calculation control. If IGRAD=0 is input to ADS, all gradient computations are done within ADS by first forward finite difference. If IGRAD=1, the user will supply gradient information as indicated by the value of INFO.
NDV	Number of design variables contained in vector X. NDV is the same as n in the mathematical problem statement.
NCON	Number of constraint values contained in array G. NCON is the same as m+ in the mathematical problem statement given in Section 1.0. NCON=0 is allowed.
X(NDV+1)	Vector containing the design variables. On the first call to ADS, this is the user's initial estimate to the design. On return from ADS, this is the design for which function or gradient values are required. On the final return from ADS (INFO=0 is returned), the vector X contains the optimum design.
VLB(NDV+1)	Array containing lower bounds on the design variables, X. If no lower bounds are imposed on one or more of the design variables, the corresponding component(s) of VLB must be set to a large negative number, say -1.0E+15.
VUB(NDV=1)	Array containing upper bounds on the design variables, X. If no upper bounds are imposed on one or more of the design variables, the corresponding component(s) of VUB must be set to a large positive number, say 1.0E+15.

TABLE 5 CONTINUED: PARAMETERS IN THE ADS ARGUMENT LIST

PARAMETER	DEFINITION
OBJ	Value of the objective function corresponding to the current values of the design variables contained in X. On the first call to ADS, OBJ need not be defined. ADS will return a value of INFO=1 to indicate that the user must evaluate OBJ and call ADS again. Subsequently, any time a value of INFO=1 is returned from ADS, the objective, OBJ, must be evaluated for the current design and ADS must be called again. OBJ has the same meaning as F(X) in the mathematical problem statement given in Section 1.0.
G(NCON)	Array containing NCON constraint values corresponding to the current design contained in X. On the first call to ADS, the constraint values need not be defined. On return from ADS, if INFO=1, the constraints must be evaluated for the current X and ADS called again. If NCON=0, array G should be dimensioned to unity, but no constraint values need to be provided.
IDG(NCON)	Array containing identifiers indicating the type of the constraints contained in array G. IDG(I) = -2 for linear equality constraint. IDG(I) = -1 for nonlinear equality constraint. IDG(I) = 0 or 1 for nonlinear inequality constraint. IDG(I) = 2 for linear inequality constraint.
NGT	Number of constraints for which gradients must be supplied. NGT is defined by ADS as the minimum of NCOLA and NCON and is returned to the user.
IC(NGT)	Array identifying constraints for which gradients are required. IC is defined by ADS and returned to the user. If INFO=2 is returned to the user, the gradient of the objective and the NGT constraints must be evaluated and stored in arrays DF and A, respectively, and ADS must be called again.
DF(NDV+1)	Array containing the gradient of the objective corresponding to the current X. Array DF must be defined by the user when INFO=2 is returned from ADS. This will not occur if IGRAD=0, in which case array DF is evaluated by ADS.
A(NRA,NCOLA)	Array containing the gradients of the NGT constraints identified by array IC. That is, column J of array A contains the gradient of constraint number K, where K=IC(J). Array A must be defined by the user when INFO=2 is returned from ADS and when NGT.GT.0. This will not occur if IGRAD=0, in which case, array A is evaluated by ADS. NRA is the dimensioned rows of array A. NCOLA is the dimensioned columns of array A.
NRA	Dimensioned rows of array A. NRA must be at least NDV+1.
NCOLA	Dimensioned columns of array A. NCOLA should be at least the minimum of NCON and 2*NDV. If enough storage is available, and if gradients are easily provided or are calculated by finite difference, then NCOLA=NCON+NDV is ideal.
WK(NRWK)	User provided work array for real variables. Array WK is used to store internal scalar variables and arrays used by ADS. WK must be dimensioned at least 100, but usually much larger. If the use has not provided enough storage, ADS will print the appropriate message and terminate the optimization.

TABLE 5 CONCLUDED: PARAMETERS IN THE ADS ARGUMENT LIST

PARAMETER	DEFINITION
NRWK	Dimensioned size of work array WK. A good estimate is NRWK = 500 + 10*(NDV+NCON) + NCOLA*(NCOLA+3) +N*(N/2+1), where N = MAX(NDV,NCOLA).
IWK(NRIWK)	User provided work array for integer variables. Array IWK is used to store internal scalar variables and arrays used by ADS. IWK must be dimensioned at least 200, but usually much larger. If the user has not provided enough storage, ADS will print the appropriate message and terminate the optimization.
NRIWK	Dimensioned size of work array IWK. A good estimate is NRIWK = 200 + NDV + NCON + N + MAX(N,2*NDV), where N = MAX(NDV,NCOLA).

4.3 Over-Riding ADS Default Parameters

Various internal parameters are defined on the first call to ADS which work well for the "average" optimization task. However, it is often desirable to change these in order to gain maximum utility of the program. This mode of operation is shown in Figures 2 and 3. After the first call to ADS, various real and integer scalar parameters are stored in arrays WK and IWK respectively. Those which the user may wish to change are listed in Tables 6 through 8, together with their default values and definitions. If the user wishes to change any of these, the appropriate component of WK or IWK is simply re-defined after the first call to ADS. For example, if the relative convergence criterion, DELOBJ, is to be changed to 0.002, this is done with the FORTRAN statement;

```
WK(12) = 0.002
```

because WK(12) contains the value of DELOBJ.

TABLE 6: REAL PARAMETERS STORED IN ARRAY WK

PARAMETER	LOCATION	DEFAULT	MODULES WHERE USED ISTRAT	IOPT	IONED
ALAMDZ	1	0.0	5	-	-
BETAMC	2	0.0	7	-	-
CT(1)	3	-0.03	-	4,5	-
CTL	4	-0.005	-	4,5	-
CTLMIN	5	0.001	-	4,5	-
CTMIN	6	0.01	-	4,5	-
DABALP(2)	7	0.0001	-	ALL	-
DABOBJ	8	ABS(F0)/1000	ALL	-	-
DABOBM	9	ABS(F0)/500	ALL	-	-
DABSTR	10	ABS(F0)/1000	ALL	-	
DELALF(3)	11	0.005	-	-	1,2,5,6
DELOBJ	12	0.001	-	ALL	-
DELOBM	13	0.01	ALL	-	-
DELSTR	14	0.001	ALL	-	-
DLOBJ1	15	0.1	-	ALL	-
DLOBJ2	16	1000.0	-	ALL	-
DX1	17	0.01	-	ALL	-
DX2	18	0.2	-	ALL	-
EPSPEN	19	-0.05	2,3,4	-	-
EXTRAP	20	5.0	-	-	ALL
FDCH	21	0.01	-	ALL	-
FDCHM	22	0.001	-	ALL	-
GMULTZ	23	10.0	8	-	-
PSAIZ	24	0.95	8	-	-
RMULT	25	5.0	1,5	-	-
RMVLMZ(4)	26	0.2	6,7,8,9	-	-
RP	27	10.0	1,5	-	-
RPMAX	28	1.0E+10	1,5	-	-
RPMULT	29	0.2	1,5	-	-
RPPMIN	30	1.0E-10	2,3,4	-	-
RPPRIM	31	100.0	2,3,4	-	-
SCFO	32	1.0	ALL	ALL	ALL
SCLMIN	33	0.001	ALL	ALL	ALL
STOL	34	0.001	-	4,5	-
THETAZ	35	0.1	-	4,5	-
XMULT	36	2.618034	-	-	1,2,3,5,6,7
ZRO	37	0.00001	ALL	ALL	ALL
PMLT	38	10.0	6,7,8,9	4,5	-

1 If IOPT=4, CT=-0.1
2 If IONED=3 or 8, DABALP=0.001
3 If IONED=3 or 8, DELALP=0.05
4 If ISTRAT=9, RMVLMZ=0.4

NOTE: F0 is the objective function value for the initial design.

TABLE 7: DEFINITIONS OF REAL PARAMETERS CONTAINED IN ARRAY WK

PARAMETER	DEFINITION
ALAMDZ	Initial estimate of the Lagrange Multipliers in the Augmented Lagrange Multiplier Method.
BETAMC	Additional steepest descent fraction in the method of centers. After moving to the center of the hypersphere, a steepest descent move is made equal to BETAMC times the radius of the hypersphere.
CT	Constraint tolerance in the Method of Feasible Directions or the Modified Method of Feasible Directions. A constraint is active if its numerical value is more positive than CT.
CTL	Same as CT, but for linear constraints.
CTLMIN	Same as CTMIN, but for linear constraints.
CTMIN	Minimum constraint tolerance for nonlinear constraints. If a constraint is more positive than CTMIN, it is considered to be violated.
DABALP	Absolute convergence criteria for the one-dimensional search when using the Golden Section method.
DABOBJ	Maximum absolute change in the objective between two consecutive iterations to indicate convergence in optimization.
DABOBM	Absolute convergence criterion for the optimization sub-problem when using sequential minimization techniques.
DABSTR	Same as DABOBJ, but used at the strategy level.
DELALP	Relative convergence criteria for the one-dimensional search when using the Golden Section method.
DELOBJ	Maximum relative change in the objective between two consecutive iterations to indicate convergence in optimization.
DELOBM	Relative convergence criterion for the optimization sub-problem when using sequential minimization techniques.
DELSTR	Same as DELOBJ, but used at the strategy level.
DLOBJ1	Relative change in the objective function attempted on the first optimization iteration. Used to estimate initial move in the one-dimensional search. Updated as the optimization progresses.
DLOBJ2	Absolute change in the objective function attempted on the first optimization iteration. Used to estimate initial move in the one-dimensional search. Updated as the optimization progresses.
DX1	Maximum relative change in a design variable attempted on the first optimization iteration. Used to estimate the initial move in the one-dimensional search. Updated as the optimization progresses.
DX2	Maximum absolute change in a design variable attempted on the first optimization iteration. Used to estimate the initial move in the one-dimensional search. Updated as the optimization progresses.
EPSPEN	Initial transition point for extended penalty function methods. Updated as the optimization progresses.
EXTRAP	Maximum multiplier on the one-dimensional search parameter, ALPHA in the one-dimensional search using polynomial interpolation/extrapolation.

TABLE 7 CONCLUDED: DEFINITIONS OF REAL PARAMETERS CONTAINED IN ARRAY WK

PARAMETER	DEFINITION
FDCH	Relative finite difference step when calculating gradients.
FDCHM	Minimum absolute value of the finite difference step when calculating gradients. This prevents too small a step when X(I) is near zero.
GMULTZ	Initial penalty parameter in Sequential Quadratic programming.
PSAIZ	Move fraction to avoid constraint violations in Sequential Quadratic Programming.
RMULT	Penalty function multiplier for the exterior penalty function method. Must be greater than 1.0.
RMVLMZ	Initial relative move limit. Used to set the move limits in sequential linear programming, method of inscribed hyperspheres and sequential quadratic programming as a fraction of the value of X(I), I=1,NDV.
RP	Initial penalty parameter for the exterior penalty function method or the Augmented Lagrange Multiplier method.
RPMAX	Maximum value of RP for the exterior penalty function method or the Augmented Lagrange Multiplier method.
RPMULT	Multiplier on RP for consecutive iterations.
RRPMIN	Minimum value of RPPRIM to indicate convergence.
RPPRIM	Initial penalty parameter for extended interior penalty function methods.
SCFO	The user-supplied value of the scale factor for the objective function if the default or calculated value is to be over-ridden.
SCLMIN	Minimum numerical value of any scale factor allowed.
STOL	Tolerance on the components of the calculated search direction to indicate that the Kuhn-Tucker conditions are satisfied.
THETAZ	Nominal value of the push-off factor in the Method of Feasible Directions.
XMULT	Multiplier on the move parameter, ALPHA, in the one-dimensional search to find bounds on the solution.
ZRO	Numerical estimate of zero on the computer. Usually the default value is adequate. If a computer with a short word length is used, ZRO=1.0E-4 may be preferred.
PMLT	Penalty multiplier for equality constraints when IOPT=4 or 5.

TABLE 8: INTEGER PARAMETERS STORED IN ARRAY IWK

PARAMETER	LOCATION	DEFAULT	MODULES WHERE USED ISTRAT	IOPT	IONED
ICNDIR	1	NDV+1	-	ALL	-
ISCAL	2	1	ALL	ALL	ALL
ITMAX	3	40	-	ALL	-
ITRMOP	4	3	-	1,2,3	-
ITRMST	5	2	ALL	-	-
JONED	6	IONED	8	-	-
JTMAX	7	20	ALL	-	-

TABLE 9: DEFINITIONS OF INTEGER PARAMETERS CONTAINED IN ARRAY IWK

PARAMETER	DEFINITION
ICNDIR	Restart parameter for conjugate direction and variable metric methods. Unconstrained minimization is restarted with a steepest descent direction every ICNDIR iterations.
ISCAL	Scaling parameter. If ISCAL=0, no scaling is done. If ISCAL=1, the design variables, objective and constraints are scaled automatically.
ITMAX	Maximum number of iterations allowed at the optimizer level.
ITRMOP	The number of consecutive iterations for which the absolute or relative convergence criteria must be met to indicate convergence at the optimizer level.
ITRMST	The number of consecutive iterations for which the absolute or relative convergence criteria must be met to indicate convergence at the strategy level.
JONED	The one-dimensional search parameter (IONED) to be used in the Sequential Quadratic Programming method at the strategy level.
JTMAX	Maximum number of iterations allowed at the strategy level.

4.4 User-Supplied Gradients

If it is convenient to supply analytic gradients to ADS, rather than using internal finite difference calculations, considerable optimization efficiency is attainable. If the user wishes to supply gradients, the flow logic given in Figure 3 is used. In this case, the information parameter, INFO, will be returned to the user with a value of INFO=2 when gradients are needed. The user calculates the NGT gradients of the constraints identified by array IC and stores these in the first NGT columns of array A. That is column I of A contains the gradient of constraint J, where J=IC(I).

4.5 Restarting ADS

When solving large and complex design problems, or when multi-level optimization is being performed, it is often desirable to terminate the optimization process and restart from that point at a later time. This is easily accomplished using the ADS program. Figure 4 provides the basic flowchart for this process. Whenever control is returned from ADS to the calling program, the entire contents of the parameter list are written to disk (or a file in a database management system). The program is then stopped at this point. Later, the program is restarted by reading the information back from disk and continuing from this point. If optimization is performed as a sub-problem within analysis, the information from the system level optimization is written to disk and the analysis is called. The analysis module can then call ADS to perform the sub-optimization task. Then, upon return from analysis, the system level information is read back from storage and the optimization proceeds as usual. From this, it is seen that considerable flexibility exists for multi-level and multi-discipline optimization with ADS, where the ADS program is used for multiple tasks within the overall design process.

The user may wish to stop the optimization at specific times during the process. The parameter IMAT is array IWK gives general information regarding the progress of the optimization. Appendix B provides details of this parameter as well as other parameters stored in WK and IWK which may be useful to the experienced user of ADS.

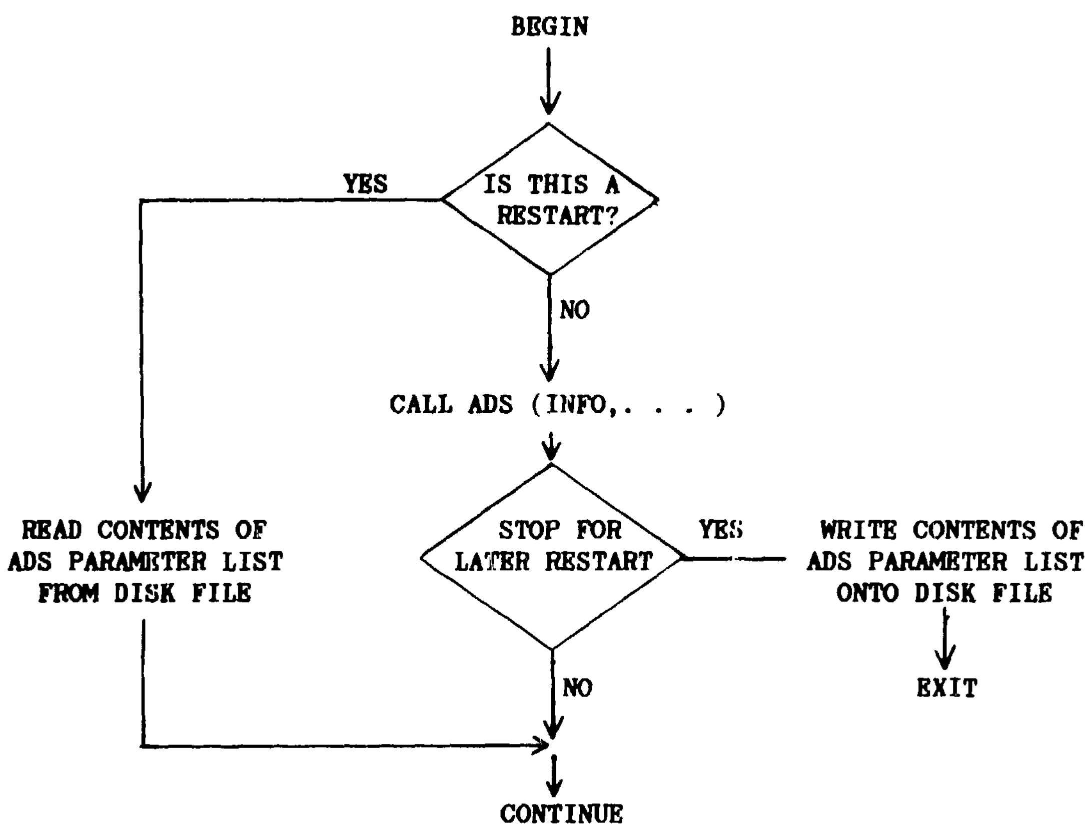

Figure 4: Restarting ADS

4.6 Choosing An Algorithm

One difficulty with a program such as ADS, which provides numerous options, is that of picking the best combination of algorithms to solve a given problem. While it is not possible to provide a concise set of rules, some general guidelines are offered here based on the author's experience. The user is strongly encouraged to try many different options in order to gain familiarity with ADS and to improve the probability that the best combination of algorithms is found for the particular class of problems being solved.

UNCONSTRAINED FUNCTIONS (NCON=0, Side Constraints OK)

```
ISTRAT=0

Is computer storage very limited?
   Yes - IOPT=1.  Are function evaluations expensive?
      Yes - Is the objective known to be approximately quadratic?
         Yes - IONED=4
         No  - IONED=3
      No  - IONED=1 or 2
   No  - Is the analysis iterative?
      Yes - IOPT=3. Are function evaluations expensive?
         Yes - Is the objective known to be approximately quadratic?
            Yes - IONED=4
            No  - IONED=3
         No  - IONED=1 or 2
      No  - IOPT=2 or 3.  Are function evaluations expensive?
         Yes - Is the objective known to be approximately quadratic?
            Yes - IONED=4
            No  - IONED=3
         No  - IONED=1 or 2
```

CONSTRAINED FUNCTIONS (NCON 0)

```
Are relative minima known to exist?
   Yes - ISTRAT=1, IOPT=3.  Are function evaluations expensive?
      Yes - IONED=3
      No  - IONED=1 or 2
   No  - Are the objective and/or constraints highly nonlinear?
      Yes - Are function evaluations expensive?
         Yes - ISTRAT=0, IOPT=4, IONED=7
         No  - ISTRAT=2, 3 or 5, IOPT=2 or 3, IONED=1 or 2
      No  - Is the design expected to be fully-constrained?
            (i.e. NDV active constraints at the optimum)
         Yes - ISTRAT=6, IOPT=5, IONED=6
         No  - Is the analysis iterative?
            Yes - ISTRAT=0, IOPT=4, IONED=7 or
                  ISTRAT=8, IOPT=5, IONED=7 or
                  ISTRAT=9, IOPT=5, IONED=7
            No  - ISTRAT=0, IOPT=5, IONED=7 or
                  ISTRAT=8, IOPT=5, IONED=7 or
                  ISTRAT=9, IOPT=5, IONED=7
```

GENERAL APPLICATIONS

Often little is known about the nature of the problem being solved. Based on experience with a wide variety of problems, a very direct approach is given here for using ADS. The following table of parameters is offered as a sequence of algorithms. When using ADS the first few times, the user may prefer to run the cases given here, rather than using the decision approach given above. It is assumed here that a constrained optimization problem is being solved. If the problem is unconstrained, ISTRAT=0, IOPT=3 and IONED=2 or 3 is recommended.

ISTRAT	IOPT	IONED	IPRINT
8	5	7	2200
0	5	7	2020
0	4	7	2020
3	5	7	2200
6	5	6	2200
5	3	3	2200
2	3	3	2200
1	3	3	2200

5.0 EXAMPLES

Consider the following two-variable optimization problem with two nonlinear constraints:

Minimize OBJ = 2*SQRT(2)*A1 + A2

$$\text{Subject to;} \quad G(1) = \frac{2*A1 + SQRT(2)*A2}{2*A1*[A1 + SQRT(2)*A2]} - 1$$

$$G(2) = \frac{1}{2*[A1 + SQRT(2)*A2]} - 1$$

$$0.01 \leq Ai \leq 1.0E+20 \qquad i=1,2$$

This is actually the optimization of the classical 3-bar truss shown in Figure 5 where, for simplicity, only the tensile stress constraints in members 1 and 2 under load P1 are included. The loads, P1 and P2, are applied separately and the material specific weight is 0.1 lb. per cubic inch. The structure is required to be symmetric so X(1) corresponds to the cross-sectional area of members 1 and 3 and X(2) corresponds to the cross-sectional area of member 2.

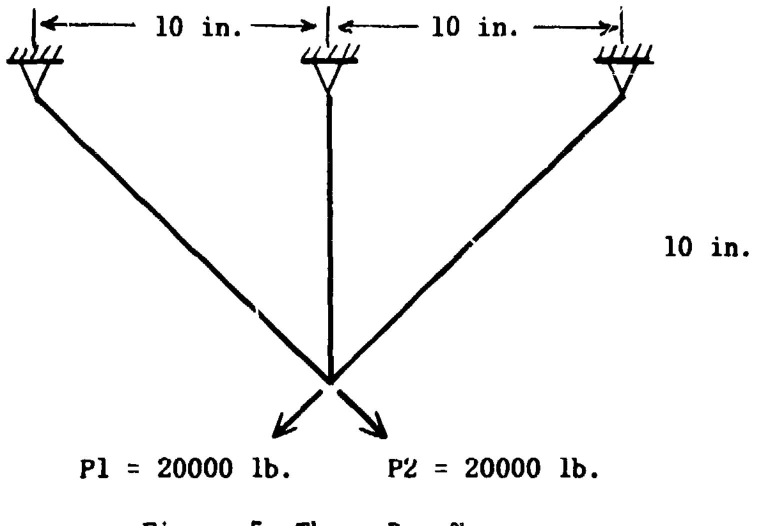

Figure 5: Three-Bar Truss

5.1 Example 1: All Default Parameters

Figure 6 gives the FORTRAN program to be used with ADS to solve this problem. Only one line of data is read by this program to define the values of ISTRAT, IOPT, IONED and IPRINT and the FORMAT is 4I5. When the optimization is complete, another case may be run by reading a new set of data. The program terminates when ISTRAT=-1 is read as data.

Figure 7 gives the results obtained with ISTRAT=0, IOPT=4, IONED=7 and IPRINT=1000. The reader is encouraged to experiment with this program using various combinations of the options from Table 4.

5.2 Example 2: Initial Parameters Are Modified

The 3-bar truss designed in Section 5.1 is now designed with the following changes in the internal parameters:

Parameter	New Value	Location in WK	Location in IWK
CT	-0.1	3	-
CTMIN	0.002	6	-
THETAZ	1.0	35	-
ITRMOP	2	-	4

The FORTRAN program used here is shown in Figure 8 and the results are given in Figure 9.

5.3 Example 3: Gradients Supplied by the User

The 3-bar truss designed in Sections 5.1 and 5.2 is designed here with user-supplied gradients. The parameters CT, CTMIN, CTMIN, THETAZ and ITRMOP are over-ridden as in Section 5.2. Also, now IPRINT=2020 to provide a more typical level of optimization output.

The FORTRAN program associated with this example is given in Figure 10. Figure 11 gives the results.

```
C      SIMPLIFIED USAGE OF ADS.  THE THREE-BAR TRUSS.
C      REQUIRED ARRAYS.
       DIMENSION X(3),VLB(3),VUB(3),G(2),IDG(2),IC(2),DF(3),A(2,2),
      1 WK(1000),IWK(500)
C      ARRAY DIMENSIONS.
       NRA=2
       NCOLA=2
       NRWK=1000
       NRIWK=500
C      PARAMETERS.
       IGRAD=0
       NDV=2
       NCON=2
C      INITIAL DESIGN.
       X(1)=1.
       X(2)=1.
C      BOUNDS.
       VLB(1)=.01
       VLB(2)=.01
       VUB(1)=1.0E+20
       VUB(2)=1.0E+20
C      IDENTIFY CONSTRAINTS AS NONLINEAR, INEQUALITY.
       IDG(1)=0
       IDG(2)=0
C      INPUT.
       READ(5,30) ISTRAT,IOPT,IONED,IPRINT
C      OPTIMIZE.
       INFO=0
 10    CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X,VLB,
      1 VUB,OBJ,G,IDG,NGT,IC,DF,A,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
       IF (INFO.EQ.0) GO TO 20
C      EVALUATE OBJECTIVE AND CONSTRAINTS.
       OBJ=2.*SQRT(2.)*X(1)+X(2)
       G(1)=(2.*X(1)+SQRT(2.)*X(2))/(2.*X(1)*(X(1)+SQRT(2.)*X(2)))-1.
       G(2)=.5/(X(1)+SQRT(2.)*X(2))-1.
C      GO CONTINUE WITH OPTIMIZATION.
       GO TO 10
 20    CONTINUE
C      PRINT RESULTS.
       WRITE(6,40) OBJ,X(1),X(2),G(1),G(2)
       STOP
 30    FORMAT (4I5)
 40    FORMAT (//5X,7HOPTIMUM,5X,5HOBJ =,E12.5//5X,6HX(1) =,E12.5,5X,
      1 6HX(2) =,E12.5/5X,6HG(1) =,E12.5,5X,6HG(2) =,E12.5)
       END
```

Figure 6: Example 1; All Default Parameters

```
 AAAAA     DDDDD     SSSSSS
A     A    D    D   S
A     A    D    D   S
AAAAAAA    D    D    SSSSS
A     A    D    D         S
A     A    D    D         S
A     A    DDDDDD   SSSSSS
```

F O R T R A N   P R O G R A M

F O R

A U T O M A T E D   D E S I G N   S Y N T H E S I S

V E R S I O N   1.10

```
CONTROL PARAMETERS
ISTRAT =     0    IOPT   =     4    IONED  =     7    IPRINT = 1000
IGRAD  =     0    NDV    =     2    NCON   =     2

                    -----------------------
                     OPTIMIZATION RESULTS
                    -----------------------


OBJECTIVE FUNCTION VALUE     .26217E+01


DESIGN VARIABLES

                  LOWER                        UPPER
VARIABLE          BOUND          VALUE         BOUND
    1          .10000E-01     .77035E+00    .10000E+21
    2          .10000E-01     .44281E+00    .10000E+21


DESIGN CONSTRAINTS

    1)    .7075E-02   -.6420E+00

FUNCTION EVALUATIONS =     56


OPTIMUM     OBJ =  .26217E+01

X(1) =  .77035E+00     X(2) =  .44281E+00
G(1) =  .70753E-02     G(2) = -.64138E+00
```

Figure 7: Example 1; Output

```
C     USAGE OF ADS.  OVER-RIDING DEFAULT PARAMETERS.
C     THE THREE-BAR TRUSS.
C     REQUIRED ARRAYS.
      DIMENSION X(3),VLB(3),VUB(3),G(2),IDG(2),IC(2),DF(3),A(2,2),
     1 WK(1000),IWK(500)
C     ARRAY DIMENSIONS.
      NRA=2
      NCOLA=2
      NRWK=1000
      NRIWK=500
C     PARAMETERS.
      IGRAD=0
      NDV=2
      NCON=2
C     INITIAL DESIGN.
      X(1)=1.
      X(2)=1.
C     BOUNDS.
      VLB(1)=.01
      VLB(2)=.01
      VUB(1)=1.0E+20
      VUB(2)=1.0E+20
C     IDENTIFY CONSTRAINTS AS NONLINEAR, INEQUALITY.
      IDG(1)=0
      IDG(2)=0
C     INPUT.
      READ(5,30) ISTRAT,IOPT,IONED,IPRINT
C     INITIALIZE INTERNAL PARAMETERS.
      INFO=-2
      CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X,VLB,
     1 VUB,OBJ,G,IDG,NGT,IC,DF,A,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
C     OVER-RIDE DEFAULT VALUES OF CT, CTMIN, THETAZ AND ITRMOP.
      WK(3)=-0.1
      WK(6)=0.002
      WK(35)=1.0
      IWK(4)=2
C     OPTIMIZE.
10    CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X,VLB,
     1 VUB,OBJ,G,IDG,NGT,IC,DF,A,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
      IF (INFO.EQ.0) GO TO 20
C     EVALUATE OBJECTIVE AND CONSTRAINTS.
      OBJ=2.*SQRT(2.)*X(1)+X(2)
      G(1)=(2.*X(1)+SQRT(2.)*X(2))/(2.*X(1)*(X(1)+SQRT(2.)*X(2)))-1.
      G(2)=.5/(X(1)+SQRT(2.)*X(2))-1.
C     GO CONTINUE WITH OPTIMIZATION.
      GO TO 10
20    CONTINUE
C     PRINT RESULTS.
      WRITE(6,40) OBJ,X(1),X(2),G(1),G(2)
      STOP
30    FORMAT (4I5)
40    FORMAT (//5X,7HOPTIMUM,5X,5HOBJ =,E12.5//5X,6HX(1) =,E12.5 5X,
     1 6HX(2) =,E12.5/5X,6HG(1) =,E12.5,5X,6HG(2) =,E12.5)
      END
```

Figure 8: Example 2; Modify Default Parameters

```
  AAAA     DDDDDD     SSSSSS
 A    A    D     D   S
 A    A    D     D   S
 AAAAAA    D     D    SSSSS
 A    A    D     D         S
 A    A    D     D         S
 A    A    DDDDDD     SSSSSS


        F O R T R A N   P R O G R A M

                    F O R

  A U T O M A T E D   D E S I G N   S Y N T H E S I S

              V E R S I O N  1.10


CONTROL PARAMETERS
ISTRAT =     0    IOPT   =    4    IONED  =    7    IPRINT = 1000
IGRAD  =     0    NDV    =    2    NCON   =    2

                  -----------------------
                   OPTIMIZATION RESULTS
                  -----------------------


OBJECTIVE FUNCTION VALUE     .26400E+01


DESIGN VARIABLES

             LOWER                         UPPER
VARIABLE     BOUND         VALUE           BOUND
    1      .10000E-01    .78640E+00     .10000E+21
    2      .10000E-01    .41569E+00     .10000E+21


DESIGN CONSTRAINTS

    1)  -.3624E-03  -.6362E+00

FUNCTION EVALUATIONS =     18


OPTIMUM      OBJ =   .26400E+01

X(1) =   .78640E+00      X(2) =   .41569E+00
G(1) = -.36236E-03       G(2) = -.63617E+00
```

Figure 9: Example 2; Output

```
C     USAGE OF ADS.  OVER-RIDING DEFAULT PARAMETERS, AND PROVIDING
C     GRADIENTS.  THE THREE-BAR TRUSS.
C     REQUIRED ARRAYS.
      DIMENSION X(3),VLB(3),VUB(3),G(2),IDG(2),IC(2),DF(3),A(2,2),
     1 WK(1000),IWK(500)
      DIMENSION B(2,2)
C     ARRAY DIMENSIONS.
      NRA=2
      NCOLA=2
      NRWK=1000
      NRIWK=500
C     PARAMETERS.
      IGRAD=1
      NDV=2
      NCON=2
C     INITIAL DESIGN.
      X(1)=1.
      X(2)=1.
C     BOUNDS.
      VLB(1)=.01
      VLB(2)=.01
      VUB(1)=1.0E+20
      VUB(2)=1.0E+20
C     IDENTIFY CONSTRAINTS AS NONLINEAR, INEQUALITY.
      IDG(1)=0
      IDG(2)=0
C     INPUT.
      READ(5,70) ISTRAT,IOPT,IONED,IPRINT
C     INITIALIZE INTERNAL PARAMETERS.
      INFO=-2
      CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X,VLB,
     1 VUB,OBJ,G,IDG,NGT,IC,DF,A,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
C     OVER-RIDE DEFAULT VALUES OF CT, CTMIN, THETAZ AND ITRMOP.
      WK(3)=-0.1
      WK(6)=0.002
      WK(35)=1.0
      IWK(4)=2
C     OPTIMIZE.
10    CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X,VLB,
     1 VUB,OBJ,G,IDG,NGT,IC,DF,A,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
      IF (INFO.EQ.0) GO TO 60
      IF (INFO.GT.1) GO TO 20
C     EVALUATE OBJECTIVE AND CONSTRAINTS.
      OBJ=2.*SQRT(2.)*X(1)+X(2)
      G(1)=(2.*X(1)+SQRT(2.)*X(2))/(2.*X(1)*(X(1)+SQRT(2.)*X(2)))-1.
      G(2)=.5/(X(1)+SQRT(2.)*X(2))-1.
C     GO CONTINUE WITH OPTIMIZATION.
      GO TO 10
```

Figure 10: Example 3; Gradients Supplied by the User

```
20    CONTINUE
C     GRADIENT OF OBJ.
      DF(1)=2.*SQRT(2.)
      DF(2)=1.0
      IF (NGT.EQ.0) GO TO 10
C     CONSTRAINT GRADIENTS.  USE ARRAY B FOR TEMPORARY STORAGE.
      D1=(X(1)+SQRT(2.)*X(2))**2
C     G(1).
      B(1,1)=-(2.*X(1)*X(1)+2.*SQRT(2.)*X(1)*X(2)+     2. *X(2)*X(2))/
     1 (2.*X(1)*X(1)*D1)
      B(2,1)=-1./(SQRT(2.)*D1)
C     G(2).
      B(1,2)=-0.5/D1
      B(2,2)=SQRT(2.)*B(1,2)
C     STORE APPROPRIATE GRADIENTS IN ARRAY A.
      DO 30 J=1,NGT
      K=IC(J)
      A(1,J)=B(1,K)
30    A(2,J)=B(2,K)
      GO TO 10
60    CONTINUE
C     PRINT RESULTS.
      WRITE(6,80) OBJ,X(1),X(2),G(1),G(2)
      STOP
70    FORMAT (4I5)
80    FORMAT (//5X,7HOPTIMUM,5X,5HOBJ =,E12.5//5X,6HX(1) =,E12.5,5X,
     1 6HX(2) =,E12.5/5X,6HG(1) =,E12.5,5X,6HG(2) =,E12.5)
      END
```

Figure 10 Concluded: Example 3; Gradients Supplied by the User

```
  AAAAA    DDDDDD     SSSSSS
 A     A   D     D   S
 A     A   D     D   S
 AAAAAAA   D     D    SSSSS
 A     A   D     D         S
 A     A   D     D         S
 A     A   DDDDDD    SSSSSS
```

```
              F O R T R A N   P R O G R A M

                          F O R

   A U T O M A T E D   D E S I G N   S Y N T H E S I S

                   V E R S I O N  1.10
```

```
CONTROL PARAMETERS
ISTRAT =    0    IOPT   =    4    IONED  =    7    IPRINT = 2020
IGRAD  =    1    NDV    =    2    NCON   =    2

SCALAR PROGRAM PARAMETERS
REAL PARAMETERS
  1) ALAMDZ =    .00000E+00     20) EXTRAP =    .50000E+01
  2) BETAMC =    .00000E+00     21) FDCH   =    .10000E-01
  3) CT     =  -.10000E+00      22) FDCHM  =    .10000E-02
  4) CTL    =  -.50000E-02      23) GMULTZ =    .10000E+02
  5) CTLMIN =    .10000E-02     24) PSAIZ  =    .95000E+00
  6) CTMIN  =    .20000E-02     25) RMULT  =    .50000E+01
  7) DABALP =    .10000E-03     26) RMVLMZ =    .20000E+00
  8) DABOBJ =    .38284E-02     27) RP     =    .10000E+02
  9) DABOBM =    .38284E-01     28) RPMAX  =    .10000E+11
 10) DABSTR =    .38284E-02     29) RMULT  =    .20000E+00
 11) DELALP =    .50000E-02     30) RPPMIN =    .10000E-09
 12) DELOBJ =    .10000E-02     31) RPPRIM =    .10000E+03
 13) DELOBM =    .10000E-01     32) SCFO   =    .10000E+01
 14) DELSTR =    .10000E-02     33) SCLMIN =    .10000E-02
 15) DLOBJ1 =    .10000E+00     34) STOL   =    .10000E-02
 16) DLOBJ2 =    .10000E+04     35) THETAZ =    .10000E+01
 17) DX1    =    .10000E-01     36) XMULT  =    .26180E+01
 18) DX2    =    .20000E+00     37) ZRO    =    .10000E-04
 19) EPSPEN =  -.50000E-01      38) PMLT   =    .10000E+02

INTEGER PARAMETERS
  1) ICNDIR =    3      4) ITRMOP =    2      6) JONED  =    7
  2) ISCAL  =    1      5) ITRMST =    2      7) JTMAX  =   20
  3) ITMAX  =   40

ARRAY STORAGE REQUIREMENTS
       DIMENSIONED  REQUIRED
ARRAY     SIZE        SIZE
  WK      1000         197
 IWK       500         184
```

Figure 11: Example 3 - Output

```
IOPT = 4; METHOD OF FEASIBLE DIRECTIONS

-- INITIAL DESIGN

OBJ =  .38284E+01

DECISION VARIABLES (X-VECTOR)
   1)    .10000E+01    .10000E+01

LOWER BOUNDS ON THE DECISION VARIABLES (VLB-VECTOR)
   1)    .10000E-01    .10000E-01

UPPER BOUNDS ON THE DECISION VARIABLES (VUB-VECTOR)
   1)    .99746E+20    .10000E+21

CONSTRAINT VALUES (G-VECTOR)
   1)   -.41831E+00   -.79289E+00


-- ITERATION    1     OBJ =  .28261E+01

DECISION VARIABLES (X-VECTOR)
   1)    .86779E+00    .37164E+00


-- ITERATION    2     OBJ =  .27594E+01

DECISION VARIABLES (X-VECTOR)
   1)    .88867E+00    .81159E+00


-- ITERATION    3     OBJ =  .26402E+01

DECISION VARIABLES (X-VECTOR)
   1)    .80834E+00    .35388E+00


-- ITERATION    4     OBJ =  .26381E+01

DECISION VARIABLES (X-VECTOR)
   1)    .79603E+00    .38657E+00


-- ITERATION    5     OBJ =  .26375E+01

DECISION VARIABLES (X-VECTOR)
   1)    .79037E+00    .40199E+00
```

Figure 11 Continued: Example 3 - Output

FINAL OPTIMIZATION RESULTS

NUMBER OF ITERATIONS = 5

OBJECTIVE = .26375E+01

DECISION VARIABLES (X-VECTOR)
1) .79037E+00 .40199E+00

CONSTRAINT VALUES (G-VECTOR)
1) .80391E-03 -.63205E+00

CONSTRAINT TOLERANCE, CT = -.51000E-01 CTL = -.30000E-02

THERE ARE 1 ACTIVE CONSTRAINTS AND 0 VIOLATED CONSTRAINTS
CONSTRAINT NUMBERS
1

THERE ARE 0 ACTIVE SIDE CONSTRAINTS

TERMINATION CRITERIA

RELATIVE CONVERGENCE CRITERION WAS MET FOR 2 CONSECUTIVE ITERATIONS

ABSOLUTE CONVERGENCE CRITERION WAS MET FOR 2 CONSECUTIVE ITERATIONS

OPTIMIZATION RESULTS

OBJECTIVE FUNCTION VALUE .26375E+01

DESIGN VARIABLES

VARIABLE	LOWER BOUND	VALUE	UPPER BOUND
1	.10000E-01	.79037E+00	.10000E+21
2	.10000E-01	.40199E+00	.10000E+21

DESIGN CONSTRAINTS

1) .5629E-03 -.6320E+00

FUNCTION EVALUATIONS = 16

GRADIENT EVALUATIONS = 5

OPTIMUM OBJ = .26375E+01

X(1) = .79037E+00 X(2) = .40199E+00
G(1) = .56288E-03 G(2) = -.63205E+00

Figure 11 Concluded: Example 3; Output

6.0 MAIN PROGRAM FOR SIMPLIFIED USAGE OF ADS

Figure 12 is a general-purpose calling program for use with ADS. The arrays are dimensioned sufficient to solve problems of up to 20 design variables and 100 constraints. Arrays IC and A are dimensioned to allow for evaluation of 30 constraint gradients. Wherever a question mark (?) is given, it is understood that the user will supply the appropriate information. Note that the statement X(I)=?, I=1,NDV is not an implied FORTRAN DO LOOP, but simply denotes that the value of the NDV design variables must be defined here.

Subroutine EVAL is the user-supplied subroutine for evaluating functions and gradients (if user-supplied). The calling statement is:

```
CALL EVAL (INFO,NDV,NCON,OBJ,X,G,DF,NGT,IC,A,NRA)
```

The parameters INFO, NDV, NCON, X, NGT, IC and NRA are input to Subroutine EVAL, while OBJ, G, DF and A are output. Depending on the user needs, this may be simplified. For example, if IGRAD=0 and NDV and NCON are not required by the analysis, the calling statement may be

```
CALL EVAL (OBJ,X,G)
```

Also, a print control may be added so, after the optimization is complete, EVAL can be called again to print analysis information.

```
C     SIMPLIFIED USAGE OF THE ADS OPTIMIZATION PROGRAM.
      DIMENSION X(21),VLB(21),VUB(21),G(100),IDG(100),IC(30),DF(21),
     * A(21,30),WK(10000),IWK(2000)
      NRA=21
      NCOLA=30
      NRWK=10000
      NRIWK=2000
C     INITIALIZATION.
      IGRAD=?
      NDV=?
      NCON=?
      X(I)=?, I=1,NDV
      VLB(I)=?,  I=1,NDV
      VUB(I)=?,  I=1,NDV
      IDG(I)=?  I=1,NCON
      ISTRAT=?
      IOPT=?
      IONED=?
      IPRINT=?
      INFO=0
10    CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X,
     * VLB,VUB,OBJ,G,IDG,NGT,IC,DF,A,NRA,NCOLA,WK,NRWK,IWK,NRIWK)
      CALL EVAL (INFO,NDV,NCON,OBJ,X,G,DF,NGT,IC,A,NRA)
      IF (INFO.GT.0) GO TO 10
C     OPTIMIZATION IS COMPLETE.  PRINT RESULTS.
      STOP
      END
```

Figure 12: Program for Simplified Usage of ADS

7.0 REFERENCES

1. Vanderplaats, G. N., "ADS - A FORTRAN Program for Automated Design Synthesis," NASA CR 172460, Oct. 1984.
2. Fleury, C. and Braibant, V., "Structural Optimization; A New Dual Method Using Mixed Variables," LTAS Report SA-115, University of Leige, Leige, Belgium, March 1984.
3. Fox, R. L., "Optimization Methods for Engineering Design," Addison-Wesley, 1971.
4. Fiacco, A. V. and McCormick, G. P., Nonlinear Programming: Sequential Unconstrained Minimization Techniques, John Wiley and Sons, 1968.
5. Kavlie, D. and Moe, J.,"Automated Design of Frame Structures," ASCE Journal of Structural Div., Vol.ST1, Jan. 1971, pp. 33-62.
6. Cassis, J. H., "Optimum Design of Structures Subjected to Dynamic Loads," Ph.D. Thesis, University of California, Los Angeles, 1974.
7. Cassis, J. H. and Schmit, L. A., "On Implementation of the Extended Interior Penalty Function," International Journal of Numerical Methods in Engineering, Vol. 10, No. 1, 1976, pp. 3-23.
8. Haftka, R. T. and Starnes, J. H.,Jr., "Applications of a Quadratic Extended Interior Penalty Function for Structural Optimization," AIAA Journal, Vol.14, June 1976, pp. 718-724.
9. Prasad, B. and Haftka, R. T., "Optimum Structural Design with Plate Finite Elements," ASCE Journal of Structural Div., Vol.ST11, Nov. 1979, pp. 2367-2382.
10. Prasad,B., "A Class of Generalized Variable Penalty Methods for Nonlinear Programming," Journal of Optimization Theory and Applications, Vol.35, No.2, Oct. 1981, pp. 159-182.
11. Rockafellar, R. T., "The Multiplier Method of Hestenes and Powell Applied to Convex Programming," Journal of Optimization Theory and Application, Vol. 12, No. 6, 1973, pp. 555-562.
12. Pierre, D. A. and Lowe, M. J., "Mathematical Programming Via Augmented Lagrangians," Applied Mathematics and Computation Series, Addison-Wesley, 1975.
13. Powell, M. J. D., "Algorithms for Nonlinear Constraints that use Lagrangian Functions," Mathematical Programming, Vol. 14, No. 2, 1978, pp. 224-248.
14. Imai, K., "Configuration Optimization of Trusses by the Multiplier Method," Ph.D. Thesis, University of California, Los Angeles, 1978.
15. Imai, K. and Schmit, L. A., "Configuration Optimization of Trusses," Journal of the Structural Division, ASCE, Vol. 107, No. ST5, May 1981, pp. 745-756.
16. Kelley, J. E., "The Cutting Plane Method for Solving Convex Programs," J. SIAM, 1960, pp. 703-712.
17. Moses, F., "Optimum Structural Design Using Linear Programming," Proc. A.S.C.E., Vol. 90, ST6, 1964, pp. 89-104.
18. Baldur, R., "Structural Optimization by Inscribed Hyperspheres," Journal of Engineering Mechanics, ASCE, Vol. 98, No. EM3, June 1972, pp. 503-508.
19. Powell, M. J. D., "The Convergence of Variable Metric Methods for Nonlinearly Constrained Optimization Calculations," Proc. Nonlinear Programming Symposium 3, Madison, Wisconsin.
20. Powell, M. J. D., "A Fast Algorithm for Nonlinearly Constrained Optimization Calculations," Report DAMTP77/NA2, University of Cambridge, England.

21. Fletcher, R. and Reeves, C. M., "Function Minimization by Conjugate Gradients," Computer Journal, Vol. 7, No. 2, 1964, pp. 149-154.
22. Davidon, W. C., "Variable Metric Method for Minimization," Argone National Laboratory, ANL-5990 Rev., University of Chicago, 1959.
23. Fletcher, R. and Powell, M. J. D., "A Rapidly Convergent Method for Minimization," Computer Journal, Vol. 6, No. 2, 1963, pp. 163-168.
24. Broydon, C. G., "The Convergence of a Class of Double Rank Minimization Algorithms," Parts I and II, J. Inst. Maths. Applns., Vol. 6, 1970, pp. 76-90 and 222-231.
25. Fletcher, R., "A New Approach to Variable Metric Algorithms," Computer Journal, Vol. 13, 1970, pp. 317-322.
26. Goldfarb, D., A Family of Variable Metric Methods Derived by Variational Means," Maths. Comput., Vol. 24, 1970, pp. 23-36.
27. Shanno, D. F., "Conditioning of Quasi-Newton Methods for Function Minimization," Maths. Comput., Vol. 24, 1970, pp. 647-656.
28. Zoutendijk, M., Methods of Feasible Directions, Elsevier Publishing Co., Amsterdam, 1960.
29. Vanderplaats, G. N. and Moses, F., "Structural Optimization by Methods of Feasible Directions," Journal of Computers and Structures, Vol. 3, Pergamon Press, July 1973, pp. 739-755.
30. Vanderplaats, G. N., "An efficient Feasible Directions Algorithm for Design Synthesis," AIAA J., Vol. 22, No. 11, Oct. 1984, pp. 1633-1640.
31. Himmelblau, D. M., Applied Nonlinear Programming, McGraw-Hill, 1972.
32. Vanderplaats, G. N., Numerical Optimization Techniques for Engineering Design: With Applications, McGraw-Hill, 1984.

APPENDIX A

QUICK REFERENCE TO ADS OPTIONS

IOPT	OPTIMIZER
1	Fletcher Reeves
2	Davidon Fletcher Powell (DFP)
3	Broydon Fletcher Goldfarb Shanno (BFGS)
4	Method of Feasible Directions
5	Modified Method of Feasible Directions

STRATEGY	ISTRAT	IOPT 1	2	3	4	5
None	0	X	X	X	X	X
SUMT, Exterior	1	X	X	X	0	0
SUMT, Linear Extended Interior	2	X	X	X	0	0
SUMT, Quadratic Extended Interior	3	X	X	X	0	0
SUMT, Cubic Extended Interior	4	X	X	X	0	0
Augmented Lagrange Multiplier Meth.	5	X	X	X	0	0
Sequential Linear Programming	6	0	0	0	X	X
Method of Centers	7	0	0	0	X	X
Sequential Quadratic Programming	8	0	0	0	X	X
Sequential Convex Programming	9	0	0	0	X	X

ONE-DIMENSIONAL SEARCH	IONED					
Golden Section Method	1	X	X	X	0	0
Golden Section + Polynomial	2	X	X	X	0	0
Polynomial Interpolation (bounded)	3	X	X	X	0	0
Polynomial Extrapolation	4	X	X	X	0	0
Golden Section Method	5	0	0	0	X	X
Golden Section + Polynomial	6	0	0	0	X	X
Polynomial Interpolation (bounded)	7	0	0	0	X	X
Polynomial Extrapolation	8	0	0	0	X	X

NOTE: An X denotes an allowed combination of algorithms.

APPENDIX B

USEFUL INFORMATION STORED IN ARRAYS WK AND IWK

Arrays **WK** and **IWK** contain information calculated by ADS which is sometimes useful in monitoring the progress of the optimization. Tables B-1 and B-2 identify parameters which may be of interest to the user. Note that these parameters must not be changed by the user during the optimization process.

TABLE B-1: REAL PARAMETERS STORED IN ARRAY WK

PARAMETER	LOCATION	DEFINITION
ALPHA	52	Move parameter in the one-dimensional search.
ALPHA3	53	Alpha at the strategy level for ISTRAT=8.
PENALT	82	The value of the penalty in SUMT methods.
SLOPF	85	The slope of the OBJ versus ALPHA function in the one-dimensional search.

TABLE B-2: INTEGER PARAMETERS STORED IN ARRAY IWK

PARAMETER	LOCATION	DEFINITION
IDAB	23	Number of consecutive times the absolute convergence criterion has been satisfied at the optimization level.
IDAB3	24	Same as IDAB, but at the strategy level.
IDEL	25	Number of consecutive times the relative convergence criterion has been satisfied at the optimization level.
IDEL3	26	Same as IDEL, but at the strategy level.
IFCALL	28	The number of times the objective and constraint functions have been evaluated.
IGCALL	29	The number of times analytic gradients have been evaluated.
IMAT	34	Pointer telling the status of the optimization process. 0 - Optimization is complete. 1 - Initialization is complete and control is being returned to the user to over-ride default parameters. 2 - Initial function evaluation. 3 - Calculating analytic gradients. 4 - Calculating finite difference gradients. NXFD identifies the design variable being changed. 5 - One-dimensional search is being performed. See LGOTO.
ITER	45	Iteration number at the optimization level.
JTER	46	Iteration number at the strategy level.
LGOTO	54	Location in one-dimensional search. 1 - Finding bounds on the solution. 2 - Golden Section method. 3 - Polynomial interpolation after Golden Section. 4 - Polynomial interpolation after getting bounds. 5 - Polynomial interpolation/extrapolation.
NAC	58	Number of active constraints.
NACS	59	Number of active side constraints.
NVC	68	Number of violated constraints.
NXFD	69	Design variable being perturbed during finite difference gradients.

APPENDIX C

SUBROUTINES NEEDED FOR A SPECIFIED COMBINATION OF ISTRAT, IOPT AND IONED

Depending on the combination of ISTRAT, IOPT and IONED, only a subset of subroutines contained in the ADS system are used. Therefore, if computer memory is limited, it may be desired only to load those routines which are actually used. This will result in "unsatisfied externals" at run time, but on most systems the program can be executed anyway since the unsatisfied external routines are not actually called. Below is a list of the routines needed for a given combination of algorithms. In some cases, slightly more routines are included than are absolutely necessary, but they are short and a more precise list would be unduly complicated.

ALWAYS LOAD THE FOLLOWING SUBROUTINES:

ADS, ADS001, ADS002, ADS004, ADS005, ADS006, ADS007, ADS009, ADS010, ADS102, ADS103, ADS105, ADS112, ADS122, ADS201, ADS206, ADS211, ADS216, ADS236, ADS237, ADS401, ADS402, ADS403, ADS420, ADS503, ADS504, ADS506, ADS510

STRATEGY LEVEL

Depending on the value of ISTRAT, the following subroutines are also required:

ISTRAT = 0, No strategy routines are added. Go to the optimizer level.

ISTRAT = 1, Add: ADS008, ADS301, ADS302, ADS508

ISTRAT = 2, Add: ADS008, ADS302, ADS303, ADS308, ADS508

ISTRAT = 3, Add: ADS008, ADS302, ADS304, ADS308, ADS508

ISTRAT = 4, Add: ADS008, ADS302, ADS305, ADS308, ADS508

ISTRAT = 5, Add: ADS008, ADS302, ADS306, ADS307, ADS508

ISTRAT = 6, Add: ADS320, ADS321, ADS323, ADS333

ISTRAT = 7, Add: ADS323, ADS330, ADS331, ADS333

ISTRAT = 8, Add: ADS207, ADS217, ADS218, ADS221, ADS223, ADS310, ADS333, ADS371, ADS375, ADS376, ADS377, ADS378, ADS404, ADS507, ADS508, ADS509

ISTRAT = 9, Add: ADS207, ADS217, ADS218, ADS221, ADS223, ADS325, ADS326, ADS509

OPTIMIZER LEVEL

Depending on the value of IOPT, the following subroutines are also required:

IOPT = 1, Add: ADS204, ADS213, ADS214, ADS509

IOPT = 2, Add: ADS213, ADS214, ADS235, ADS404, ADS503, ADS509

IOPT = 3, Add: ADS213, ADS214, ADS235, ADS404, ADS503, ADS509

IOPT = 4, Add: ADS201, ADS205, ADS207, ADS217, ADS218, ADS221, ADS223, ADS507

IOPT = 5, Add: ADS201, ADS202, ADS203, ADS207, ADS209, ADS217, ADS218, ADS221, ADS223, ADS235, ADS507

ONE-DIMENSIONAL SEARCH LEVEL

Depending on the value of IONED, the following subroutines are also required:

IONED = 1-4, Add: ADS116, ADS117, ADS118, ADS121, ADS126, ADS127

IONED = 5-8, Add: ADS101, ADS104, ADS106, ADS108, ADS109, ADS110, ADS111, ADS115, ADS119, ADS123, ADS124, ADS125, ADS502

APPENDIX D

ADS SYSTEM SUBROUTINES

The subroutines in the ADS system are listed here with a very brief description of each. Most subroutines are internally documented, and the user is referred to the program listing for more details.

Generally, ADS001-ADS099 are control level routines. ADS101-ADS199 are one-dimensional search level routines, ADS201-ADS299 are optimization level routines and ADS301-ADS399 are strategy level routines. ADS401-ADS499 are print routines and ADS501-ADS599 are utility routines.

ROUTINE	PURPOSE
ADS	- Main control routine for optimization.
ADS001	- Control one-dimensional search level.
ADS002	- Control optimizer level.
ADS003	- Control strategy level.
ADS004	- Define work array storage allocations.
ADS005	- Initialize scalar parameters to their default values.
ADS006	- Initialize scale factors.
ADS007	- Calculate scale factors, scale, unscale.
ADS008	- Calculate gradients of pseudo-objective for ISTRAT=1-5.
ADS009	- Re-order IC and A arrays.
ADS010	- Calculates convergence criteria parameters.
ADS101	- Coefficients of linear polynomial.
ADS102	- Coefficients of quadratic polynomial.
ADS103	- Coefficients of cubic polynomial.
ADS104	- Zeroes of polynomial to third-order.
ADS105	- Minimums of polynomial to third-order.
ADS106	- Evaluate n-th order polynomial.
ADS108	- Find minimum of a function by polynomial interpolation.
ADS109	- Find zeroes of a function by polynomial interpolation.

ROUTINE	PURPOSE
ADS110	- Evaluate slope of n-th order polynomial.
ADS111	- Polynomial interpolation for constraint boundaries.
ADS112	- Find ALPMAX so NDV side constraints are encountered.
ADS115	- Control one-dimensional search for constrained functions.
ADS116	- Control one-dimensional search for unconstrained functions.
ADS117	- Polynomial interpolation of unconstrained function, within bounds.
ADS118	- Polynomial interpolation of unconstrained function, no bounds given.
ADS119	- Polynomial interpolation of constrained function, no bounds given.
ADS121	- Find bounds on minimum of unconstrained function.
ADS122	- Initial interior points for Golden Section method.
ADS123	- Constrained one-dimensional search by Golden Section method.
ADS124	- Update bounds and get new interior point in Golden Section method, constrained.
ADS125	- Find bounds on minimum of constrained function.
ADS126	- Unconstrained one-dimensional search by Golden Section method.
ADS127	- Update bounds and get new interior point by Golden Section method, unconstrained.
ADS201	- Identify NGT most critical constraints.
ADS202	- Invert matrix B and store back in B.
ADS203	- Delta-X back to boundary in Modified Method of Feasible Directions.
ADS204	- Fletcher-Reeves unconstrained minimization.
ADS205	- Method of Feasible Directions.
ADS206	- X = Xold + ALPHA*S, subject to side constraints.
ADS207	- Maximum component (magnitude) of each column of A.

ROUTINE	PURPOSE
ADS209	- Calculate B = A-Transpose times A.
ADS211	- Update convergence parameters IDEL and IDAB.
ADS213	- Calculate initial ALPHA for one-dimensional search based on objective function value.
ADS214	- Calculate initial ALPHA for one-dimensional search based on X-values.
ADS216	- Finite difference gradients of objective and constraints.
ADS217	- Solve direction-finding task for Methods of Feasible Directions.
ADS218	- Solve special LP sub-problem from ADS217.
ADS221	- Push-off factors for Methods of Feasible Directions.
ADS223	- Identify active side constraints.
ADS231	- Modified Method of Feasible Directions.
ADS235	- Variable Metric Methods, IOPT=2,3.
ADS236	- Search direction for Variable Metric Methods.
ADS237	- Penalty for equality constraints, IOPT=4,5.
ADS301	- Exterior Penalty Function Method, ISTRAT=1.
ADS302	- Calculates penalty for penalty function methods, ISTRAT=1-5.
ADS303	- Linear Extended Penalty Function Method, ISTRAT=2.
ADS304	- Quadratic Extended Penalty Function Method, ISTRAT=3.
ADS305	- Cubic Extended Penalty Function Method, ISTRAT=4.
ADS306	- Augmented Lagrange Multiplier Method, ISTRAT=5.
ADS307	- Update Lagrange Multipliers, ISTRAT=5.
ADS308	- Calculate penalty parameters, ISTRAT=5.
ADS310	- Sequential Quadratic Programming, ISTRAT=8.
ADS320	- Sequential Linear Programming, ISTRAT=6.
ADS321	- Control solution of LP sub-problem, ISTRAT=6.

ROUTINE	PURPOSE
ADS323	- Update move limits, ISTRAT=6,7.
ADS325	- Sequential Convex Programming, ISTRAT=9.
ADS326	- Solve convex sub-problem, ISTRAT=9.
ADS330	- Method of Centers, ISTRAT=7.
ADS331	- Control solution of LP sub-problem, ISTRAT=7.
ADS333	- Calculate maximum constraint value.
ADS371	- Control solution of QP sub-problem, ISTRAT=8.
ADS375	- Temporary objective, ISTRAT=8.
ADS376	- Gradient of pseudo-objective for one-dimensional search, ISTRAT=8.
ADS377	- Change in objective gradients, ISTRAT=8.
ADS378	- Update Hessian matrix, ISTRAT=8.
ADS401	- Print arrays.
ADS402	- Print array title and array. Calls ADS401.
ADS403	- Print scalar control parameters.
ADS404	- Print Hessian matrix.
ADS420	- Print final optimization results.
ADS501	- Evaluate scalar product of two vectors.
ADS502	- Find maximum component of vector.
ADS503	- Equate two vectors.
ADS504	- Matrix-vector product.
ADS506	- Initialize symmetric matrix to the identity matrix.
ADS507	- Normalize vector by dividing by maximum component.
ADS508	- Calculate gradient of pseudo-objective for ISTRAT=1-5. Called by ADS008.
ADS509	- Identify active side constraints.
ADS510	- Scale, unscale the X-vector.

Standard Bibliographic Page

1. Report No.: NASA CR-177985

2. Government Accession No.:

3. Recipient's Catalog No.:

4. Title and Subtitle: ADS - A FORTRAN Program for Automated Design Synthesis - Version 1.10

5. Report Date: September 1985

6. Performing Organization Code:

7. Author(s): G. N. Vanderplaats

8. Performing Organization Report No.:

9. Performing Organization Name and Address: University of California - Santa Barbara, Santa Barbara, CA 93106

10. Work Unit No.:

11. Contract or Grant No.: NAG1-567

12. Sponsoring Agency Name and Address: National Aeronautics & Space Administration, Washington, DC 20546

13. Type of Report and Period Covered: Contractor Report

14. Sponsoring Agency Code: 505-33-53-16

15. Supplementary Notes: Langley Technical Monitor: Jaroslaw Sobieski. Final Report

16. Abstract

A new general-purpose optimization program for engineering design is described. ADS (Automated Design Synthesis - Version 1.10) is a FORTRAN program for solution of nonlinear constrained optimization problems. The program is segmented into three levels: strategy, optimizer, and one-dimensional search. At each level, several options are available so that a total of over 100 possible combinations can be created. Examples of available strategies are sequential unconstrained minimization, the Augmented Lagrange Multiplier method, and Sequential Linear Programing. Available optimizers include variable metric methods and the Method of Feasible Directions as examples, and one-dimensional search options include polynomial interpolation and the Golden Section method as examples.

Emphasis is placed on ease of use of the program. All information is transferred via a single parameter list. Default values are provided for all internal program parameters such as convergence criteria, and the user is given a simple means to over-ride these, if desired.

The program is demonstrated with a simple structural design example.

17. Key Words (Suggested by Authors(s)): Optimization, nonlinear mathematical programing

18. Distribution Statement: Unclassified - Unlimited. Star Category 61

19. Security Classif.(of this report)	20. Security Classif.(of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	47	A03

For sale by the National Technical Information Service, Springfield, Virginia 22161

NASA Langley Form 63 (June 1985)