LA-GRANT
IN-39-CR
326548
P 26

Final Report

to

NASA Langley Research Center
Langley, VA

TWO POINT EXPONENTIAL APPROXIMATION METHOD FOR STRUCTURAL OPTIMIZATION OF PROBLEMS WITH FREQUENCY CONSTRAINTS

Contract #. NAG-1-1144

G. M. Fadel
School of Mechanical Engineering
Georgia Institute of Technology
Atlanta, GA 30332

January 31, 1991

TWO POINT EXPONENTIAL APPROXIMATION METHOD FOR STRUCTURAL OPTIMIZATION OF PROBLEMS WITH FREQUENCY CONSTRAINTS

Georges M. Fadel
School of Mechanical Engineering
Georgia Institute of Technology
Atlanta, GA, 30332 USA

Abstract The Two Point Exponential Approximation Method was introduced by Fadel et al. (Fadel, 1990), and tested on structural optimization problems with stress and displacement constraints. The results reported in earlier papers were promising, and the method, which consists in correcting Taylor series approximations using previous design history, is tested in the present paper on optimization problems with frequency constraints. The aim of the research is to verify the robustness and speed of convergence of the Two Point Exponential Approximation method when highly non-linear constraints are used.

Introduction

In the practice of optimization, especially when complex structural, thermal, aerodynamic or other analyses are needed, the computer time required to perform the analyses is critical. Most large optimization problems have been formulated such that the number of full scale analyses are minimal. This is generally accomplished by reducing the original problem to an approximate, simpler model which can be optimized within certain constraints. The original problem is then solved with the optimized approximate design variables, and iterations are performed until overall convergence is attained. The critical aspect of the procedure is the quality of the approximation. For a very highly non-linear problem, linear approximations are valid only in a very small domain around the original design point, whereas in better behaved problems, larger moves can be accomplished. The trade-off between the quality of approximation and number of real analyses is what dictates the overall time needed for reaching the optimum (if at all reachable).

Derivation of the Two Point Exponential Approximation

Several traditional approximation methods were summarized in the paper by Fadel et al (Fadel, 1990) ranging from the simple Taylor series in the form:

$$g(X) = g(X_o) + \sum_i (x_i - x_{oi}) \frac{\partial g(X_o)}{\partial x_i}$$

to the reciprocal, hybrid, and higher order approximations. The authors then introduced the Two Point Exponential approximation which is an extension of the simpler Taylor series, adjusted by matching the derivatives at the previous design point. This correction term is incorporated into an exponent which is computed after each real analysis for each constraint, and with respect to each design variable. The exponent acts as a measure of goodness of fit: If the linear approximation is valid for a certain constraint, the exponent is close to or equal to 1, if the reciprocal approximation is more appropriate, the exponent approaches or is equal to -1. In other cases, the exponent varies between -1 and 1, correcting the approximation and improving the fit of the data.

The Two Point Exponential Approximation is derived as mentioned earlier by matching the slopes at previous design points. Initially, one substitutes x^{p_i} for x in the Taylor series:

$$g(X) = g(X_o) + \sum_i (x_i^{p_i} - x_{oi}^{p_i}) \frac{\partial g(X_o)}{\partial x_i^{p_i}}$$

and after resubstitution, one can write:

$$g(X) = g(X_o) + \sum_i \left(\left(\frac{x_i}{x_{oi}}\right)^{p_i} - 1\right) \frac{x_{oi}}{p_i} \frac{\partial g(X_o)}{\partial x_i}$$

with the exponent evaluated according to:

$$p_i = \frac{\log \left\{ \dfrac{\left(\dfrac{\partial g(X_1)}{\partial x_i}\right)}{\left(\dfrac{\partial g(X_o)}{\partial x_i}\right)} \right\}}{\log \left\{\dfrac{x_{1i}}{x_{oi}}\right\}}$$

The point X_1 refers to the design point at the previous iteration and X_o refers to the current design point from where the approximation is carried out. Note that at the first iteration, since no previous design history exists, a linear or reciprocal step is carried out, depending on the preference of the user.

The results reported in the earlier paper compared the linear, reciprocal and Two Point Exponential approximations on structural problems with stress and displacement constraints. Three problems of different sizes were used, namely the standard three bar truss problem, a 25 bar truss transmission tower, and a 52 bar truss tower. The results showed that the Two Point Exponential approximation generally displayed a much smoother behavior than the other two methods. It contributed to reducing the oscillations between successive iterations, and required less iterations to reach the optimum in most cases. The overhead involved in computing the exponents proved to be insignificant. The exponents have to be computed after each real analysis and used during the optimization of the approximate problem. Care has to be taken in the code development to avoid divisions by zero, and to avoid having to compute the logarithm of a negative number. In such cases, the algorithm should be written in a way that it would revert to a linear or reciprocal step.

Two Point Exponential Approximation and Frequency Constraints

After ascertaining the merit of the approximation in the case of stress and displacement constraints, it was suggested to test the method on frequency type constraints. The frequency constraints are generally highly non-linear, and further testing of the method was warranted to confirm its value for general structural optimization problems. For this purpose, two test problems of different complexity and size were selected. The approximation method is tested on both the problems, and results and conclusions are reported. Both problems were taken from the literature to ensure correctness.

Test problem: Cantilever Beam

The first test problem is taken from Pritchard and Adelman (Pritchard, 1990). The 193 inch long hollow cantilever beam with square cross section (Figure 1) has four design variables: the height and width of the beam cross-section, and the two wall thicknesses (sides, top and bottom). The beam is divided into ten elements. The first element near the base has a slightly different modulus of elasticity, but all other characteristics are uniform over the length of the beam. The dimensions and physical characteristics of the standard beam X_0 are:

H	=	5.00 in
B	=	3.75 in
t	=	0.80 in
d	=	0.10 in

and moduli of elasticity: (Element 1 is at the wall)

$$E_{2\text{-}10} = 5.85E6$$
$$E_1 = 4.90E6$$

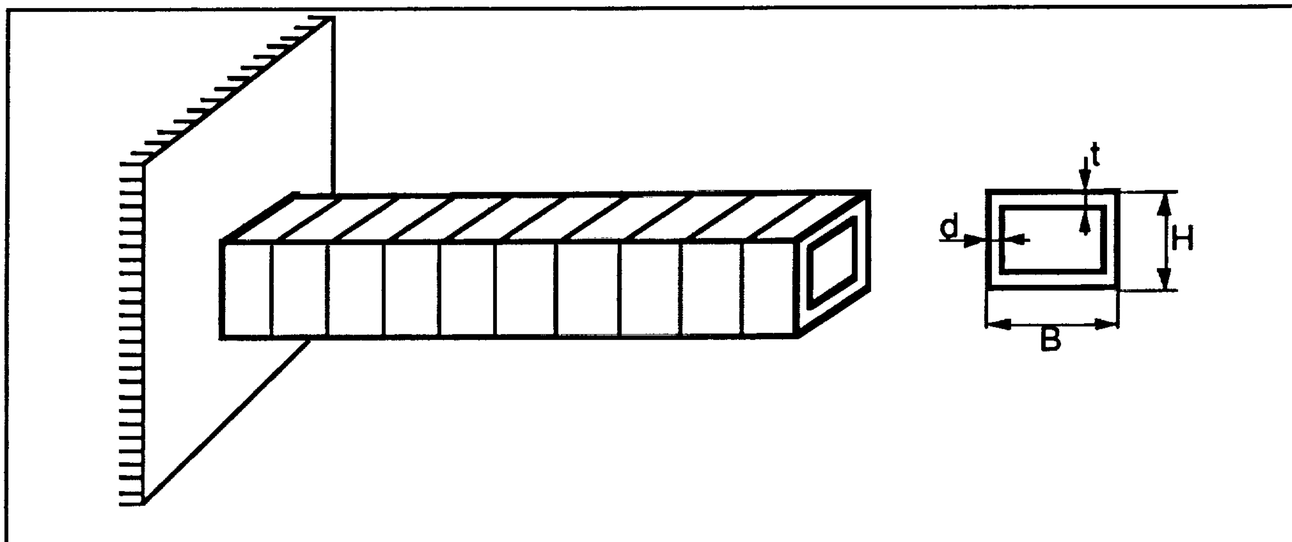

Figure 1. Cantilever Beam (Pritchard, 1990)

The problem was analyzed using the ANSYS (Swanson, 1990) finite element package, and optimizations were carried out with the program CONMIN (Vanderplaats, 1973). The first test consisted in evaluating the approximations to the first bending frequency when one variable was modified. The height of the beam cross section was selected as the design variable, and the results are tabulated in Table 1.

H	linear	reciprocal	exact	2 pt exp.	H	rel err lin [%]	rel err 2pt [%]
2	1.7303	-3.64094	1.67448	1.62987	2	1.050261153	0.83990199
3	2.9239	1.332393	2.88837	2.88399	3	0.668228188	0.08239831
4	4.1175	3.81906	4.10689	4.10836	4	0.199018652	0.027637706
5	5.3111	5.31106	5.31106	5.31106	5	0	0
6	6.5047	6.305727	6.49736	6.49678	6	0.137449021	0.010981808
7	7.6983	7.016203	7.66538	7.66855	7	0.619085456	0.059730265
8	8.8919	7.54906	8.81562	8.82852	8	1.435494986	0.242964664
9	10.085	7.963504	9.94882	9.97827	9	2.572744424	0.554566968
10	11.279	8.29506	11.0658	11.119	10	4.01539429	1.002004366
	dfdh0=	(H=5)	1.1936			p=	0.929378792
	dfdh1=	(H=6)	1.17833				

Table 1. Cantilever Beam analysis. Evaluation of approximations based on Beam Height H.

These results are illustrated in Figure 2. The errors resulting from both the linear and Two Point Exponential approximations are plotted as a function of the beam height H. The reference point X_0 is the point H=5, and the Two point Exponential approximation uses the previous analysis point at X_1 as the point where H=6. The graphs show the superior performance of the new approximation in the case of changes in one design variable.

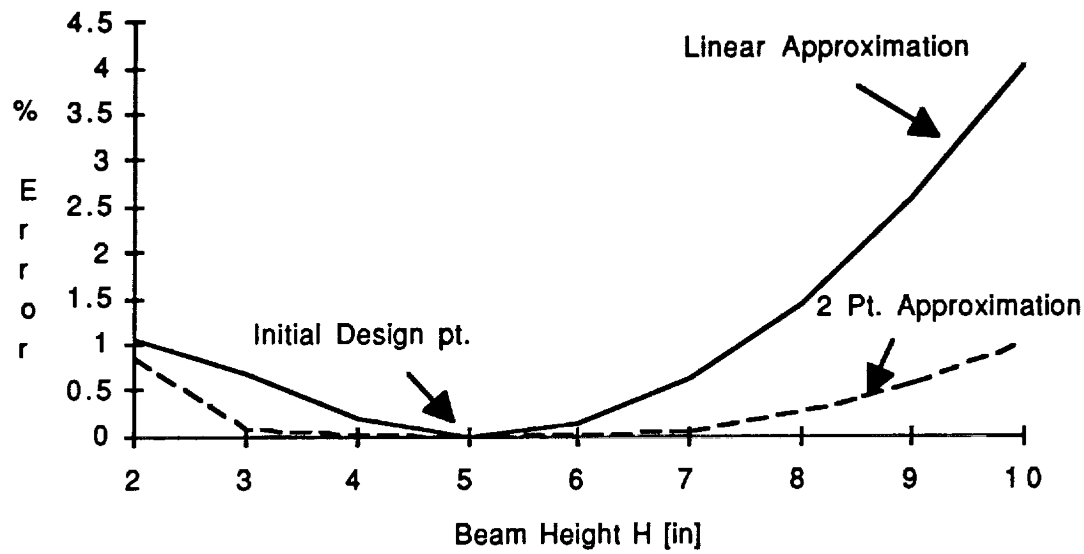

Figure 2. Approximation error as a function of beam height H
Cantilever Beam Problem

When one considers variations in multiple variables simultaneously, the advantages of one approximation versus another are less easily demonstrated. In this study, we first considered changing all four variables of the cantilever beam problem simultaneously by progressive percentages. Figure 3 illustrates the relative errors of the approximations of the first bending frequency as a function of the relative change in all four design variables. These changes are certainly not indicative of performance within an optimization exercise, but they do provide some measure of goodness easily displayable. Note that the starting point for the approximation is the point with abscissa 0. The linear approximation is carried out from this point forward (increasing all four design variables by x%), and then backward (decreasing all four variables by x%). For the Two Point Exponential approximation, the starting point is the same X_0, and the "previous" design point X_1 is at abscissa x=.2. In this case, increasing x means backtracking, whereas decreasing x means progressing in the direction established by the

two successive design points. The figure shows that the Two Point Exponential approximation seems to better fit the real function below the design point, and is slightly worse than the linear approximation above the design point. It is hoped that during an optimization, the design variables would either increase or decrease monotonically, and the Two Point Exponential approximation would perform better than the linear. Note that the results of both approximations were very sensitive to the derivatives obtained through finite differences in the analysis program (ANSYS).
The true test of an approximation however, is to perform the optimization exercise. This is the subject of the next section.

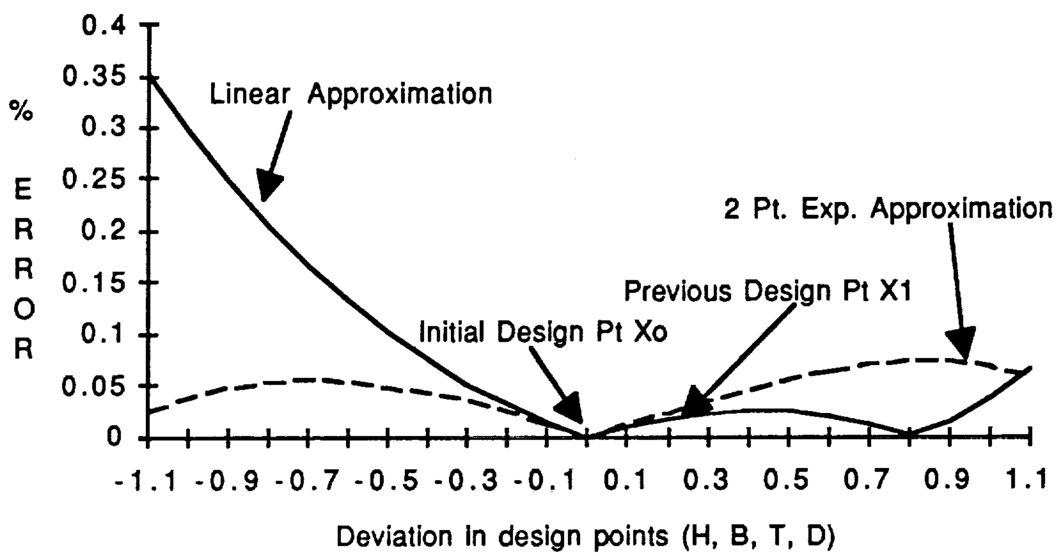

Figure 3. Approximation error as a function of all four variables
Cantilever Beam Problem

Optimization of the Cantilever Beam Problem.

Since a true test of the approximations can only be obtained in an optimization problem, the Cantilever Beam example discussed above was reformulated as an optimization exercise. The initial design variables are the ones given above as vector X_O, and the object of the problem is to find the minimal weight subject to frequency constraints. The first frequency constraint is the first bending frequency of the beam which has to be below a certain minimum value and the second frequency above another value. This would ensure a separation of natural frequencies, and could be used as a design problem. The first attempt to solve the problem considered two design variables, namely the height and width of the beam, leaving the thicknesses constant. The constraints (first and second frequencies) are limited to 5 Hz and 30 Hz respectively (F1 < 5Hz, F2 >30Hz). The allowable error is 0.1 and the move limits are 50% in all three cases. The results are tabulated below:

	Linear	Reciprocal	2 Pt. Exp.
0	6.68	6.68	6.68
1	3.60146	3.62224	3.60146
2	2.13858	2.16792	2.15689
3	1.51095	1.56556	1.56387
4	1.83129	1.55081	1.5507
5	1.52809	1.55081	1.5507
6	1.54845		
7			

Table 2. Variation of Cross sectional area as function of iteration number.

Because of the similarity of results, a graph of the variation of objective (cross sectional area) with respect to iteration number would not provide any additional information. From the table above, one can only deduce that in this particular case, the three approximations perform relatively similarly. All three reach the optimum in roughly the same amount of steps. The linear approximation seems to reach a smaller optimum, but this result is because this particular approximation in this problem causes one of the constraints to be slightly violated, and at the final result, the second frequency constraint is active, but very close to be violated, whereas in the two other methods, the second frequency constraint is active, and satisfied. Table 3 lists some of the results for the above problem. In all three cases, the beam width is driven to the minimum (0.5 in), and the second frequency constraint becomes active.

	Linear B	Linear F2	Reciprocal B	Reciprocal F2	2 Pt Exp B	2 Pt Exp F2
1	3.75	33.6109	3.75	33.6109	3.75	33.6109
2	1.875	29.6286	1.875	30.3661	1.875	29.6286
3	0.9375	29.1512	0.9375	30.0812	0.9375	29.7322
4	0.5	28.9012	0.5	30.4118	0.5	30.3652
5	0.75	28.1872	0.5	30.005	0.5	30.0019
6	0.5	29.3767	0.5	30.005	0.5	30.0019
7	0.5	29.9399				

Table 3 Cantilever Beam. Active constraints as function of iteration number. Beam width B driven to >= 0.5 in, second natural frequency driven to >= 30Hz.

When one considers all four parameters: height, width and thicknesses, as design variables, the problem should be more complicated and the approximations less well behaved.

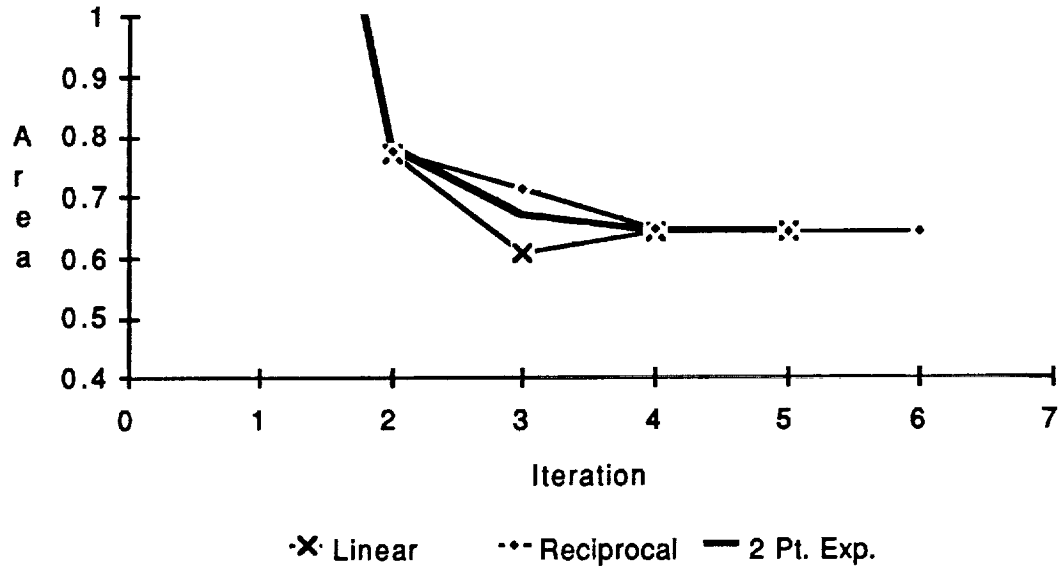

Figure 4. Cantilever beam. Cross sectional area (objective)
Four design variables, two constraints.

Figure 4. illustrates the variation of the objective with respect to iteration number in the case of four design variables. In this case again, all three approximations behave relatively similarly, with the linear and Two Point Exponential approximations reaching the minimum in 5 steps whereas the reciprocal this time, takes one additional step. The interesting observation however, is the path to the minimum taken by all three approximations. In order to show the differences, the value axis (area) was magnified with a maximum at 2 inches. The first two iterations are therefore not visible, but one can see that the Two Point Exponential approximation is the smoothest behaved function.

Figure 5. illustrates in the same problem (four design variables, two constraints), the variation of the second natural frequency. The problem consisted in minimizing the area subject to the second frequency remaining above 30Hz. The figure shows that the Two Point Exponential method shows similar oscillative behavior as the other methods, but with a smaller amplitude.

The two results described sofar show that for two relatively simple problems with frequency constraints, the Two Point Exponential approximation behaves at least as good, if not better than the best of the linear or reciprocal approximations.

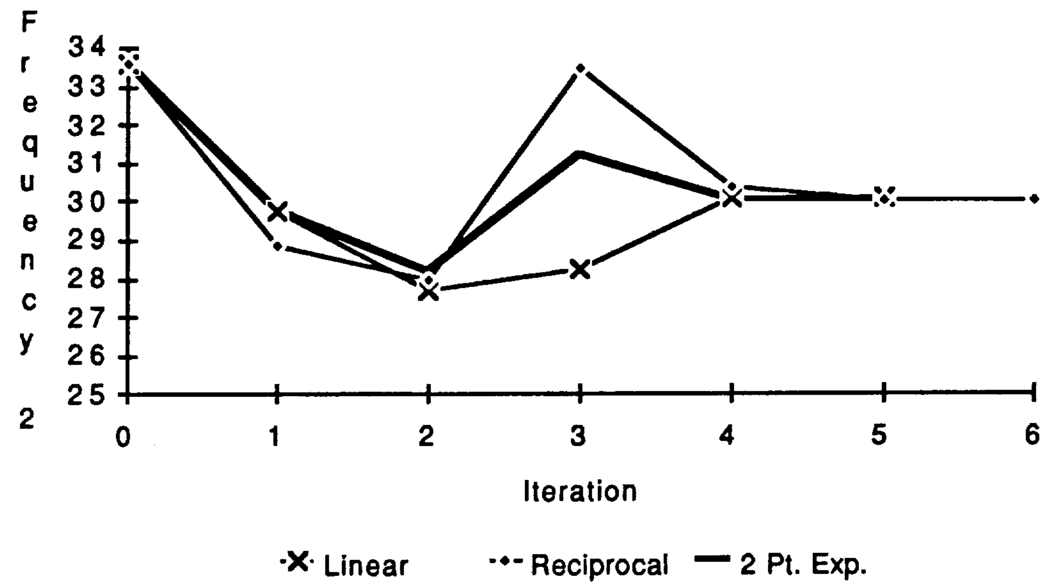

Figure 5. Cantilever beam problem, four design variables
Variation of second frequency constraint

Conclusion

The Two point Exponential Approximation was tested on problems with frequency constraints. The results obtained sofar show that the method is at least as performing as the best of the traditional methods like the linear or reciprocal approximation. It does also perform as a more controlled method which should be used when the problem to be solved does not have uniformly linearly behaved or uniformly reciprocally behaved constraints and objectives.

Acknowledgements

The author wishes to thank Dr. Jean Francois Barthelemy and Dr. Jacek Sobieski from NASA Langley for their comments and suggestions. This work was supported in part by NASA under contract NAG-1-1144.

References

Fadel, G.M., Riley, M.F. and Barthelemy, J.F.M., "Two Point Exponential Approximation Method for Structural Optimization", Structural Optimization, 2, 117-124, 1990

Pritchard, J.I., Adelman, H.M, "Differential Equation Based Method for Accurate Approximations in Optimization", NASA Technical Memorandum, 102639, AVSCOM Technical Memorandum 90-B-006, 1990.

Swanson Analysis Systems Inc, "ANSYS, Engineering Analysis System. User's Manual." Vols I and II, 1990

Vanderplaats, G.N, "CONMIN, a Fortran Program for Constrained Function Minimization. User's Manual." NASA Technical Memorandum, TM-X-62,282, 1973

APPENDIX A

Numerical Results of Optimization runs

Linear approximation			<=5	>=30	
	H	B	OBJ	CON1	CON2
0	5	3.75	6.68	5.31106	33.6109
1	4.60731	1.875	3.60146	4.68022	29.6286
2	4.79292	0.9375	2.13858	4.60463	29.1512
3	5.15476	0.5	1.51095	4.56506	28.9012
4	4.75644	0.75	1.83129	4.45203	28.1872
5	5.24046	0.5	1.52809	4.64033	29.3767
6	5.34226	0.5	1.54845	4.7295	29.9399
7					
Reciprocal Approximation					
0	5	3.75	6.68	5.31106	33.6109
1	4.71122	1.875	3.62224	4.797	30.3661
2	4.93959	0.9375	2.16792	4.75187	30.0812
3	5.4278	0.5	1.56556	4.80423	30.4118
4	5.35406	0.5	1.55081	4.73982	30.005
5	5.35406	0.5	1.55081	4.73982	30.005
6					
Two Point Exponential Approximation					
0	5	3.75	6.68	5.31106	33.6109
1	4.60731	1.875	3.60146	4.68022	29.6286
2	4.88447	0.9375	2.15689	4.69663	29.7322
3	5.41935	0.5	1.56387	4.79686	30.3652
4	5.35349	0.5	1.5507	4.73932	30.0019
5	5.35349	0.5	1.5507	4.73932	30.0019
6					
7					

Cantilever Beam results 2 variables 2 constraints

	H	B	T	D	OBJ	CON1	CON2
				Linear		<=5	>=30
1	5	3.75	0.8	0.1	6.68	5.31106	33.6109
2	4.25085	1.875	0.4	0.05	1.84509	4.69586	29.7274
3	4.3655	0.9375	0.2	0.05	0.77155	4.37265	27.6857
4	5.27703	0.5	0.1	0.05	0.6077	4.46111	28.2445
5	5.65837	0.5	0.1	0.05	0.64584	4.75696	30.1133
6	5.65837	0.5	0.1	0.05	0.64584	4.75696	30.1133
7							
				Reciprocal			
1	5	3.75	0.8	0.1	6.68	5.31106	33.6109
2	4.13298	1.875	0.4	0.05	1.8333	4.56184	28.8809
3	4.41934	0.9375	0.2	0.05	0.77693	4.42169	27.9955
4	6.34962	0.5	0.1	0.05	0.71496	5.2913	33.4862
5	5.70696	0.5	0.1	0.05	0.6507	4.7946	30.351
6	5.63605	0.5	0.1	0.05	0.64361	4.73967	30.0041
7	5.63605	0.5	0.1	0.05	0.64361	4.73967	30.0041
8							
				2 Pt Exponential			
1	5	3.75	0.8	0.1	6.68	5.31106	33.6109
2	4.25085	1.875	0.4	0.05	1.84509	4.69586	29.7274
3	4.45606	0.9375	0.2	0.05	0.78061	4.45508	28.2064
4	5.84612	0.521375	0.1	0.05	0.66889	4.92356	31.1652
5	5.63885	0.5	0.1	0.05	0.64389	4.74184	30.0178
6	5.63885	0.5	0.1	0.05	0.64389	4.74184	30.0178

Cantilever Beam results. 4 variables 2 constraints

APPENDIX B
Ansys input file and Program listing

```
H=5.
B=3.75
T=0.8
D=0.1
/TITLE, Beam model for approximation testing  3D model
FINISH
/PREP7
KAN,2
KAY,1,-1
KAY,2,3
KAY,7,3
C***  compute area and IZZ
IYY1=(T**3)*B
IYY2=IYY1/12
PAR1=H-(T*2)
PAR3=B-(D*2)
PAR2=(H-T)/2
IYY3=(T*B)*(PAR2**2)
IYY4=(IYY2+IYY3)*2
IYY5=D*(PAR1**3)
IYY6=IYY5/6
IYY =IYY6+IYY4
AREA=((T*B)+(PAR1*D))*2
C***  end of calculations
ET,1,3                       * 2D elastic beam
R,1,AREA,IYY,H
MP,EX,1,4.9e6                * material properties for element 1
MP,DENS,1,0.00018
MP,EX,2,5.85e6               * material properties for other elements
MP,DENS,2,0.00018
N,1,0
N,11,193
FILL
/PNUM,NODE,1
NPLOT
MAT,1
E,1,2
MAT,2
E,2,3
EGEN,9,1,2
EPLOT
D,1,ALL
M,2,UY,11,UX,ROTZ
SAVE
ITER,1,1
SFWRITE
FINISH
/SOLVE
FINISH
```

```
/POST1
set,,1
*get,fre1,freq
set,,2
*get,fre2,freq
set,,3
*get,fre3,freq
FINISH
/OPT
FACT=.99999
H1=H*FACT
H2=H/FACT
B1=B*FACT
B2=B/FACT
OPVAR,H,DV,H1,H2
OPVAR,B,DV,B1,B2
OPVAR,AREA,OBJ
OPVAR,PAR1,SV,.1,H
OPVAR,PAR3,SV,.1,B
OPVAR,FRE1,SV,.1,10
OPVAR,FRE2,SV,.1,100.
OPVAR,FRE2,SV,.1,150.
OPCOPY
H=H*1.001
RUN,2
B=B*1.001
H=H/1.001
RUN,3
T=T*1.001
B=B/1.001
RUN,4
D=D*1.001
T=T/1.001
RUN,5
OPLIST,ALL,,1
FINISH
/EOF
```

```
C
C23456789012345678901234567890123456789012345678901234567890123456789012
C       1         2         3         4         5         6         7
C
C      program to read an ANSYS file and extract the necessary data for
C      optimization, call conmin, and use approx to solve approximate prob
C
C      Georges Fadel   Sept 1990
C                      Oct  1990
C                      Jan  1991
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C           commons for CONMIN call
      COMMON /CNMN1/ DELFUN,DABFUN,FDCH,FDCHM,CT,CTMIN,CTL,CTLMIN,
     1 ALPHAX,ABOBJ1,THETA,OBJ,NDV,NCON,NSIDE,IPRINT,NFDG,NSCAL,LINOBJ,
     2 ITMAX,ITRM,ICNDIR,IGOTO,NAC,INFO,INFOG,ITER
      COMMON /CNMN2/ X(6),DF(6),G(15),ISC(15),IC(15),A(6,15),AF(7)
      COMMON /CNMN4/ VLB(6),VUB(6),SCAL(6)
C     the next two are for approx subroutine.  Second common just to pass
C     flags to approx
      COMMON /INFOIN/ DV(4), FUNC(7), GRAD(4,7)
      COMMON /INFOLD/ DV1(4), FUNC1(7), GRAD1(4,7)
      COMMON /FLAGS/ IFLAG,ICALL,IDEBUG
      DIMENSION P(4,7), RATX(4), RATDER(4,7)
      DIMENSION S(6),G1(15),G2(15),B(15,15),C(15),MS1(30)
C          end of conmin non-executable
      DIMENSION CONS(5,5),OOBJ(4),GMAX(6)
      CHARACTER*4 START(5),T(5)
      CHARACTER*12 FILNM,FILNM1,FILNM2,FILNM3
      CHARACTER*80 TT
      LOGICAL TOF
      DATA START(1),START(2),START(3),START(4),START(5)/'LIST',' OPT',
     1 'IMIZ','ATIO','N SE'/
C      name of file (File='        ') written from batch file into
C          temp.dat is read into FNAME.
C
C     some parameters that have to be set for each optimization program:
C          nlines in output file
C          number of design variables NDV
C          Number of constraints NCON
C          Increment factor used to compute finite differences in
C          finite element program: FACT = 1. - actual FACT
C          DF means derivative of objective wrt design variable
C          A means derivative of constraint wrt design variable
C          and remember to adjust dimensions to read all needed data
C          in X(NDV), CONS(NCON,NDV), OOBJ(NDV)
C          DF(NDV),A(NDV,NCON)
C
C          Also, the output data includes a maximum of 6 cases per row.  If
C          NCON is more than 6, then, an additional read statement has to b
C          written for the next batch of results.
C
C      conmin requirements +++++++++++++++++++++++++++++
C       IGOTO    Sets start of optimization loop
C       IPRINT   Print control: 0 print nothing
C                               1 print initial and final function informat
C                               2 1st debug level print 1 + control paramet
C                                 function value and X at each iteration.
C                               3 2nd debug level print 2 + constraints, ac
C                                 or violated constraints, move parameters.
```

```
C                                         approaches 0 as optimum gets closer
C                                       4 full debug
C       NDV       Number of decision variables
C       ITMAX     Max number of iterations
C       NCON      Number of constraint functions G(J)
C       NSIDE     Number of side constraints (upper, lower bounds)
C       G         Constraints at initial design point
C       CONS      Constraints at finite differences
C       ICNDIR    Conjugate direction restart parameter
C       NSCAL     Scaling control parameter
C       NFDG      Gradient calculation control parameter 0: calculated by F
C                                                        1: externally supp
C                                                        2: obj external, r
C       FDCH      Relative change of decision variable for FD calc.
C       FDCHM     Minimum step for FD
C       CT        Constraint thickness parameter
C       CTMIN     Minimum abs value of CT
C       CTL       Constraint thickness for linear and side constraints
C       CTLMIN    Minimum abs value of CTL
C       THETA     Mean value of push off factor(for highly non-linear probl
C       NACMX1    Estimate of number of active constraints
C       DELFUN    Minimum change in OBJ to indicate convergence
C       DABFUN    Same as DELFUN, but absolute not relative error
C       LINOBJ    0 means non-linear, 1 means linear
C       ITRM      (3) number of consecutive iterations for convergence
C       X(N1)     Vector of decision variables
C       VLB(N1)   Lower bound on variables X(I)
C       VUB(N1)   Upper bound on variables X(I)
C       SCAL(N5)  Vector of scaling parameters not used if NSCAL=0
C       ISC(N2)   Linear constraint identification vector
C
C       GMAX(NCON) LIMITS OF CONSTRAINTS
C
C      IPRINT=2   SUPPLIED IN EXTERNAL FILE
C      NDV=4      SUPPLIED IN EXTERNAL FILE
C      SET NUMBER OF CONSTRAINTS TO REQUIRED NUMBER (INITIALLY 1, THEN 6)
C      NCON=1     SUPPLIED IN EXTERNAL FILE

      IGOTO=0
      NFDG=0
      ITMAX=50
C     NACMX1=15
      NSIDE=8
      ICNDIR=0
      NSCAL=0
      LINOBJ=0
      N1=6
      N2=15
      N3=15
      N4=15
      N5=30
      ITRM=3
      FDCH=0.
      FDCHM=0.
      CT=0.
      CTMIN=0.
      CTL=0.
      CTLMIN=0.
      THETA=0.
      DELFUN=10.E-8
```

```
      DABFUN=10.E-8
      NAC=0
      ALPHAX=0.1
      ABOBJ1=0.1
      ICALL=1
      DO 9 I=1,N2
        ISC(I)=0
 9    CONTINUE
C     end of conmin variables definition
C          +++++++++++++++++
      NLINES=20000
      FACT=0.001
C          +++++++++++++++++  MAKE SURE THIS IS CORRECT  TOTAL NUMBER OF
C                             AND INCREMENT FACTOR IN ANSYS FOR DERIVATIV

C     create a file called OPTIM.DAT in which the filenames of the
C     initial result data file and file to be used to store results are
C     written. One name on each line.  Next, enter a number representing
C     the magnitude of the move limits in %
C
      OPEN(UNIT=3,STATUS='OLD',FILE='OPTIM.DAT')
         read(3,99)FILNM,FILNM1,FILNM2,FILNM3
         read(3,98)GMOVE
         read(3,*)NDV,NCON
         read(3,*)IDEBUG,IPRINT
         read(3,*)(GMAX(I),I=1,NCON)
      CLOSE(UNIT=3)

      OPEN(UNIT=8,STATUS='OLD',FILE=FILNM)
      OPEN(UNIT=9,ACCESS='TRANSPARENT',FORM='UNFORMATTED',FILE=FILNM1)
      OPEN(UNIT=10,STATUS='OLD',FILE='HISTORY.DAT')
      OPEN(UNIT=11,STATUS='OLD',FILE='FLAGS.DAT')

C ******************************************************************
C     READ FILNM TO EXTRACT INFO FOR DERIVATIVES CALCULATION
C

      READ(8,100)T(1),T(2),T(3),T(4),T(5)
      DO 1000 L=1,NLINES
C         find first line of results
          IF(T(1).NE.START(1).OR.T(5).NE.START(5)) THEN
              READ(8,100)T(1),T(2),T(3),T(4),T(5)
          ELSE
              READ(8,101)
C             read some blank lines to get to beginning of data
C             initially do 10 i=1,ndv  THIS SHOULD BE ACCORDING TO FILE
              if(idebug.ge.3) PRINT *,' DESIGN VARIABLES  '
              DO 10 I=1,4
C                 read the design variables X(I)
                  READ(8,102) X(I)
                  if(idebug.ge.3) PRINT *, X(I)
C                 Compute the move limits
                  VLB(I)=X(I)*(1.-GMOVE/100.)
                  VUB(I)=X(I)*(1.+GMOVE/100.)
                  if(idebug.ge.3) PRINT *, VLB(I),VUB(I)
10            CONTINUE
C
C             ADD THE FOLLOWING LOWER BOUNDS FOR PROBLEM TO BE REALISTIC
C
              IF(VLB(1).LE.2.) VLB(1)=2.0
```

```
            IF(VLB(2).LE.0.5) VLB(2)=0.5
            IF(VLB(3).LE.0.1) VLB(3)=0.1
            IF(VLB(4).LE.0.05) VLB(4)=0.05
C
C           and upper limits
            IF(VUB(1).GE.15.) VUB(1)=15.0
            IF(VUB(2).GE.15.) VUB(2)=15.0
            IF(VUB(3).GE.(X(2)/2.)) VUB(3)=X(2)/2.
            IF(VUB(4).GE.(X(1)/2.)) VUB(4)=X(1)/2.

C           Read some more blank lines
            READ(8,103)
C           and then the Objective function at the design point OBJ
C           and the objective at finite differences from the origin
            READ(8,104)OBJ,(OOBJ(J),J=1,NDV)
            if(idebug.ge.1) THEN
                 PRINT *,' OBJECTIVE AND RESULTS OF FDs '
                 PRINT *, OBJ,(OOBJ(J),J=1,NDV)
            ENDIF
C           convert constraints into <=0 constraints and scale
            if(idebug.ge.1)PRINT *,' CONSTRAINTS AND RESULTS OF FDs '
            DO 11 J=1,NCON
                 READ(8,104) G(J),(CONS(J,K),K=1,NDV)
                 if(idebug.ge.1) PRINT *, G(J),(CONS(J,K),K=1,NDV)
                 G(J)=G(J)/GMAX(J)-1.
                 IF(J.EQ.2) G(J)=-G(J)
                 DO 13 KK=1,NDV
                     CONS(J,KK)=CONS(J,KK)/GMAX(J)-1
                     IF(J.EQ.2) CONS(J,KK)=-CONS(J,KK)
13               CONTINUE
                 if(idebug.ge.1) PRINT *,' CORR ',G(J),(CONS(J,K)
     1           ,K=1,NDV)
11          CONTINUE
C
C           Now compute the derivatives:
C
            DO 12 I=1,NDV
                  DF(I)=(OOBJ(I)-OBJ)/X(I)/FACT
                  if(idebug.ge.1) PRINT *,'OBJ DER ',DF(I)
                  DO 12 J=1,NCON
                     A(I,J)=(CONS(J,I)-G(J))/X(I)/FACT
                     if(idebug.ge.1) PRINT *,' DERIV ',A(I,J)
12          CONTINUE

C           w r i t e    values to confirm
            WRITE(9)NDV,(X(I),I=1,NDV),OBJ,NCON,(G(J),J=1,NCON),
     1      (DF(II),II=1,NDV),((A(K,M),K=1,NDV),M=1,NCON)
            if(idebug.ge.2) THEN
                 print *,' SUMMARY '
                 print *,NDV,(X(I),I=1,NDV)
                 print *,OBJ,NCON,(G(J),J=1,NCON)
                 print *,(DF(II),II=1,NDV)
                 print *,((A(K,M),K=1,NDV),M=1,NCON)
            ENDIF
C           replace values into conmin arrays and form.  they will
C           be passed to approx through common.
            FUNC(1)=OBJ
            DO 20 I=1,NDV
                 DV(I)=X(I)
                 GRAD(I,1)=DF(I)
```

```
                 DO 20 JJ=1,NCON
                      GRAD(I,JJ+1)=A(I,JJ)
 20            CONTINUE
               DO 21 J=1,NCON
                    FUNC(J+1)=G(J)
 21            CONTINUE
               GOTO 999
           ENDIF
1000   CONTINUE

       CLOSE(UNIT=8)
C
 999   CONTINUE
C       INITIALIZE CONSTRAINT IDENTIFICATION VECTOR, ISC.
       DO 310 J=1,NCON+1
 310      ISC(J)=0

C**********************************************************************

C      SOLVE OPTIMIZATION.

 350   CONTINUE
       if(idebug.ge.2)print *,'before conmin',X(1),X(2),X(3),X(4)

       CALL CONMIN(X,VLB,VUB,G,SCAL,DF,A,S,G1,G2,B,C,ISC,IC,MS1,
     1 N1,N2,N3,N4,N5)

       if(idebug.ge.2)print *,'after conmin',X(1),X(2),X(3),X(4)
       IF(IGOTO.EQ.0) THEN

C          reached optimum

           if(idebug.ge.2)then
                print *, 'final results'
                print *, ' '
                print *, 'OBJECTIVE = ',OBJ
                print *,' X VECTOR ',(X(I),I=1,NDV)
                print *,' G VECTOR ',(G(J),J=1,NCON)
           endif
           WRITE(10,*) OBJ,(X(I),I=1,NDV),(G(J),J=1,NCON)
C          write info to new file to rerun ansys
C          first, we have to read the input file for ansys and then rewrite
C          it with new values
           OPEN(UNIT=4,STATUS='OLD',FILE=FILNM2)
           OPEN(UNIT=5,STATUS='UNKNOWN',FILE=FILNM3)
C          READ AND WRITE FILE
           WRITE(5,110)(X(I),I=1,NDV)
           DO 363 NN=1,NDV
              READ(4,*) TT
363        CONTINUE
           DO 361 NN=1,NLINES
              READ(4,111,END=362) TT
              WRITE(5,111)TT
361        CONTINUE
362        CONTINUE
           ICALL=1
           REWIND(11)
           WRITE(11,112)ICALL,IFLAG
           CLOSE(UNIT=11)
```

```
            S   T   O   P

          ELSE

C            no convergence yet ...
             rewind(11)
             READ(11,112)ICALL,IFLAG
             IF(ICALL.EQ.1) THEN
C                first call to approximation, copy file and compute exponent
C               IFLAG=    1  LINEAR
C                         2  RECIPROCAL
C                         3  TWO POINT EXPONENTIAL
C
                 ICALL=0
                 REWIND(11)
                 WRITE(11,112)ICALL,IFLAG
                 IFLAGT=IFLAG

                 IF(IFLAG.EQ.3) THEN
                    INQUIRE(FILE='SCNDGRD.DAT',EXIST=TOF)
                    IF(TOF) THEN
                      OPEN(UNIT=7,ACCESS='TRANSPARENT',FORM='UNFORMATTED'
     1                ,STATUS='OLD',FILE='SCNDGRD.DAT')
                      READ(7)NDV,(DV1(I),I=1,NDV),FUNC1(1),NCON,(FUNC1(J)
     1                ,J=2,NCON+1),(GRAD1(L,1),L=1,NDV),((GRAD1(K,M)
     2                ,K=1,NDV),M=2,NCON+1)
                      if(idebug.ge.4) then
                        print *, 'old point: ',(DV1(I),I=1,NDV)
                        print *, 'old obj. ', FUNC1(1)
                        print *, 'old constr ',(FUNC1(J),J=2,NCON+1)
                        print *, 'old grads ',((GRAD1(K,M),K=1,NDV)
     1                     ,M=1,NCON)
                      endif
                      REWIND(7)
                      WRITE(7)NDV,(DV(I),I=1,NDV),FUNC(1),NCON,(FUNC(J)
     1                ,J=2,NCON+1),(GRAD(L,1),L=1,NDV),((GRAD(K,M)
     2                ,K=1,NDV),M=2,NCON+1)
                      if(idebug.ge.4) then
                        print *, 'Xo point: ',(DV(I),I=1,NDV)
                        print *, ' obj. ', FUNC(1)
                        print *, ' constr ',(FUNC(J),J=2,NCON+1)
                        print *, ' grads ',((GRAD(K,M),K=1,NDV)
     1                     ,M=1,NCON)
                      endif
                      CLOSE(UNIT=7)

                    ELSE

C                     first call, no data in SCNDGRD.DAT yet.  put it in
                      IFLAG=1
                      OPEN(UNIT=7,ACCESS='TRANSPARENT',FORM='UNFORMATTED'
     1                ,STATUS='NEW',FILE='SCNDGRD.DAT')

                      WRITE(7)NDV,(DV(I),I=1,NDV),FUNC(1),NCON,(FUNC(J)
     1                ,J=2,NCON+1),(GRAD(L,1),L=1,NDV),((GRAD(K,M)
     2                ,K=1,NDV),M=2,NCON+1)
                      CLOSE(UNIT=7)
                    ENDIF
                 ENDIF
                 NFUNCS=NCON+1
```

```
C                -----------------------------------------------------------
C               COMPUTATION OF EXPONENT BASED ON IFLAG
C
                DO 710 I=1,NDV
                  IF(DV(I).EQ.0.) THEN
                      RATX(I)=1.E8
                  ELSE
                      RATX(I)=DV1(I)/DV(I)
                  ENDIF
                  if(idebug.ge.3) THEN
                   PRINT *,'INITIAL CALCULATIONS IFLAG= ',IFLAG,'X(I) = '
     1             ,X(I), I, 'DV1(I)/DV(I) ',RATX(I)
                  ENDIF
                  IF((IFLAG.EQ.1).OR.(X(I).EQ.0.).OR.(RATX(I).EQ.1.))
     1            THEN
                    if(idebug.ge.2) print *,' In linear code '
C                   LINEAR APPROXIMATION
                    DO 711 J=1,NFUNCS
                        P(I,J)=1.
  711               CONTINUE
                  ELSE
                    IF((IFLAG.EQ.2).OR.(DV(I).EQ.0.)) THEN
                        if(idebug.ge.2)print *,' In Reciprocal code '
C                       RECIPROCAL APPROXIMATION
                        DO 712 J=1,NFUNCS
                            P(I,J)=-1.
  712                   CONTINUE
                    ELSE
                        if(idebug.ge.2)print *,' In 2 point code '
C                       2 POINT EXPONENTIAL APPROXIMATION
                        DO 713 J=1,NFUNCS
                            IF(GRAD(I,J).EQ.0.) THEN
                               P(I,J)=1.
                            ELSE
                               RATDER(I,J)=GRAD1(I,J)/GRAD(I,J)
                               IF((RATX(I).LE.0.).OR.(RATDER(I,J).LE.0.))
     1                            THEN
                                  P(I,J)=1.
                               ELSE
                                 P(I,J)=DLOG(RATDER(I,J))/DLOG(RATX(I))+1
                                 IF(P(I,J).GE.1.) THEN
                                    P(I,J)=1.
                                 ELSE
                                    IF(P(I,J).LE.-1.) P(I,J)=-1.
                                 ENDIF
                               ENDIF
                            ENDIF
  713                   CONTINUE
                    ENDIF
                  ENDIF
                  if(idebug.ge.2)PRINT *,'I, EXPONENT  *****',I,P(I,1)
  710           CONTINUE
C            -------------------------------------------------------------
            IFLAG=IFLAGT
            ENDIF
            IF(INFO.EQ.1) THEN
                AF(1)=FUNC(1)
                if(idebug.ge.3) PRINT*,'OBJ ',obj
                DO 359 J=1,NCON
```

```
                AF(J+1)=FUNC(J+1)
                if(idebug.ge.3)PRINT*,'CONS #',j, G(J)
 359          CONTINUE
              if(idebug.ge.2)PRINT *,'CALL TO APPROXIMATION '
C             this is the call to the approximation
              CALL APPROX(X,AF,P,NDV,NCON)
C                 Resubstituting values in OBJ and CONS
              OBJ=AF(1)
              if(idebug.ge.3) PRINT *,'OBJ ',obj
              DO 360 J=1,NCON
                 G(J)=AF(J+1)
                if(idebug.ge.3) PRINT *,'CONS #',j, G(J)
 360          CONTINUE
          ELSE
              if(idebug.ge.3)PRINT *, '# Info ne 1 ???  ',INFO
          ENDIF
      ENDIF
      GOTO 350
C                       F    O    R    M    A    T    S
98    FORMAT(F3.0)
99    FORMAT(A12/A12/A12/A12)
100   FORMAT(5A4)
101   FORMAT(1X,//)
102   FORMAT(5X,E12.6)
103   FORMAT(1X,/////////)
104   FORMAT(5X,6E13.6)
110   FORMAT('H=',E12.6/'B=',E12.6/'T=',E12.6/'D=',E12.6)
C110   FORMAT('H=',E12.6/'B=',E12.6)
111   FORMAT(A80)
112   FORMAT(2I2)
C
      END
      SUBROUTINE APPROX(AV,AF,P,NDV,NCON)
C
C      THIS SUBROUTINE IS CALLED FROM OPTRUN TO PERFORM
C      VARIOUS APPROXIMATIONS OF THE FUNCTIONS (OBJECTIVES
C      AND CONSTRAINTS).  A FLAG WILL SELECT LINEAR,
C      RECIPROCAL OR IMPROVED APPROXIMATION.  TWO SETS OF DATA
C      ARE NEEDED SINCE THE IMPROVED APPROXIMATION RELIES ON
C      PAST ANALYSES TO IMPROVE THE APPROXIMATION.
C
C      Georges Fadel June 1989
C                    Oct  1990
C                    Jan  1991
C     AV is the vector of VARIABLES
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION AV(4), AF(7), P(4,7)
      COMMON /INFOIN/ DV(4), FUNC(7), GRAD(4,7)
      COMMON /FLAGS/ IFLAG,ICALL,IDEBUG
      NFUNCS=NCON+1
C-------------------------------------------------------------
C     NOW THE EXPONENT IS KNOWN, LETS COMPUTE THE APPROXIMATING FUNCTION
C
      if(idebug.ge.1) PRINT*, 'EXPONENT KNOWN '
      DO 30 J=1,NFUNCS
         DO 20 I=1,NDV
           IF((P(I,J).EQ.1.).OR.(ABS(P(I,J)).LE.0.00001))THEN
              AF(J)=AF(J)+(AV(I)-DV(I))*GRAD(I,J)
           ELSE
```

```
            IF(P(I,J).EQ.-1.) THEN
               AF(J)=AF(J)+(AV(I)-DV(I))*(DV(I)/AV(I))*GRAD(I,J)
            ELSE
               AF(J)=AF(J)+((AV(I)/DV(I))**P(I,J)-1.)
     1             *DV(I)*GRAD(I,J)/P(I,J)
            ENDIF
         ENDIF
   20    CONTINUE
   30 CONTINUE
  999 RETURN
      END
```

APPENDIX C
OPTIM.DAT file

```
BEAM4.OUT
BEAM4.INP
BEAM4.OLD
BEAM4.DAT
50.
4
3
0 4
5.0 30. 80.
```

APPENDIX D

OPTIMI.BAT batch file to execute optimization

```
echo off
cls
echo   OPTIMIZATION BATCH FILE TO TEST APPROXIMATIONS WITH ANSYS AND CONMI
echo ▓
echo             ...... START OF OPTIMIZATION PROCEDURE ......
echo ▓
echo          call to ANSYS with initial design variables set in file
echo                    xxxxxxx.dat in ansys format
ERASE %1.OUT
echo ▓
call ANSYS -I %1.dat -O %1.out
COPY %1.DAT %1.OLD
echo ▓
echo        First ANSYS run COMPLETED.  Results are written to %1.out
echo ▓
:loop1
call browse %1.out
echo ▓
echo               +++++++++++++  IN LOOP ++++++++++++++++
echo ▓
echo      call optimization program using design variables and derivatives
echo ▓
call optrun2
echo ▓
echo ▓
echo            ======== CONVERGED IN APPROXIMATION LOOP =========
copy hist.dat+history.dat hist.dat
echo ▓
ERASE %1.OUT
call ANSYS -I %1.DAT -O %1.out
echo ▓
echo                        RERUN ANALYSIS (ANSYS)
echo ▓
REM if not converged
goto loop1
REM else
stop
```