determination of the proper state of the thermosphere poses no fundamental obstacle to studies of aerodynamic structures in the thermosphere.

References

[1] *1962 U.S. Standard Atmosphere,* U.S. Government Printing Office, Washington, D.C., 1962.

[2] Alden, H. L. and Habert, R. H., "Gasdynamics of High Altitude Rocket Plumes," MITHRAS Rept. MC-63-80-R1, July 1964.

[3] Jarvinen, P. O. and Hill, J. A. F., "Universal Model for Underexpanded Rocket Plumes in Hypersonic Flow," 12th JANNAF Liquid Propulsion Meeting, Las Vegas, Nevada, Nov. 1970.

[4] Draper, J. S. and Moran, J. P., "A Study of Wind Tunnel Simulation of High Altitude Rocket Plumes," Tech. Rept. TR-72-111, Feb. 1973, Air Force Rocket Propulsion Laboratory (AFRPL), Edwards Air Force Base, Calif.

[5] Draper, J. S. and Sutton, E. A., "A Nomogram for High-Altitude Plume Dimensions," *Journal of Spacecraft and Rockets,* Vol. 10, Oct. 1973, pp. 682–684.

[6] Jacchia, L. G., "Revised Static Models of the Thermosphere and the Exosphere with Empirical Temperature Profiles," Special Rept. 332, May 1971, Smithsonian Astrophysical Observatory, Washington, D.C.

[7] *U.S. Standard Atmosphere Supplements, 1966,* Washington, D.C., 1966.

[8] *Solar-Geophysical Data* (Prompt Reports), No. 324-PART I, Aug. 1971, U.S. Department of Commerce, National Oceanic and Atmospheric Administration, Environmental Data Service, Washington, D.C.

Performance of the "Direct Search Design Algorithm" as a Numerical Optimization Method

MICHAEL PAPPAS*
New Jersey Institute of Technology, Newark, N.J.

Introduction

MATHEMATICAL programing (MP) optimization procedures have broad applicability in engineering and have found increasing acceptance in many areas. Such procedures, by means of an optimal search strategy, solve, or attempt to solve the problem: Find those values $\bar{x}_n$ of the variables x_n that minimize (or maximize) the "objective" function $f(x_n)$. That is, find

$$f(\bar{x}_n) = \min[f(x_n)] \qquad (1)$$

Often the solution is constrained by one or more relations that can be written in the form

$$g_j(x_n) \leqq 0 \qquad j = 1, 2, \ldots J \qquad (2)$$

Constrained linear problems can be solved efficiently and reliably. Nonlinear problems are, in contrast, generally quite difficult and, in fact, no method exists that will guarantee a solution to the general constrained nonlinear MP problem.

Still, many algorithms for the solution of nonlinear problems have been proposed and successfully applied. The Direct Search Design Algorithm (DSDA), is a constrained nonlinear, mathematical programing procedure, proposed for the automated optimal synthesis of structures, which has been successfully applied to a number of structural problems.[1-4] Since the method has worked well in several applications the question arises: How does the DSDA compare with other nonlinear techniques that have also been successfully used for engineering design?

An excellent recent study by Eason and Fenton[5] evaluating the performance of seventeen different optimization codes on ten constrained nonlinear problems, provides a convenient benchmark against which the DSDA can be compared. This paper, therefore, attempts to provide an answer to the previous question by extending the work of Eason and Fenton to include an evaluation of the performance of the DSDA on these same problems.

Study and Results

A general FORTRAN IV code based on the DSDA using the original variable step pattern search, penalty function, and optimality check described in Ref. 1 was employed in this study. A similar code was also used for the study described in Ref. 6. This code requires the user to supply only the objective and constraint functions, although the convergence criteria and initial step size may also be specified. If no step size is specified a step size is internally generated.

All ten problems of Refs. 5 and 7 were run with the DSDA code from the starting points specified in these references. The internally generated step sizes were used for all problems and no algorithm control constants were changed during the study. Thus there was no tuning of the code to individual problems. All problems were run on an IBM 370 model 91 using the "G" level compiler and single precision.

Table 1 briefly describes the codes studied in Refs. 5 and 7 with the addition of the DSDA studied here. Additional details on these codes can be found in Refs. 1, 5, and 7. Table 2 presents results of the performance of these codes on the ten test problems in the form of a success matrix. The numerical entries indicate the normalized time required for solution,[5] the symbol P indicates progress toward a solution and a blank indicates failure. Definitions of "solution" and "progress" vary for each problem and are given in Ref. 7 as well as a detailed description of each problem.

Table 1 Code descriptions

Name	Description	Algor class[a]
ADRANS	Random search followed by pattern moves	DS
CLIMB	Rosenbrock search	DS
DAVID	Davidon-Fletcher-Powell with numerical derivatives	GF
DFMCG	Fletcher-Reeves conjugate gradient method with secant approximation derivatives	G
DFMFP	Davidon-Fletcher-Powell with secant approximation derivatives	G
FMIND	Hook & Jeeves pattern search	DS
GRAD4	Steepest descent method	G
GRID1	Grid and star network search, with shrinkage	AR
MEMGRD	Davidon-Fletcher-Powell with retained step length information	GF
NMSERS	Simplex search	DS
PATSH	Modified pattern search	DS
PATRN0	Modified pattern search, dome strategy	DS
PATRN1	Modified pattern search, ridge strategy	DS
RANDOM	Random search with shrinkage	AR
SEEK1	Pattern search followed by random search	DS
SEEK3	Modified pattern search	DSF
SIMPLX	Modified simplex search	DSF
DSDA	Modified pattern search followed by Mugele's search	DS

[a] DS = direct search, DSF = direct search employing SUMT strategy and penalty function, G = gradient procedure, GF = gradient procedure using SUMT strategy and penalty function, AR = area reduction method.

Received September 19, 1974. Grateful acknowledgement is given to Y. Moradi, a graduate student, for his efforts in helping prepare and run the problems of this study.

Index categories: Computer Technology and Computer Simulation Techniques; Structural Design, Optimal.

*Associate Professor, Mechanical Engineering Department. Member AIAA.

Table 2 Performance of optimization codes[a]

Code Name	Problem Number 1	2	3	4	5	6	7	8	9	10
ADRANS	2.64	0.458	1.65	P	0.100	0.654	0.159	0.069	P	P
CLIMB				0.015	0.005			0.007		
DAVID		0.188	0.84	1.132	0.075	0.046				
DFMCG			P	0.015	P			0.004		
DFMFP			P	0.038	0.250		P	0.387		
FMIND		0.004	P		0.140		0.003	0.003	P	3.74
GRAD4		P	P	0.283	0.393		P	0.004		P
GRID1	P	P		0.033	P	P		0.037	P	P
MEMGRD		0.143	0.91	0.059	0.066	0.067	P			
NMSERS		0.019	0.045	0.060	0.002	0.012	P	0.007	0.039	P
PATSH	1.78	0.220	1.00	0.013	0.018	0.020	0.010	0.009	P	3.54
PATRN0	P	P	P	0.021	P			0.002	P	1.95
PATRN1	P	P	P	0.008	0.002	P		0.001	1.02	1.20
RANDOM		P		P	P	0.024	1.20	0.013		P
SEEK1	P	P	P		0.010		0.010	0.007	P	1.53
SEEK3		0.102	0.14	0.212	0.035	0.191	0.141	0.013	4.20	P
SIMPLX	2.97	0.297	1.31	0.196	0.035	0.191	0.262	0.050	P	
DSDA	P	0.021	0.047	0.104	0.003	0.008	0.145	0.004	1.94	1.35

[a] Numerical entry indicates normalized time required for solution, and P indicates progress toward a solution.[5,7]

Table 3 Relative ranking of optimization codes

Generality				Efficiency				Gen. & Eff.	
n_s		N		f_a		$\bar{f}_a$		T_a	
9	DSDA	9.5	DSDA	1.3	PATRN1	0.1	NMSERS	9	DSDA
	PATSH		PATSH	3.3	NMSERS	0.2	PATRN1	15	PATSH
8	SEEK3	8.5	SEEK3	4.1	DSDA	0.3	DSDA	16	NMSERS
	SIMPLX		SIMPLX	12	PATSH	0.6	PATSH		PATRN1
7	ADRANS		ADRANS	16	FMIND	0.8	MEMGRD	18	SEEK3
	NMSERS	8.0	NMSERS	20	MEMGRD	0.9	FMIND	21	SIMPLX
5	PATRN1	7.0	PATRN1		SEEK3	1.0	SEEK3	24	ADRANS
	FMIND	6.0	FMIND	38	SIMPLX	1.5	SIMPLX	25	FMIND
	MEMGRD		SEEK1	50	DAVID	1.7	DAVID	26	MEMGRD
	DAVID	5.5	MEMGRD	54	ADRANS	2.4	ADRANS	27	DAVID
4	SEEK1		PATRN0						
3	CLIMB	5.0	DAVID						
	DFMFP		GRAD4						
	GRAD4		GRID1						
	PATRN0		RANDOM						
	RANDOM	4.0	DFMFP						
2	DFMCG	3.0	CLIMB						
	GRID1		DFMCG						

The data from Table 2 may be applied to a number of rating schemes for comparing the codes tested. Table 3 presents relative rankings of the codes using some of the rating criterion of Ref. 5. The rating equations used are as follows: The number of problems solved by a particular code is n_s and

$$N = n_s + n_p/2 \tag{3}$$

where n_p is the number of problems in which a P rating was achieved. The efficiency ratings are given by

$$f_a = \sum_{p=1}^{10} \frac{b_{ap}[t_{ap}/\text{min}_a(t_{ap})]}{n_s} \tag{4}$$

where $b_{ap} = 1$ if code a solved problem p and zero otherwise, t_{ap} is the normalized time required for solution (cpu time divided by cpu time required to run a standard timing program),[5] and $\text{min}_a(t_{ap})$ is the shortest time required by any of the 17 codes studied in Ref. 5 to solve problem p. The other efficiency criterion is

$$\bar{f}_a = \sum_{p=1}^{10} \frac{b_{ap}[t_{ap}/\text{mean}_a(t_{ap})]}{n_s} \tag{5}$$

where $\text{mean}_a(t_{ap})$ is the average time required by the codes studied in Ref. 5 to solve problem p. Here the overall rating number for generality (reliability) and efficiency (speed) is given by

$$T_a = \sum_{p=1}^{10} t_{ap} \tag{6}$$

where t_{ap} is set equal to twice the time used by the slowest code solving a problem p that code "a" could not solve. This penalty time is used to penalize code unreliability. Only codes that solved half or more of the problems are rated for efficiency.

The tables presented are essentially similar to those given in Ref. 5, except that the DSDA code is included in those given in this paper. Thus, the rating scheme and presentation of results used here are essentially those of Ref. 5. The DSDA solved all problems except Problem 1. On this problem an objective function value within $\frac{1}{2}\%$ of the optimum was attained but since convergence approaching the accuracy specified for this problem (0.003%) was not achieved it was not considered to have solved the problem. In most engineering situations, however, the level of convergence achieved would be considered quite adequate.

Based on its performance on the ten problems of Ref. 5, the DSDA appears to be a relatively fast and reliable optimization code. It appears comparable in reliability to the most general code investigated in Ref. 5 (PATSH) and yet comparable in speed to the fast codes (NMSERS, PATRN1). The DSDA was about as fast, or faster than, any code solving Problems 3, 5, and 6 and was significantly faster than average in all others except Problem 9 where it was about average in speed. Compared to the relatively reliable codes ($n_s \leqq 8$) the DSDA was faster than PATSH on six of eight problems that both codes solved, faster than SEEK3 in six of seven problems and faster than SIMPLX in all problems solved by both schemes. Thus on the basis of this comparison the DSDA appears to be a superior nonlinear MP code.

It may be seen that the gradient based procedures, including those employing the SUMT strategy and penalty function,[8] so widely used in engineering problems, performed rather poorly. The best gradient based code DAVID solved only half the problems. The direct search procedures in general provided much better performance although only two, the DSDA and PATSH, solved, or approached the solution to, all ten problems.

It should be noted, however, that this comparison employed rather small test problems (two to five variables with zero to ten behavior constraints). It is not clear that the direct search procedures would also possess reliability and speed superior to the gradient methods on large problems. Still the superiority in this study of the better direct search procedures is dramatic while the performance of the gradient schemes is rather dismal. The unreliability of the popular gradient schemes is particularly disturbing since these schemes performed best on those problems without any active behavior constraints (problems 4, 5, and 8). No gradient scheme solved half or more of the problems where a constraint is active at the optimum. Furthermore, although the better direct search procedures may not be as reliable on larger problems one would certainly not expect the reliability of the gradient procedures to improve.

Conclusion

Although it is recognized that such favorable findings on the DSDA code presented by its developer can be viewed with some scepticism, it should be noted that the performance cited is based on rating schemes and problems selected by Eason and Fenton and not by the author, and that this performance was achieved with a conscious effort to avoid any tuning of the code to individual problems. Thus these results indicate that the DSDA appears to be a superior nonlinear mathematical programing procedure at least on relatively small problems. It must be stated, however, that the comparison study cannot be considered sufficiently thorough to make specific claims on the relative superiority of the better direct search procedures.

References

[1] Pappas, M. and Amba-Rao, C. L., "A Direct Search Algorithm for Automated Optimum Structural Design," *AIAA Journal*, Vol. 9, March 1971, pp. 387–393.

[2] Pappas, M. and Amba-Rao, C. L., "Structural Synthesis of Thin Cylindrical Shells with Spiral-Type Stiffeners," *AIAA Journal*, Vol. 8, Aug. 1970, pp. 1529–1530.

[3] Pappas, M. and Allentuch, A., "Structural Synthesis of Frame Reinforced, Submersible, Circular Cylindrical Hulls," *Computers and Structures*, Vol. 4, Jan. 1974, pp. 253–280.

[4] Pappas, M. and Allentuch, A., "Mathematical Programming Procedures for Mixed Discrete-Continuous Design Problems," *Transactions of the Asme, Ser. B: Journal of Engineering for Industry*, Vol. 96, Feb. 1974, pp. 201–209.

[5] Eason, E. D. and Fenton, R. G., "A Comparison of Numerical Optimization Methods for Engineering Design," *Transactions of the ASME, Ser. B: Journal of Engineering for Industry*, Vol. 96, Feb. 1974, pp. 196–200.

[6] Pappas, M., "Use of Direct Search in Automated Optimal Design," *Transactions of the ASME, Ser. B: Journal of Engineering for Industry*, Vol. 94, May 1972, pp. 395–401.

[7] Eason, E. D. and Fenton, R. G., "Testing and Evaluation of Numerical Methods for Design Optimization," Rept. UTME-TP 7204, Sept. 1972, Technical Publication Series, Dept. of Mech. Eng., Univ. of Toronto, Toronto, Canada.

[8] Fracco, A. V. and McCormick, G. P., *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, Wiley, New York, 1968.

Boundary Layers on a Rotating Disk

TUNCER CEBECI*
California State University, Long Beach, Calif.
AND
DOUGLAS E. ABBOTT†
Purdue University, Lafayette, Ind.

Nomenclature

A	= damping-length constant
c_{f_θ}	= circumferential local skin-friction coefficient
f	= radial stream function
g	= circumferential stream function
r	= radial coordinate (radius from axis of rotation)
R_r	= rotational Reynolds number, $\omega r^2/\nu$
R_θ	= Reynolds number based on momentum thickness, $\theta\omega r/\nu$
u, v, w	= mean radial, axial, and circumferential velocity components, respectively
y	= axial coordinate = distance perpendicular to the disk
γ_{tr}	= intermittency factor
ε	= eddy viscosity
ε^+	= dimensionless eddy viscosity, ε/ν
η	= similarity parameter
θ	= momentum thickness, $\theta = \int_0^\infty w/\omega r(1 - w/\omega r)\,dy$
μ	= dynamic viscosity
ν	= kinematic viscosity = μ/ρ
ρ	= density
τ	= shear stress
ω	= angular velocity of disk

Subscripts

i	= inner region
o	= outer region
w	= wall

Primes denote differentiation with respect to η

Introduction

A DISK rotating in an infinite quiescent fluid is one of the simplest types of three-dimensional boundary-layer flow. Depending on the Reynolds number based on radius and angular velocity, we may have laminar, transitional, and turbulent boundary layers. According to experiments,[1] the flow is laminar if the Reynolds number is less than approximately 1.85×10^5. The flow is transitional for Reynolds number between 1.85×10^5 and 2.85×10^5. It is fully turbulent for Reynolds number greater than 2.85×10^5.

In this Note, we discuss the prediction of laminar, transitional, and turbulent boundary layers on a rotating disk by an efficient numerical method. The method employs the eddy viscosity concept to model the Reynolds shear stress terms and has been previously used to compute two-dimensional boundary layers[2] and recently three-dimensional boundary layers.[3,4] Results are

Received October 2, 1974; revision received December 12, 1974. This work was supported by the National Science Foundation Grant No. ENG74-11895.

Index categories: Boundary Layers and Convective Heat Transfer—Turbulent.

* Adjunct Professor, Mechanical Engineering Department. Member AIAA.

† Professor, Mechanical Engineering Department.