Basic Research on an Application of Geometric Programming to Sequential Unconstrained Minimization Technique

Basic Research on an Application of Geometric Programming to Sequential Unconstrained Minimization Technique

Hiroyuki Sugimoto

Abstract

Geometric programming provides a powerful tool for solving algebraic nonlinear programming subject to linear and nonlinear constraints, but it is rather difficult to apply the method to a general optimizing problem. In this paper, a penalty term in the transformed objective function in process of the calculation by SUMT is approximated with a single-term posynomial and makes it possible to apply geometric programming to a general minimization problem. This paper also explains the three numerical examples and the approaches to the optimum points are shown in the figures.

1. Introduction

Geometric programming was discovered first by Zener early in the 1960's and after that developed by Zener, Duffin and Peterson. This method provides a powerful tool for solving algebraic nonlinear programming problems subject to linear and nonlinear constraints, and in recent years the application of this method is studied in mainly Chemical and Civil Engineering fields.

Geometric programming displays its ability especially, when the objective function and the constraints are all posynomials and the number of degrees of difficulty is small. Once some of the coefficients in the polynomials are negative or, even if all the coefficients are positive, the number of degrees of difficulty are relatively great, it will be difficult to apply efficiently geometric programming to such problems. Then, to overcome this difficulty, A. B. Templeman proposed to approximate a general fuction with a single-term posynomial[1], and C. Beightler and D. T. Phillips explained in their book the technique of reducing a plynomial to a posynomial by condensation[2]. The former paper dealt with the problem of minimum weight design of truss structures. But, in the case of minimum weight design of truss bridges, the number of terms in the objective fuction is equal to that of the design variables. So, even if each constraint is approximated with a single-term posynomial, the number of degrees of difficulty may be still equal to that of constraints−1 and it is seemed to be hard to use the method to such problems. Until now, the problems were solved principally by Sequential Linear Programming (SLP) or Sequential Unconstrained Minimization Technique (SUMT). But some disadvantages of each technique were pointed out. The former technique requires much memory capacity of a computer, while the latter technique, although the possibility of converging into a global optimum point is improved, requires long computing time in optimizing a transformed objective fuction. Then this paper deals with the application of geometric programming into the optimization of the transformed objective function. In the method proposed here, the penalty term in the transformed objective function is approximated with a single-term posynomial, and by

applying geometric programming to this function it is possible to get the approximate optimum value by only one iteration, so far as the point is in the feasible region.

In this paper the constraints are to be general polynomials and the objective function to be a posynomial, in which the number of terms is equal to that of the design variables.

2. Posynomial Approximation of Transformed Objective Function

A general optimization problem is defined as follows,

minimize

$$f=\sum_{i=1}^{n} c_i \prod_{k=1}^{n} x_k^{a_{ik}}, \tag{1}$$

in which

$$c_i>0\ ;\ i=1\cdots\cdots n,$$

$$x_k>0\ ;\ k=1\cdots\cdots n, \tag{2}$$

subject to

$$g_j\geqq 0\ ;\ j=1\cdots\cdots m,$$

In equation (1) x_k (k=...n) are design variables and a_{ik} (i=1...n, k=1...n) are arbitrary real numbers.

The primary constrained minimization problem defined above is transformed into a sequence of unconstrained minimization problems. The function is as follows,

$$F=\sum_{i=1}^{n} c_i \prod_{k=1}^{n} x_k^{a_{ik}}+\gamma_l \sum_{j=1}^{m} (g_j)^{-\beta};\quad ;\ l=1\cdots\cdots L, \tag{3}$$

in which, γ_l is a response factor and β is an arbitrary positive real number.

If $x_k^{(1)}$(k=1...n) are feasible values of the variables, the second term in the equation (3) is approximated with a single-term posynomial as follows,

$$\gamma_l \sum_{j=1}^{m} (g_j)^{-\beta}=c_{n+1} \prod_{k=1}^{n} x_k^{a_{n+1\,k}}, \tag{4}$$

in which

$$c_{n+1}=\gamma_l\{\sum_{j=1}^{m} (g_j^{(1)})^{-\beta}\} \prod_{k=1}^{n} (x_k^{(1)})^{-a_{n+1\,k}}, \tag{5}$$

$$a_{n+1\,k}=-\frac{\beta x_k^{(1)}}{\sum_{j=1}^{m}(g_j^{(1)})^{-\beta}} \sum_{j=1}^{m} \left(\frac{\partial g_j}{\partial x_k}\right)^{(1)} (g_j^{(1)})^{-(\beta+1)}\ ;\ k=1\cdots\cdots n. \tag{6}$$

After all, the primary problem defined by equations (1), (2) is transformed into a problem of optimizing a posynomial with zero degrees of difficulty as follows,

$$F=\sum_{i=1}^{n+1} c_i \prod_{k=1}^{n} x_k^{a_{ik}}. \tag{7}$$

3. Minimization of An Unconstrained Posynomial

From equation (7) the minimization problem of an unconstrained posynomial is defined as follows,

minimize

$$F=\sum_{i=1}^{n+1} c_i \prod_{k=1}^{n} x_k^{a_{ik}}, \tag{7}$$

in which

$$c_i>0\ ;\ i=1\cdots\cdots n+1,$$
$$x_k>0\ ;\ k=1\cdots\cdots n,$$
$$\alpha_{ik}\ ;\ \text{arbitrary real numbers},i=1\cdots\cdots n,k=1\cdots\cdots n+1$$

The number of degrees of difficulty of equation (7) is equal to zero, so applying geometric programming to it, the design variables **x** are obtained easily as follows.

First, the normality and orthogonality conditions are

$$\sum_{i+1}^{n+1}\lambda_i=1, \tag{8}$$

$$\sum_{i=1}^{n+1}\alpha_{ik}\lambda_i=0\ ;\ k=1\cdots\cdots n, \tag{9}$$

in which λ_i (i=1...n+1) are dual variables. Matrix expression of equations (8), (9) is as follows,

$$\begin{bmatrix}1 & 1 & \cdots\cdots\cdots 1 & 1\\ \alpha_{11} & \alpha_{21} & \cdots\cdots\cdots \alpha_{n1} & \alpha_{n+11}\\ \alpha_{12} & \alpha_{22} & \cdots\cdots\cdots \alpha_{n2} & \alpha_{n+12}\\ \vdots & \vdots & \vdots & \vdots\\ \alpha_{1n} & \alpha_{2n} & \cdots\cdots\cdots \alpha_{nn} & \alpha_{n+1n}\end{bmatrix}\begin{bmatrix}\lambda_1\\ \lambda_2\\ \vdots\\ \lambda_n\\ \lambda_{n+1}\end{bmatrix}=\begin{bmatrix}1\\ 0\\ 0\\ \vdots\\ 0\end{bmatrix} \tag{10}$$

The equation (10) is divided into next two equations.

$$I\lambda+\lambda_{n+1}=1\ , \tag{11}$$
$$\boldsymbol{A}\lambda+\boldsymbol{B}\lambda_{n+1}=0, \tag{12}$$

in which

$$I=[1\ 1\cdots\cdots\cdots\cdots 1]\ , \tag{13}$$
$$\boldsymbol{B}=[\alpha_{n+11}\ \alpha_{n+12}\cdots\cdots\cdots\alpha_{n+1n}]^T\ , \tag{14}$$
$$\lambda=[\lambda_1\ \lambda_2\cdots\cdots\cdots\lambda_n]^T\ , \tag{15}$$

$$\boldsymbol{A}=\begin{bmatrix}\alpha_{11} & \alpha_{21} & \cdots\cdots\cdots \alpha_{n1}\\ \alpha_{12} & \alpha_{22} & \cdots\cdots\cdots \alpha_{n2}\\ \vdots & \vdots & \vdots\\ \alpha_{1n} & \alpha_{2n} & \cdots\cdots\cdots \alpha_{nn}\end{bmatrix}\ . \tag{16}$$

The dual variable λ_{n+1} corresponding to the penalty term is obtained by substituting equation (12) into equation (11),

$$\lambda_{n+1}=\frac{1}{1-I\boldsymbol{A}^{-1}\boldsymbol{B}} \tag{17}$$

By substituting equation (17) into equation (12) the dual variables are obtained as follows,

$$\lambda=-\frac{\boldsymbol{A}^{-1}\boldsymbol{B}}{1-I\boldsymbol{A}^{-1}\boldsymbol{B}}\ . \tag{18}$$

Knowing the dual variables λ, the design variables are

$$x_i=10^{X_i}\ ;\ i=1\cdots\cdots n, \tag{19}$$

in which

$$X=(\boldsymbol{A}^{-1})^{T}S, \tag{20}$$

$$S=\begin{bmatrix}\log\ \lambda_1-\log\ c_1+\log\ z\\ \log\ \lambda_2-\log\ c_2+\log\ z\\ \vdots\\ \log\ \lambda_n-\log\ c_n+\log\ z\end{bmatrix}, \tag{21}$$

$$z=\prod_{i=1}^{n+1}\left(\frac{C_i}{\lambda_i}\right)^{\lambda_i}. \tag{22}$$

In calculating equation (20) $\mathbf{A}^{-1}$ was obtained previously in equation (17) and is constant through the iteration.

The iteration of the method proposed here proceeds as follows :

1. Start with an initial $\mathbf{x}^{(1)}$ and set m, n, c, α and β.
2. Compute $\mathbf{A}^{-1}$.
3. Set l=0.
4. Set l=l+1, γ_l and $\mathbf{x}^{(l)}=\mathbf{x}$.
5. Compute c_{n+1}, α_{n+1k}(k=1...n) by equations (5), (6).
6. Set **B** by equation (14).
7. Compute λ by equations (17), (18).
8. If λ_i is negative, modify $\mathbf{x}^{(l)}$ and repeat from step 5.
9. Compute **x** by equations (19), (21) and (22).
10. Repeat from step 4 until the design variables are thought to be converged.

In step 8, in what direction the initial design variables $\mathbf{x}^{(1)}$ are to be modified may be a difficult problem, but it is seemed to be a better way to do it in the direction of loosening the constraints.

Special case having the objective function as follows is considered next,

$$f=\sum_{i=1}^{n}c_i x_i^{d}, \tag{23}$$

In this case $\mathbf{A}^{-1}$ is simplified as follows,

$$\boldsymbol{A}^{-1}=\frac{1}{d}\begin{bmatrix}1 & 0 & \cdots\cdots & 0\\ 0 & 1 & \cdots\cdots & 0\\ \vdots & \vdots & & \vdots\\ 0 & 0 & & 1\end{bmatrix}. \tag{24}$$

Substituting equation (24) into equations (17), (18) and (19), λ and **x** are obtained as follows,

$$\lambda_i=-\frac{\alpha_{n+1i}}{d-\sum_{j=1}^{n}\alpha_{n+1j}}\ ;i=1\cdots\cdots n, \tag{25}$$

$$\lambda_{n+1}=\frac{d}{d-\sum_{j=1}^{n}\alpha_{n+1j}}\quad , \tag{26}$$

$$x_i=\left(\frac{\lambda_i z}{c_i}\right)^{\frac{1}{d}}\quad ; i=1\cdots\cdots n. \tag{27}$$

4. Numerical Examples

Three numerical examples are solved by the method above mentioned. In these examples the constraints are identical and as follows,

$$g_1=4.5x_1-x_2^2+6x_2-13.5\geqq 0 \tag{28}$$

$$g_2=x_1+2x_2^2-6x_2+2\geqq 0 \tag{29}$$

$$g_3=x_1\geqq 0 \tag{30}$$

$$g_4=x_2-x_1\geqq 0 \tag{31}$$

In the feasible region formed by above constraints, as shown in under figures, two optimum points are found and one of them is thought to be a local optimum point.

In these examples β and γ_l are as follows,

$$\beta=1.0,$$
$$\gamma_l=\gamma_{l-1}/10\ ;\ l=2\cdots\cdots,$$
$$\gamma=1.0,$$

and in **4−1** and **4−2**,

$$x_l=0\ .$$

4−1 Example 1

The objective function is as follows,

$$f=x_1^d+cx_2^d \tag{32}$$

Refering to equations (27), (28) and (29), the problems in the case of d=1,2 and c=0.1, 0.5, 0.8 were calculated respectively. The results are in Tables 1−1~2−3 and the approaches to the optimum point are shown in Figures 1−1~2−3. The dot-dash-lines in these figures are corresponding with the objective functions.

Table 1-1 d=1.0, c=0.1

initial value	x_1	x_2	f
(4.0,4.0)	1.0160	2.7433	1.2903
(5.0,0.5)	1.0149	2.7513	1.2900
(2.0,3.0)	1.0208	2.7386	1.2947
(3.0,1.0)	1.0183	2.7359	1.2919

Table 2-1 d=2.0, c=0.1

initial value	x_1	x_2	f
(4.0,4.0)	1.1056	2.3476	1.7735
(5.0,0.5)	1.0700	2.4667	1.7533
(2.0,3.0)	1.0739	2.4364	1.7468
(3.0,1,0)	1.0966	2.3628	1.7608

Table 1-2 d=1.0, c=0.5

initial value	x_1	x_2	f
(4.0,4.0)	1.1050	2.3593	2.2846
(5.0,0.5)	1.1055	2.3605	2.2858
(2.0,3.0)	1.0996	2.3488	2.2740
(3.0,1.0)	1.9276	0.9593	2.4072

Table 2-2 d=2.0, c=0.5

initial value	x_1	x_2	f
(4.0,4.0)	1.1039	2.3380	3.9518
(5.0,0.5)	1.1106	2.3498	3.9944
(2.0,3.0)	1.1149	2.3551	4.0162
(3.0,1.0)	1.0986	2.3415	3.9483

Table 1-3 d=1.0, c=0.8

initial value	x_1	x_2	f
(4.0,4.0)	1.1056	2.3455	2.9820
(5.0,0.5)	1.1208	2.3780	3.0232
(2.0,3.0)	1.1126	2.3477	2.9908
(3.0,1.0)	2.3730	0.5257	2.7936

Table 2-3 d=2.0, c=0.8

initial value	x_1	x_2	f
(4.0,4.0)	1.1057	2.3405	5.6050
(5.0,0.5)	2.0160	0.8818	4.6862
(2.0,3.0)	1.0979	2.3377	5.5771
(3.0,1.0)	1.9283	0.9584	4.4533

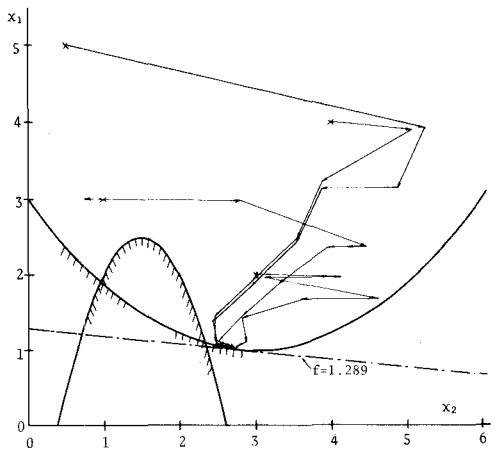

Fig. 1 − 1 d =1.0, c =0.1

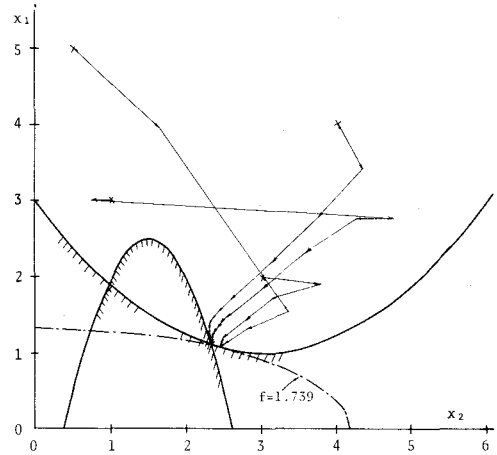

Fig. 2 − 1 d =2.0, c =0.1

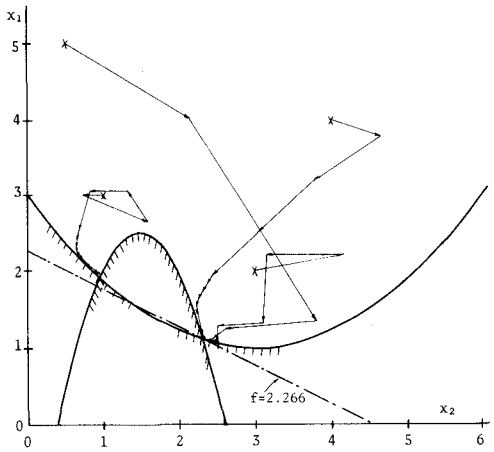

Fig. 1 − 2 d =1.0, c =0.5

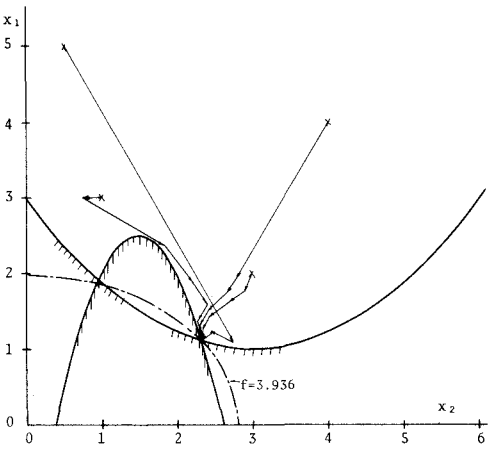

Fig. 2 − 2 d =2.0, c =0.5

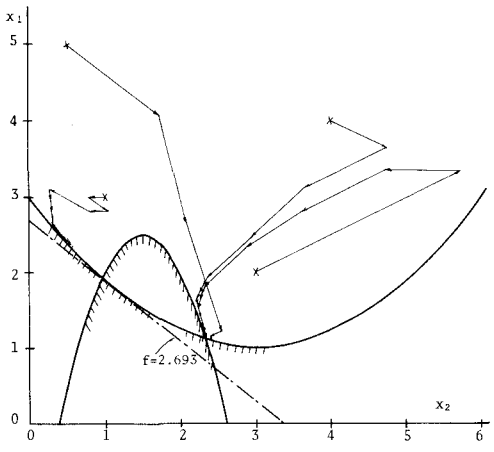

Fig. 1 − 3 d =1.0, c =0.8

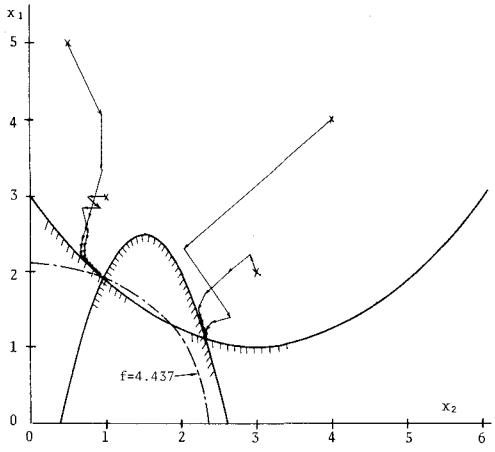

Fig. 2 − 3 d =2.0, c =0.8

4−2 Example 2

The objective function is as follows,

$$f=x_1/x_2+cx_2^2 \tag{33}$$

The problems in the case of c=0.1, 1.0 were calculated respectively. The approaches to the optimum point are shown in Figure 3−1 and 3−2.

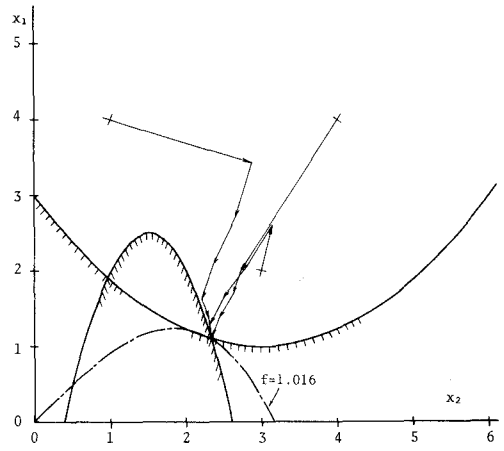

Fig. 3 − 1 c =0.1

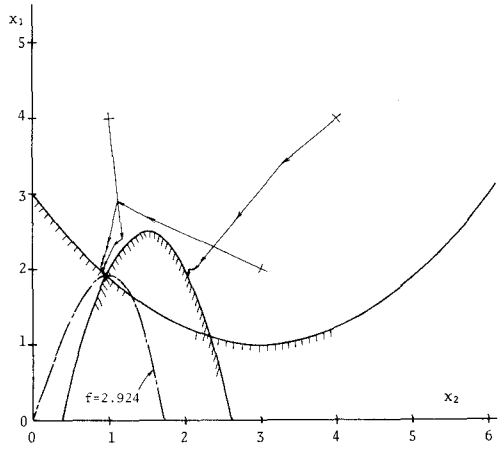

Fig. 3 − 2 c =1.0

4−3 Example 3

The objective function is as follows,

$$f=x_1x_2+0.1x_2^2 \tag{34}$$

The function defined above, as shown in Figure 4−1, is not related to the value of the design variable x_1 and approaches to zero, if the design variable x_2 do so. Applying the method proposed here to such a function, it happens to be frequently that the values of λ are negative and that it is difficult to find the direction in which the design variables are to be modified. So, in this paper, the objective function is approximated with a linear function as follows successfully,

$$f=(x_2^{(1)})x_1+(x_1^{(1)}+0.2x_2^{(1)})x_2. \tag{35}$$

The approaches to the optimum point in the case of x_t=0.5, 1.0 and 1.5 respectively are shown in Figures 4−1~4−3.

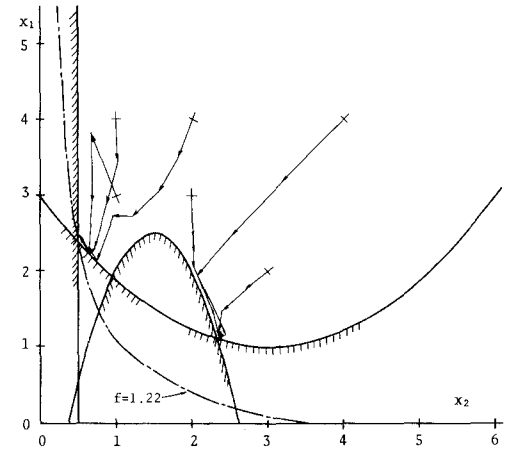

Fig. 4 − 1 x_t=0.5

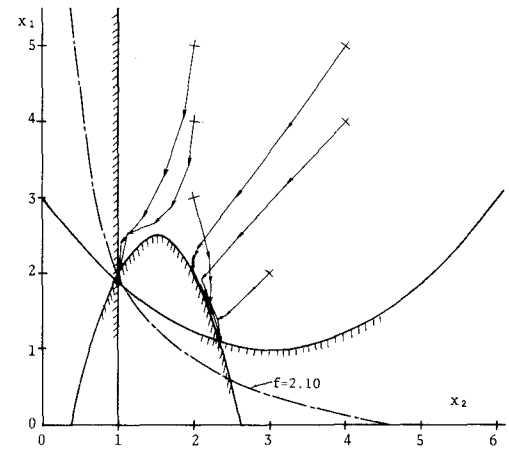

Fig. 4 − 2 x_t=1.0

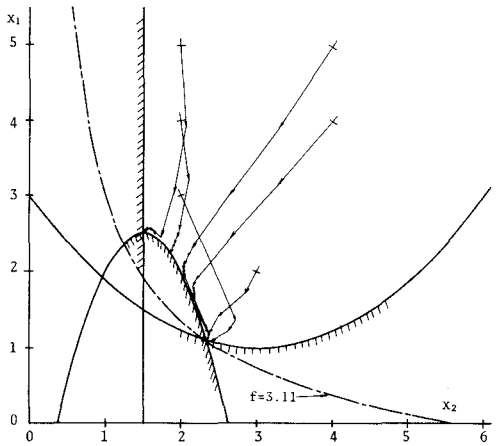

Fig. 4 − 3 $x_t=1.5$

5. Conclusions And Comments

1. Approximating a penalty term in the transformed objective function with a single-term posynomial, it is possible to apply geometric programming to the general minimization problems efficiently.
2. If the initial values are selected properly, the convergence is good as shown in the figures above. And it is noticeable that, in spite of being a global optimum point in a very narrow region in the case of $x_t=1.0$ in Example 3, the design variables approaches to the point very smoothly, when the initial design variables are (5, 2) and (4, 2).
3. Hereafter, by solving more concrete problems, it is intended to make a comparison of the computing time and accuracy of the method proposed here, SLP and SUMT by direct search method or DFP.

(Received May. 19, 1978)

References

1) A. B. Templeman, "Optimum Truss Design Using Approximating Functions", Optimization In Structural Design, Springer-Verlag, 1975.
2) Charles S. Beightler, Don T. Phillips, "Applied Geometric Programming", John Wiley & Sons, 1976.