Transformed Newton's Method

By

Ronald Kelly Tulk

A Thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Mineral Economics.

Signed: Ronald K. Falk
Student

Golden, Colorado

Date: Oct 11, 1976

Approved: [signature]
Thesis Advisor

J. Paul Matthews
Head of Department

Golden, Colorado

Date: Oct. 12, 1976

Abstract

A modification of Newton's classical solution technique for nonlinear problems is developed that utilizes transformed functions and variables. Several examples of its application to primal programming problems are presented as well as an application of the technique for the solution of the dual geometric programming problem.

Table of Contents

List of Tables and Figures

Introduction

The very common problem of finding a solution to a nonlinear equation, $f(x)=0$, has great practical importance. It occurs in such diverse fields as electrical engineering, for design analysis, and economics, for rate of return calculations in discounted cash flow problems.

One of the oldest and most venerable solution techniques is that of Newton (1) which involves the linear approximation of the function at some point. The linear approximation is then utilized to calculate a new approximate solution with the entire process being repeatedly applied until the solution is of the desired accuracy.

In general, for N simultaneous equations in N variables $(f_i(x_1,\ldots,x_N)=0;\ i=1,\ldots,N)$, the equations of Newton's method, for which the incremental variable adjustments, Δx_j, must be iteratively solved, are

$$\sum_{j=1}^{N}\left(\frac{\partial f_i(x_1,\ldots x_N)}{\partial x_j}\right)\Delta x_j = -f_i(x_1,\ldots,x_N),\quad i=1,\ldots,N; \qquad [1]$$

where both the functions and the partial derivatives are evaluated at the current approximate solution. The new approximate solution is $x_j = x_j + \Delta x_j$ $(j=1,\ldots,N)$. For the specific situation of one equation in one variable $(f(x)=0)$, the equations of [1] simplify to the iterative relation

$$x_{k+1} = x_k - \frac{f(x_k)}{\left(\frac{\partial f(x)}{\partial x}\right)\Big|_{x=x_k}} \quad . \qquad [2]$$

Newton's method is very general in nature and will guarantee convergence to a solution if the initial approximate solution is sufficiently close to the actual solution. The only restrictions are that the functions and their first and second partial derivatives (assuming that they exist) are continuous and bounded in the domain of interest.

If the restriction of having each term of the functions being limited to an algebraic expression is included, a nonlinear approximation method can be developed that is very similar to that of Newton, but which possesses certain desirable properties that are not present when Newton's method is applied directly to the functions. The purpose of this dissertation is to develop a nonlinear approximation technique that is an alternative to a direct application of Newton's method. Finally, to illustrate its application and for comparison purposes, several examples will be provided.

At this time, I would like to acknowledge and thank the Colorado School of Mines and the United States Department of Health, Education and Welfare for the financial assistance afforded me the past two years.

Development

Many nonlinear problems can be formulated so that each term has the specific algebraic form $t_i = c_i \prod_{j=1}^{N} x_j^{\alpha_{ij}}$. Each term, t_i, includes a real coefficient, c_i, positive real variables, x_j, and real exponents, α_{ij}. This term structure is the same term structure as is employed in geometric programming ((2), (3), (4), and (5)).

These terms are combined either by addition or subtraction to form equations, inequality constraints, or objective functions to accommodate the problem at hand.

Since the variables are restricted to non-negative values, each term will always be either non-negative or non-positive, depending on whether its coefficient is positive or negative (respectively). An expression of several terms is called a posynomial if the coefficient of each term is positive. If there are both positive and negative coefficients, the expression is a signomial. It should be noted that each signomial expression can be represented as the difference of two posynomials (6). This is an important observation because the property of the posynomial that all terms are non-negative is the foundation of the technique to be developed. The fact that each signomial can be decomposed into two posynomials allows the complete application of the technique to the general signomial form.

In the mathematical notation that follows, certain non-universal notation will be employed which will now be defined. If more than one variable is present, $\bar{x}$ will be a vector which represents the variables, while $\bar{x}_k$ will represent the vector of variables, but for which a particular value has been assigned to each variable--the k^{th} solution point. x_j will be the j^{th} variable of $\bar{x}$, and x_{jk} is the value of x_j at the k^{th} solution point, $\bar{x}_k$. The natural logarithm will be denoted by $\ln$. Also, $\frac{\partial g(\bar{x}_k)}{\partial x_j}$ will signify the partial derivative of $g(\bar{x})$ with respect to x_j and then evaluated at $\bar{x}_k$, or,

$$\frac{\partial g(\bar{x}_k)}{\partial x_j} \equiv \left.\frac{\partial g(\bar{x})}{\partial x_j}\right|_{\bar{x}=\bar{x}_k}.$$

The geometric inequality which provides the basis for the dual program of geometric programming is also the basis for the technique to be developed. But instead of dealing with the dual optimization problem, this technique attempts to solve the primal problem, and in addition, can be used to solve nonlinear equations when optimization is not involved.

Generally, the geometric inequality (7) states that the arithmetic average of positive numbers is never less than the geometric mean, or stated mathematically, $\sum_{i=1}^{M} \delta_i u_i \geq \prod_{i=1}^{M} (u_i^{\delta_i})$, where $\sum_{i=1}^{M} \delta_i = 1$ and the numbers, u_i, and the weights, δ_i, are positive. A more useful form of the inequality is obtained with the substitution of variable $U_i = \delta_i u_i$. The inequality becomes $\sum_{i=1}^{M} U_i \geq \prod_{i=1}^{M}\left(\frac{U_i}{\delta_i}\right)^{\delta_i}$, with $\sum_{i=1}^{M}\delta_i$ still equal to unity.

This form of the geometric inequality will be used to generate a monomial, or one term, approximation of a general posynomial, $g(\bar{x}) = \sum_{i=1}^{M} c_i \prod_{j=1}^{N} x_j^{\alpha_{ij}}$. The approximation is achieved by first substituting the posynomial terms in place of the U_i terms on both sides of the inequality. The inequality becomes $\sum_{i=1}^{M} c_i \prod_{j=1}^{N} x_j^{\alpha_{ij}} \geq \prod_{i=1}^{M}\left(\frac{c_i \prod_{j=1}^{N} x_j^{\alpha_{ij}}}{\delta_i}\right)^{\delta_i}$, where the left side of the inequality is the posynomial and the right side is the approximation monomial. The approximation is complete once the proper values of the δ_i weights have been chosen. The weights are chosen to be $\delta_i = \frac{c_i \prod_{j=1}^{N} x_{jK}^{\alpha_{ij}}}{\sum_{\ell=1}^{M} c_\ell \prod_{j=1}^{N} x_{jK}^{\alpha_{ij}}}$. The sum of the weights is thus guaranteed to equal one. Substituting these functional weights into the monomial approximation, $g(\bar{x}, \bar{x}_K)$, the approximation becomes

$$g(\bar{x},\bar{x}_K) = \prod_{i=1}^{M}\left(\frac{c_i \prod_{j=1}^{N} x_j^{\alpha_{ij}}}{\left(\frac{c_i \prod_{j=1}^{N} x_{jK}^{\alpha_{ij}}}{\sum_{\ell=1}^{M} c_\ell \prod_{j=1}^{N} x_{jK}^{\alpha_{\ell j}}}\right)}\right)^{\left(\frac{\alpha_{ij} c_i \prod_{j=1}^{N} x_{jK}^{\alpha_{ij}}}{\sum_{\ell=1}^{M} c_\ell \prod_{j=1}^{N} x_{jK}^{\alpha_{\ell j}}}\right)} ;$$

which, after simplification, reduces to:

$$g(\bar{x},\bar{x}_K) = g(\bar{x}_K) \prod_{j=1}^{N}\left(\frac{x_j}{x_{jK}}\right)^{\left(\frac{\sum_{i=1}^{M} \alpha_{ij} c_i \prod_{j=1}^{N} x_{jK}^{\alpha_{ij}}}{\sum_{i=1}^{M} c_i \prod_{j=1}^{N} x_{jK}^{\alpha_{ij}}}\right)}$$

Further simplification is achieved by noting that the exponent of the monomial can be replaced by an equivalent expression:

$$\frac{\sum_{i=1}^{M} a_{ij} c_i \prod_{j=1}^{N} x_{jK}^{a_{ij}}}{\sum_{i=1}^{M} c_i \prod_{j=1}^{N} x_{jK}^{a_{ij}}} = \frac{x_j}{g(\bar{x}_K)}\left(\frac{\partial g(x_K)}{\partial x_j}\right) = \frac{\partial \ln g(\bar{x}_K)}{\partial \ln x_j}$$

The one term posynomial approximation has now become:

$$g(\bar{x},\bar{x}_K) = g(\bar{x}_K) \prod_{j=1}^{N} \left(\frac{x_j}{x_{jK}}\right)^{\left(\frac{\partial \ln g(\bar{x}_K)}{\partial \ln x_j}\right)} \qquad [3]$$

where $\bar{x}_K$ is the point around which the posynomial is being approximated. This is the same result as that obtained by Duffin, Peterson and Zener (8).

Evaluating the approximation monomial at the approximation point results in:

$$g(\bar{x},\bar{x}_K)\Big|_{\bar{x}=\bar{x}_K} = g(\bar{x})\Big|_{\bar{x}=\bar{x}_K} \qquad [4]$$

In other words, the monomial approximation, $g(\bar{x},\bar{x}_K)$, and the posynomial, $g(\bar{x})$, are coincident at the approximation point. Taking the partial derivative of the monomial with respect to the ℓ^{th} variable, x_ℓ, gives:

$$\frac{\partial g(\bar{x},\bar{x}_K)}{\partial x_\ell} = \left(\frac{\partial g(\bar{x}_K)}{\partial x_\ell}\right)\left(\frac{x_{\ell K}}{x_\ell}\right)\prod_{j=1}^{N}\left(\frac{x_{jK}}{x_j}\right)^{-\left(\frac{\partial \ln g(\bar{x}_K)}{\partial \ln x_j}\right)} .$$

Evaluating this partial derivative at the approximation point reveals $\frac{\partial g(\bar{x},\bar{x}_K)}{\partial x_\ell}\Big|_{\bar{x}=\bar{x}_K} = \frac{\partial g(\bar{x})}{\partial x_\ell}\Big|_{\bar{x}=\bar{x}_K}$. More generally, the gradient of the approximation monomial equals the gradient of the posynomial when these are evaluated at the approximation point, or,

$$\nabla g(\bar{x},\bar{x}_\kappa)\Big|_{\bar{x}=\bar{x}_\kappa} = \nabla g(\bar{x})\Big|_{\bar{x}=\bar{x}_\kappa} \quad . \qquad [5]$$

Because of equations [4] and [5], the approximation monomial and the posynomial are tangent at the approximation point. The approximation is nonlinear and generally tends to be a better approximation to the posynomial than would be a linear approximation as is used in Newton's method. The non-linear approximation of a signomial is further enhanced because, as will be developed later, each posynomial of a signomial will be individually approximated by a separate monomial.

Duffin, Peterson and Zener utilized this monomial approximation (equation [3]) to convert positive signomial expressions to proximate posynomials because their dual techniques were only applicable to posynomial problems. They did not pursue primal applications of this technique as will be investigated here.

If the posynomial is limited to one variable, one of the more advantageous aspects of this approximation method can be demonstrated. The one variable approximation is

$$g(x,x_\kappa) = g(x_\kappa)\left(\frac{x}{x_\kappa}\right)^{\left(\frac{\partial \ln g(x_\kappa)}{\partial \ln x}\right)} \quad . \qquad [3a]$$

The exponent, $\frac{\partial \ln g(x)}{\partial \ln x}$, is interesting in that it represents the relative rate of change of the posynomial with respect to the relative rate of change of the variable, and as such, is termed the elasticity of the posynomial with respect to the variable (from the economic usage of the term elasticity, (9)).

Two properties of posynomial elasticities are that:

1) they are bounded above by the largest exponent of the variable, and
2) they approach their upper bound relatively rapidly.

These properties, together, give the monomial approximation (equation [3a]) an exponent, $\frac{\partial \ln g(x)}{\partial \ln x}$, that is fairly insensitive relative to the magnitude of the variable.

This can be readily demonstrated by the following example posynomial: $g(x) = x^5+8x^4+x^3+2x^2+x+12$. Tabulated in Table 1 are posynomial elasticities for this posynomial for various values of x which illustrate the relative stability of the quantity.

Table 1.

x	$\frac{\partial \ln g(x)}{\partial \ln x}$
1000	4.992
100	4.925
80	4.908
60	4.882
40	4.832
20	4.710
10	4.544
5	4.431
4	4.263
3	4.129
2	3.758
1	1.800

These data are plotted in Figure 1 to further illustrate the stability of posynomial elasticities.

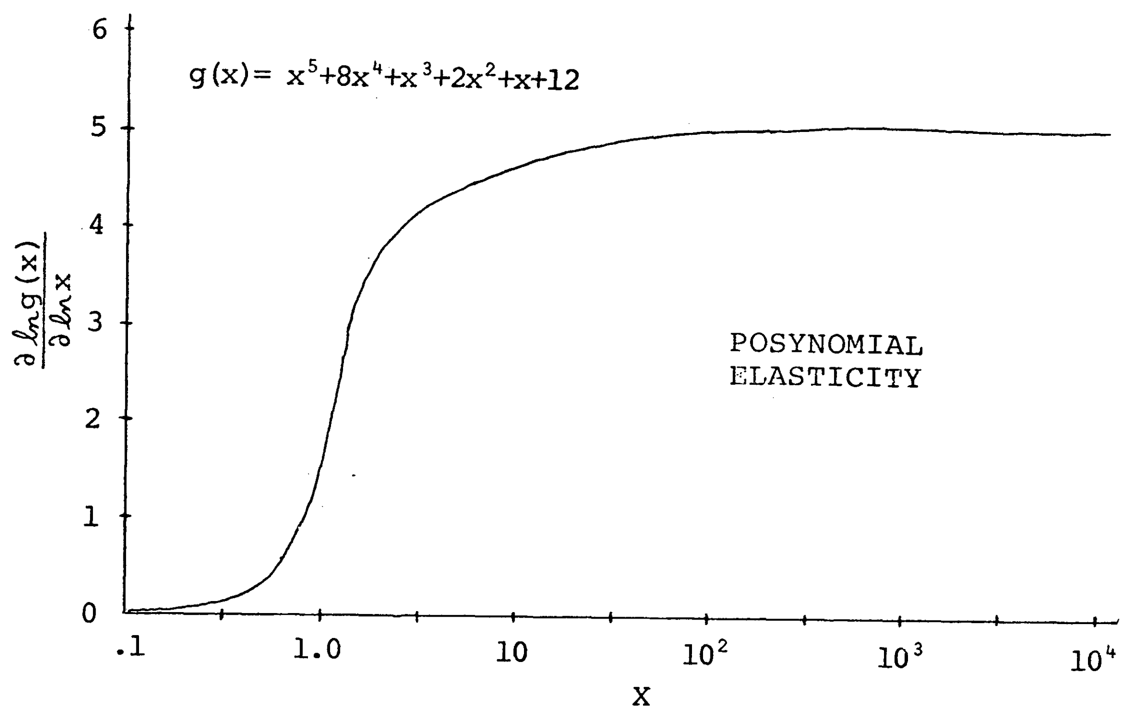


Figure 1.

The monomial approximation exponent can also be viewed as a weighted average of the variable exponents of the individual terms, weighted by the term's relative value in the posynomial. Thus, as x becomes larger, the larger exponents will be increasingly more heavily weighted.

So far, only individual posynomials have been approximated without any mention being made of practical applications of this approximation technique.

As was noted previously, every signomial equation can be decomposed into the difference of two posynomials, g(x) and h(x). The procedure is to equate the posynomials and then approximate each posynomial with a monomial about the same approximation point. In the case of only one variable, the variable can be solved for directly to obtain a new approximate solution point. If more than one variable is involved, the natural logarithm of each monomial can be taken to yield an equation that is linear in the natural logarithms of the variables. The iterative formula for the root of a signomial (one variable in one equation) will be developed first with subsequent extension to the iterative solution of N variables in N nonlinear equations.

Let f(x) be a signomial in one variable so that

$$f(x) = \sum_{i=1}^{M} c_i x^{\alpha_i} .$$

This signomial can be decomposed into two posynomials g(x) and h(x), or, f(x) = g(x) - h(x). Equating the posynomials results in: g(x) = h(x). Approximating these posynomials with monomials about some approximation point, x_k, gives:

$$g(x,x_k) = g(x_k)\left(\frac{x}{x_k}\right)^{\left(\frac{\partial \ln (g(x_k))}{\partial \ln x}\right)} = h(x_k)\left(\frac{x}{x_k}\right)^{\left(\frac{\partial \ln (h(x_k))}{\partial \ln x}\right)} = h(x,x_k) .$$

Solving for the variable, x, the following equation is produced:

$$x_{k+1} = x_k \left(\frac{h(x_k)}{g(x_k)}\right)^{\left(\frac{\partial \ln \frac{g(x_k)}{h(x_k)}}{\partial \ln x}\right)^{-1}} . \qquad [6]$$

The character and nature of this iterative solution technique can be made clearer if the natural logarithm of equation [6] is taken. The result is

$$\ln x_{K+1} = \ln x_K - \left(\frac{\ln \frac{g(x_K)}{h(x_K)}}{\frac{\partial \ln \frac{g(x_K)}{h(x_K)}}{\partial \ln x}} \right) , \qquad [7]$$

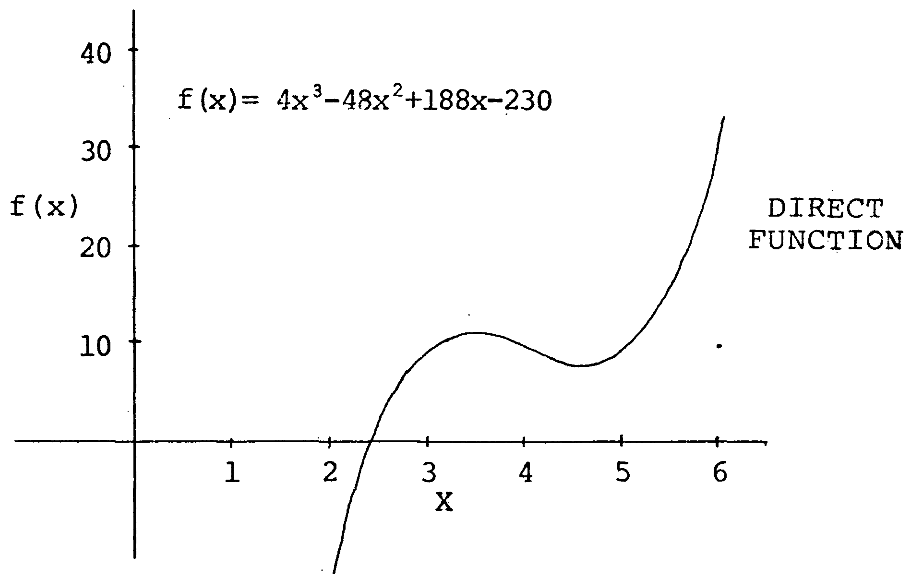
which is clearly Newton's method (equation [2]) applied to a transformed function and variable, $\ln \frac{g(x)}{h(x)}$ and $\ln x$, respectively. As a result, everything that can be said of Newton's method can be said of the technique indicated by equation [6], namely, that convergence to a solution can be guaranteed if the initial starting point is sufficiently close to the actual solution.

The main advantages of using Newton's method applied to a transformed function and variable are that, on the basis of this author's computational experience, the technique is much less sensitive to the value of the initial starting solution point and the convergence is often more rapid. The insensitivity to the initial solution point is the result of the fact that $\ln \frac{g(x)}{h(x)}$ is asymptotic to a linear function of $\ln x$. As x becomes large, only the terms in g(x) and h(x) with the largest exponents remain significant. If $c_g x^{\alpha_g}$ and $c_h x^{\alpha_h}$ are the largest exponential terms of g(x) and h(x), respectively, then as x becomes large, $\ln \frac{g(x)}{h(x)}$ approaches $\ln\left(\frac{c_g}{c_h}\right) + (\alpha_g - \alpha_h) \ln x$, which is a straight line with respect to $\ln x$. Thus, as x_K becomes very large, x_{K+1} will approach $\left(\frac{C_h}{C_g}\right)^{(\alpha_g - \alpha_h)^{-1}}$. Therefore, an initial solution point for the largest root of a signomial could always be given as $x_o = \left(\frac{C_h}{C_g}\right)^{(\alpha_g - \alpha_h)^{-1}}$ regardless of whether the direct function and variable or the transformed function and variable are used with Newton's method. However, when the transformed equation, $\ln \frac{g(x)}{h(x)} = 0$, and transformed variable, $\ln x$, are used, this starting point is the automatic result of the first iteration (for large initial values of x_o). 'Automatic starting', though far from infallible, is a decided advantage for multiple variable problems where a good starting point is not as obvious as for the one variable case.

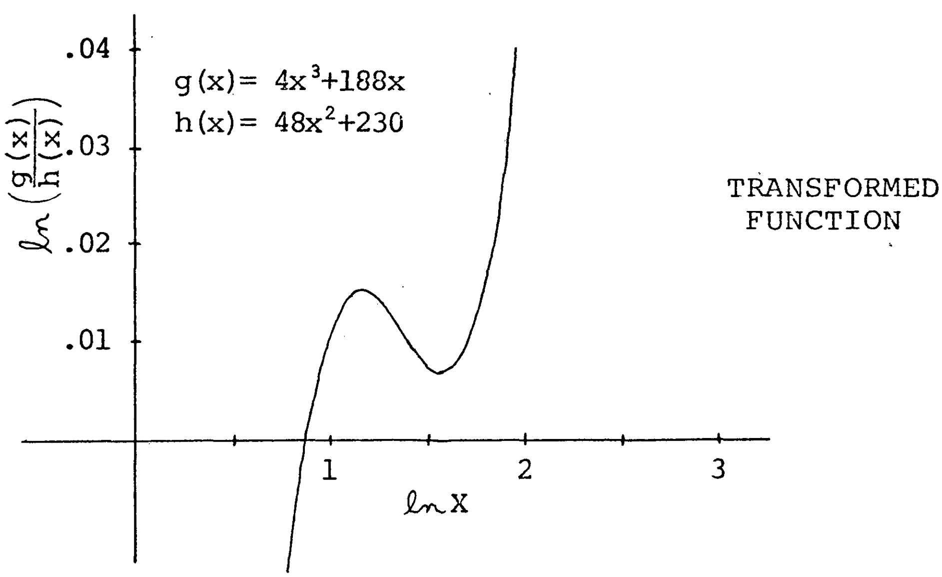
As with Newton's direct method, this technique can experience difficulty with double roots (slower convergence and diminished accuracy) and with functions which have relative minimums or maximums in the domain of x that is greater than the largest solution point. Figure 2 illustrates that a relative minimum 'kink' of a direct function may still be present with the transformed function.

This technique can easily be extended to the more general multiple variable case (following the equations of [1]). The N equations are still composed of posynomial differences which can be represented by

$$f_i(x) = g_i(x) - h_i(x) = 0 \; ; \quad i=1,\ldots,N \; .$$



40
30
20
10
f(x)
f(x)= 4x³-48x²+188x-230
1
2
3
4
5
6
X
DIRECT FUNCTION



.04
.03
.02
.01
ln(g(x)/h(x))
g(x)= 4x³+188x
h(x)= 48x²+230
1
2
3
ln X
TRANSFORMED FUNCTION

Figure 2,

The linear set of equations for which the Δ x are iteratively solved for are:

$$\sum_{j=1}^{N}\left(\frac{\partial \ln\left(\frac{g_i(x_k)}{h_i(x_k)}\right)}{\partial \ln x_j}\right) \Delta \ln x_j = -\ln\left(\frac{g_i(x)}{h_i(x)}\right); \qquad i=1,\ldots,N \qquad [8]$$

This method could be further generalized by dropping the algebraic term structure requirement, but while still maintaining the positivity constraints of:

1) g(x) and h(x) > 0 ; (i=1,...,N), and

2) x > 0 ; (j=1,...,N).

Monotonic functions, such as the exponential and logarithmic functions, could then be readily incorporated by the technique indicated by the equations of [8]. Periodic functions, such as trigonometric functions, could also be used, but with decreased effectiveness.

Optimization problems, both constrained and unconstrained, can be handled by this method. Constrained optimization problems can be converted to unconstrained optimization problems by use of the Lagrangian function. Partial derivatives of the unconstrained optimization function can be taken with respect to each variable and set equal to zero. This results in a set of nonlinear equations whose number equals the number of variables and the solution can be determined as indicated above.

General Application
Of The Foregoing Technique

Having developed the basis of the technique, the method will now be further demonstrated by several examples. Three types of problems will be presented: 1) the determination of a positive root of a signomial, 2) the solution of simultaneous signomial equations, and 3) the solution of optimization problems.

When the positive roots of a signomial are roughly known, the method presented here (which, for convenience, will be termed the transformed Newton's method) can be used (as is Newton's direct method) to determine the roots more precisely. When the roots are not known, the philosophy of application of the transformed Newton's method is to find the largest positive root first, due to the fact that the method is less sensitive to the initial approximate solution (as long as the initial approximate solution is greater than the largest real root). Once the largest real root has been determined, the signomial, if it is integer exponentiated (a polynomial), can be reduced by synthetic division and the next largest real root can then be obtained by using the root that had just been eliminated as a new approximate solution starting point. The process would continue until all positive roots had been found.

Negative roots of polynomials can be located by using the substitution of variable $x=-y$. This substitution of variable produces a polynomial whose positive roots correspond to the negative roots of the original polynomial.

The method of finding the negative roots of a polynomial by finding the positive roots of a transformed polynomial can easily be illustrated by the following example: $p(x)=x^4+2x^3+6x^2+13x-48$. Substituting $x=-y$, the polynomial becomes $q(y)=y^4-2y^3+6y^2-13y-48$, and the negative roots of the original polynomial can now be located.

The indifference of the transformed Newton's method to the initial approximate solution can be demonstrated by using $f(x)=x^5-3.7x^4+7.4x^3-10.8x^2+10.8x-6.8$ as an example, (10). For this signomial the posynomials are $g(x)=x^5+7.4x^3+10.8x$ and $h(x)=3.7x^4+10.8x^2+6.8$. The iterative root equation becomes:

$$x_{k+1} = x_k\left(\frac{3.7x_k^4+10.8x_k^2+6.8}{x_k^5+7.4x_k^3+10.8x_k}\right)^{\left(\frac{5x_k^5+22.2x_k^3+10.8x_k}{x_k^5+7.4x_k^3+10.8x_k}-\frac{14.8x_k^4+21.6x_k^2}{3.7x_k^4+10.8x_k^2+6.8}\right)^{-1}} .$$

The largest positive root is 1.70 and 10^8 will be used as an initial approximate solution. Thus, $x_0=10^8$ and

$$x_1=(10^8)\left(\frac{3.700000044\times10^{32}}{1.000000015\times10^{40}}\right)^{\frac{1}{.999999999}}$$

The first iteration has gone from $x_0 = 10^8$ to

$x_1 = 3.699999916$.

To further illustrate the point, if $x_0 = 28$,

$$x_1 = (28)\left(\frac{2282701.199}{17373115.18}\right)^{\frac{1}{4.981229665 - 3.992569504}}$$

$$= (28)(.131392740)^{1.011469906}$$

$$= (28)(.128369386)$$

$= 3.594342770$. There is very little difference in x_1 whether $x_0 = 10^8$ or $x_0 = 28$ is used to initialize the transformed Newton's method.

Plotted in Figure 3 are both the direct function, $f(x) = x^5 - 3.7x^4 + 7.4x^3 - 10.8x^2 + 10.8x - 6.8$, and the transformed function, $\ln\left(\frac{g(x)}{h(x)}\right) = \ln\left(\frac{x^5 + 7.4x^3 + 10.8x}{3.7x^4 + 10.8x^2 + 6.8}\right)$.

Tabulated in Table 2 is a comparison of Newton's direct method (Column I) with the transformed Newton's method (Column II). For Newton's direct method, an initial value of $x_0 = 4.0$ was used, while $x_0 = 28.0$ was used for the transformed Newton's method. Using Newton's direct method with $x_0 = 10^8$ or 28.0, the results of the first iteration would have been $x_1 = 2.0 \times 10^7$ or 22.542743, respectively, with very slow subsequent convergence.

Included in Appendix A is a listing of a simple FORTRAN program that determines the largest positive root of a signomial using the transformed Newton's method.

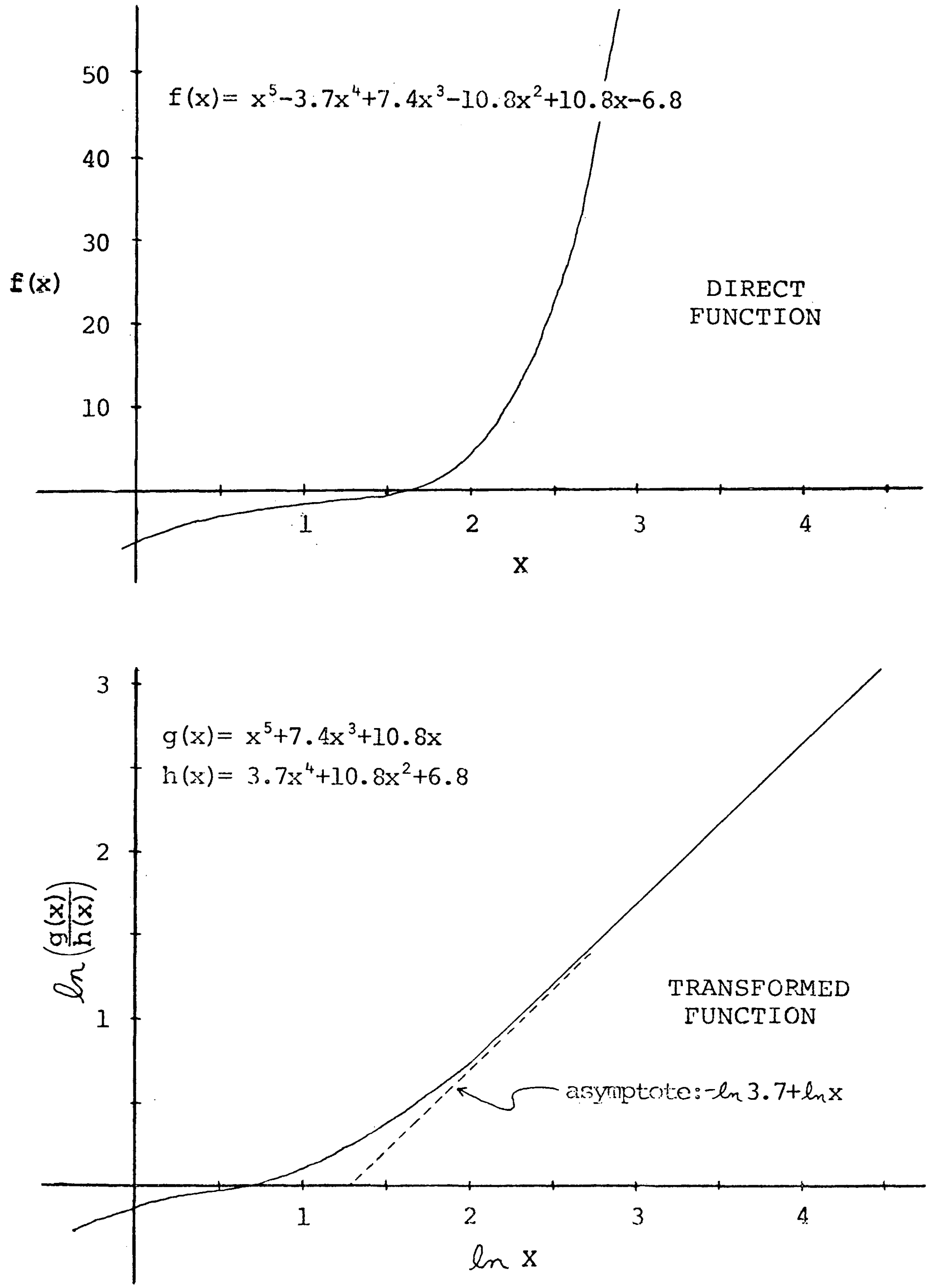

50
40
30
20
10
f(x)
f(x)= x^5-3.7x^4+7.4x^3-10.8x^2+10.8x-6.8
DIRECT
FUNCTION
1
2
3
4
X
3
2
1
g(x)= x^5+7.4x^3+10.8x
h(x)= 3.7x^4+10.8x^2+6.8
ln(g(x)/h(x))
TRANSFORMED
FUNCTION
asymptote: -ln 3.7+ln x
1
2
3
4
ln X

Figure 3.

Table 2.

Iteration	Column I (direct)	Column II (transformed)
0	4.00	28.00
1	3.32397127	3.59434277
2	2.78383604	2.62183010
3	2.35772288	1.80138809
4	2.03506139	1.70335440
5	1.81985310	1.70000330
6	1.71994643	1.70000000
7	1.70063257	
8	1.70000061	
9	1.70000003	
10	1.70000000	

The application of the transformed Newton's method to simultaneous nonlinear equations will be illustrated by the solution of the following simple problem:

$$x_1^2 + x_1 x_2^3 = 9$$

$$3x_1^2 x_2 - x_2^3 = 4 ,$$

the solution to which is $x_1 = 1.3364$ and $x_2 = 1.7542$. The related set of equations which are to be iteratively solved for $\Delta \ln x_j$ are:

$$\left.\left(\frac{2x_1^2 + x_1 x_2^3}{x_1^2 + x_1 x_2^3}\right)\right|_{\bar{x}_k} \Delta \ln x_1 + \left.\left(\frac{3x_1 x_2^3}{x_1^2 + x_1 x_2^3}\right)\right|_{\bar{x}_k} \Delta \ln x_2 = \left.\ln\left(\frac{9}{x_1^2 + x_1 x_2^3}\right)\right|_{\bar{x}_k} ,$$

$$\left.\left(\frac{6x_1^2 x_2}{3x_1^2 x_2}\right)\right|_{\bar{x}_k} \Delta \ln x_1 + \left.\left(\frac{3x_1^2 x_2}{3x_1^2 x_2} - \frac{3x_2^3}{x_2^3 + 4}\right)\right|_{\bar{x}_k} \Delta \ln x_2 = \left.\ln\left(\frac{x_2^3 + 4}{3x_2^2 x}\right)\right|_{\bar{x}_k} .$$

The new values of the variables are obtained by the relation $\ln x_{j,k+1} = \ln x_{j,k} + \Delta \ln x_j$, which can be equivalently express as $x_{j,k+1} = x_{j,k} e^{\Delta \ln x_j}$. The iterative solutions are tabulated in Table 3 for both

Newton's direct method and the transformed method.

Table 3.

Iteration	Direct Newton's Method x_1	Direct Newton's Method x_2	Transformed Newton's Method x_1	Transformed Newton's Method x_2
0	100.00	100.00	100.00	100.00
1	66.6667	77.7767	1.1470	1.9866
2	45.7058	60.0011	1.3311	1.7690
3	32.1095	45.9486	1.3363	1.7543
4	22.9972	34.9768	1.3364	1.7542
5	16.7077	26.5037		
6	12.2619	20.0165		
7	9.0628	15.0808		
8	6.7314	11.3421		
9	5.0186	8.5183		
10	3.7555	6.3903		
11	2.8270	4.7909		
12	2.1567	3.5972		
13	1.6996	2.7277		
14	1.4367	2.1438		
15	1.3429	1.8396		
16	1.3355	1.7591		
17	1.3363	1.7542		
18	1.3364	1.7542		

Simultaneous nonlinear equations, as in the prior example, can be involved in optimization problems. To find a relative optimum, the first partial derivatives of the objective function (Lagrange function if constraints are involved) are taken with respect to each variable and set equal to zero--resulting in simultaneous nonlinear equations.

The method will be demonstrated using Rosenbrock's function (11):

$$\text{minimize: } Y = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2,$$

the solution of which is $x_1 = x_2 = 1.0$. To facilitate a solution of this problem by the transformed Newton's method, the proper form must first be obtained. Upon squaring the terms, the problem becomes:

minimize: $Y = 100x_2^2 - 200x_1^2 x_2 + 100x_1^4 + 1 - 2x_1 + x_1^2$.

Taking partial derivatives and equating them to zero gives:

$\frac{\partial Y}{\partial x_1} = -400x_1 x_2 + 400x_1^3 - 2 + 2x_1 = 0$, and

$\frac{\partial Y}{\partial x_2} = 200x_2 - 200x_1^2 = 0$.

The equations can be iteratively solved as described previously giving the results tabulated in Table 4.

Table 4.

Iteration	x_1	x_2
0	100.00	100.00
1	1.01701	1.03431
2	.99925	.99850
3	1.00001	1.00002

Had Newton's direct method been used on Rosenbrock's function with an initial starting solution of $x_1 = x_2 = 100$, the first iteration would have resulted in $x_1 = 99.99995$ and $x_2 = 9999.98996$; and Newton's direct method would not have converged to the actual solution.

Difficulties arise when the problem can not be put in the correct form. If the problem had been instead:

minimize: $Y = 100(x_1^2 - x_2)^6 + (x_1 - 1)^{10}$, or,

minimize: $Y = 100(x_1^2 - x_2)^{2.1} + (x_1 - 1)^{3.6}$, multiplying the algebraic terms out completely would be very tedious or impossible. In these instances, the addition of a new variable and constraint for each such unwieldy term will greatly simplify the problem formulation. For the current example, the following new variables and relations are introduced: $x_3=x_1^2-x_2$ and $x_4=x_1-1$. Rosenbrock's function has now become a constrained optimization problem:

$$\text{minimize: } Y = x_3^2+x_4^2 ,$$
$$\text{subject to: } x_3=x_1^2-x_2 \text{ and } x_4=x_1-1 . \qquad [9]$$

Forming the unconstrained Lagrangian function, with x_5 and x_6 being Lagrange variables, results in:

$$\text{minimize: } \hat{Y} = x_3^2+x_4^2-x_5(x_3-x_1^2+x_2)-x_6(x_4-x_1+1) .$$

Partial derivatives are taken as before giving as the set of simultaneous equations:

$$2x_1x_5+x_1x_6=0,$$
$$-x_5=0,$$
$$2x_3-x_5=0,$$
$$2x_4-x_6=0,$$
$$-x_3+x_1^2-x_2=0,$$
$$-x_4+x_1-1=0 .$$

The solution is $x_1=x_2=1.0$ and $x_3=x_4=x_5=x_6=0$.

A difficulty arises with the transformed Newton's method when a very prevalent variable, such as a Lagrange variable, is equal to zero at optimality (as is the case for the Lagrange variables associated with the substitute variable constraints of [9]). The problem is that the coefficients in the linear matrix can all become very small for each such variable and instability can result. This problem can be substantially mitigated by translating the Lagrange variables by unity, or $x_{L_i} = x'_{L_i} - 1.0$, where x_{L_i} is the i^{th} original Lagrange variable. As a result of the translation of Lagrange variables, the transformed method can utilize the very helpful technique of employing substitute variables as an aid in problem formulation.

After the Lagrange variables have been translated, the simultaneous nonlinear equations are:

$$2x_1x_5 + x_1x_6 - 2x_1 - x_1 = 0,$$

$$-x_5 + 1 = 0,$$

$$2x_3 - x_6 + 1 = 0,$$

$$2x_4 - x_6 + 1 = 0,$$

$$-x_3 + x_1^2 - x_2 = 0,$$

$$-x_4 + x_1 - 1 = 0.$$

The solution is now $x_1 = x_2 = x_5 = x_6 = 1.0$ and $x_3 = x_4 = 0$.

That $x_3 = 0$ and $x_4 = 0$ at optimality does not pose a significant problem because substitution variables occur only twice in the simultaneous equations and have relatively little influence. The iterative solution for the current problem is tabulated in Table 5.

The feature of Lagrange variable translation is incorporated in the FORTRAN computer program of which a listing is given in Appendix B.

Table 5.

Iteration	x_1	x_2	x_3	x_4	x_5	x_6
0	100	100	100	100	1	1
1	.61132	.34225	.49785	.40894	1.00000	.99999
2	1.43769	2.01098	.00124	.10832	.99999	.99999
3	.97775	.95638	.00040	.03601	1.00000	.99999
4	1.00145	1.00287	.00014	.01279	.99999	.99999
5	1.00006	1.00012	.00005	.00464	.99999	1.00000
6	1.00001	1.00002	.00001	.00170	1.00000	.99999

The final example of the primal application of the transformed Newton's method will be the problem from (12):

minimize: $x_3^2+x_4^2$,

subject to: $x_3 - x_1^2 + 12x_2 + 1 = 0$

and $x_4 - 49x_1^2 - 49x_2^2 + 84x_1 + 2324x_2 + 681 = 0$.

The solution is $x_1 = 21.0266$, $x_2 = 36.7600$ and $x_3 = x_4 = 0$. The results are tabulated in Table 6, with x_5 and x_6 again being Lagrange variables.

Table 6.

Iteration	x_1	x_2	x_3	x_4	x_5	x_6
0	20.00	20.00	20.00	20.00	1.00	1.00
1	23.6726	63.0494	.4445	.4445	1.0000	1.0000
2	21.5648	38.6742	.1150	.1150	1.0000	1.0000
3	21.0295	36.7706	.0380	.0380	1.0000	1.0000
4	21.0266	36.7600	.0134	.0134	1.0000	1.0000
5	21.0266	36.7600	.0048	.0048	1.0000	1.0000

A Specific Application
To The Dual Geometric Program

The main advantage of having a primal and a dual program is that one of them usually proves easier to solve, or has other specific benefits. The preceding chapter explored the general application of the transformed Newton's method to, among other things, the primal geometric programming problem. The intention of this chapter is to illustrate how the transformed Newton's method may be employed to solve the dual geometric programming problem.

The tractability of the dual objective function is greatly enhanced if the primal problem is unconstrained. Therefore, if the primal problem is constrained, it is advisable to form the unconstrained Lagrangian as a preliminary procedure.

For the unconstrained primal geometric programming problem from (13),

$$\text{minimize: } \sum_{i=1}^{M} \sigma_i \, c_i \prod_{j=1}^{N} x_j^{\alpha_{ij}} ,$$

with the implicit constraints

$x_j > 0$, $\quad j=1,\ldots,N$;

$c_i > 0$; $\sigma_i = \pm 1$; $i=1,\ldots,M$,

the associated dual geometric programming problem is (for $\Theta > 0$ and dual variables δ_i):

maximize: $\Theta = \prod_{i=1}^{M} \left(\frac{c_i}{\delta_i}\right)^{\sigma_i \delta_i}$,

subject to: $\sum_{i=1}^{M} \sigma_i \delta_i = 1$ and $\sum_{i=1}^{M} \sigma_i \alpha_{ij} \delta_i = 0$; $j=1,\ldots,N$,

with all variables still required to be positive.

Since the natural logarithm is a monotonic function, an extreme point of $\ln\Theta$ will also be an extreme point of Θ. An equivalent dual objective function would be to maximize $\ln\Theta$. The dual objective function becomes

$$\ln\Theta = \ln \prod_{i=1}^{M} \left(\frac{c_i}{\delta_i}\right)^{\sigma_i \delta_i} ,$$

which can be alternatively represented as

$$\ln\Theta = \sum_{i=1}^{M} (\sigma_i \delta_i \ln c_i - \sigma_i \delta_i \ln \delta_i) .$$

The dual problem is now:

maximize: $\ln\Theta = \sum_{i=1}^{M} (\sigma_i \delta_i \ln c_i - \sigma_i \delta_i \ln \delta_i)$,

subject to: $\sum_{i=1}^{M} \sigma_i \delta_i = 1$ and $\sum_{i=1}^{M} \sigma_i \alpha_{ij} \delta_i = 0$; $j=1,\ldots,N$.

With the addition of one data manipulation step, the procedure for solving the dual problem will be similar to the solution technique for the primal problem. The Lagrangian of the dual problem will be formed, partial derivatives will be taken with respect to the variables and equated to zero, and the resulting simultaneous equations solved through a modified iterative application of the transformed Newton's method.

Forming the Lagrangian of the dual problem gives

maximize: $\hat{\Theta} = \sum_{i=1}^{M} (\sigma_i \delta_i \ln c_i - \sigma_i \delta_i \ln \delta_i) - \lambda_0 (\sum_{i=1}^{M} \sigma_i \delta_i - 1) - \sum_{j=1}^{N} \lambda_j (\sum_{i=1}^{M} \sigma_i \alpha_{ij} \delta_i)$,

where $\lambda_0, \lambda_1, \ldots, \lambda_N$ are Lagrange variables. The partial derivatives of $\hat{\Theta}$ with respect to the variables are, when equated to zero:

$$\frac{\partial\hat{\Theta}}{\partial\lambda_0} = 1 - \sum_{i=1}^{M}\sigma_i\ \delta_i = 0,$$

$$\sum_{i=1}^{M}\sigma_i\ \delta_i = 1, \qquad [10]$$

$$\frac{\partial\hat{\Theta}}{\partial\lambda_j} = \sum_{i=1}^{M}\sigma_i\ \alpha_{ij}\ \delta_i = 0; \qquad j=1,\ldots,N,$$

$$\sum_{i=1}^{M}\sigma_i\ \alpha_{ij}\ \delta_i = 0; \qquad j=1,\ldots,N, \qquad [11]$$

$$\frac{\partial\hat{\Theta}}{\partial\delta_i} = \sigma_i \ln c_i - \sigma_i \ln\delta_i - \sigma_i - \sigma_i\ \lambda_0 - \sum_{j=1}^{N}\sigma_i\ \alpha_{ij}\ \lambda_j = 0; \qquad i=1,\ldots,M,$$

$$\ln\delta_i + \lambda_0 + \sum_{j=1}^{N}\alpha_{ij}\ \lambda_j = \ln c_i - 1; \qquad i=1,\ldots,M. \qquad [12]$$

Equations [10], [11] and [12] represent a system of N+M+1 nonlinear equations in N+M+1 variables which are separable from one another.

The equations of [12] can also be represented by:

$$\begin{aligned} \ln\delta_1 + \lambda_0 + \alpha_{11}\lambda_1 + \ldots + \alpha_{1N}\lambda_N &= \ln c_1 - 1, \\ \ln\delta_2 + \lambda_0 + \alpha_{21}\lambda_1 + \ldots + \alpha_{2N}\lambda_N &= \ln c_2 - 1, \\ &\vdots \\ \ln\delta_M + \lambda_0 + \alpha_{M1}\lambda_1 + \ldots + \alpha_{MN}\lambda_N &= \ln c_M - 1. \end{aligned} \qquad [12]$$

Due to the structure of [12] (and since $M \geq N+1$ in order to avoid over specification of the problem), any N+1 of the equations can be used to explicitly solve for the N+1 Lagrange variables in functional terms of the natural logarithms of the M dual variables. The functional λ's can be substituted into the remaining M-N-1 equations to produce M-N-1 equations of the general form:

$$\sum_{i=1}^{M} b_{ij} \ln\delta_i = B_i; \qquad j=N+2,\ldots,M. \qquad [12a]$$

The manipulation step of eliminating the Lagrange variables is primarily intended for computational enhancement and is actually not necessary. The system of N+M+1 equations in N+M+1 variables could be used to iteratively solve for the dual and Lagrange variables utilizing the method to be described. However, if the Lagrange variables are eliminated, only M equations in M variables must be solved repetitively.

With the elimination of the Lagrange variables, the set of simultaneous equations has now become an M×M system:

$$\sum_{i=1}^{M} \sigma_i \; \delta_i = 1 \; ; \qquad [10]$$

$$\sum_{i=1}^{M} \sigma_i \; \alpha_{ij} \, \delta_i = 0 \, , \qquad j=1,\ldots,N \, ; \qquad [11]$$

$$\sum_{i=1}^{M} b_{ij} \, \ln \delta_i = B \, , \qquad j=N+2,\ldots,M \, . \qquad [12a]$$

It is this set of equations to which a modified transformed Newton's method will be applied to solve for the M dual variables. The term modified is used because only the first N+1 equations (the equations of [10] and [11]) will be 'log-linearized' by the transformed Newton's method for an iterative solution of the resulting set of M equations which are linear in the natural logarithms of the dual variables.

The procedure to log-linearize an equation is the one that was employed previously, but with the modification in the equations of [8] that $\Delta \ln x_j = \ln x_{j,k+1} - \ln x_{j,k}$ (where $\ln x_{j,k}$ is known and $\ln x_{j,k+1}$ is the next iterative solution value for $\ln x_j$) . Thus the general equation $f(\bar{\delta}) = g(\bar{\delta}) - h(\bar{\delta}) = 0$ would become:

$$\sum_{i=1}^{M}\left(\frac{\partial \ln \frac{g(\bar{\delta}_k)}{h(\bar{\delta}_k)}}{\partial \ln \delta_i}\right)\ln \delta_{i,K+1} = \ln \frac{h(\bar{\delta}_k)}{g(\bar{\delta}_k)} + \sum_{i=1}^{M}\left(\frac{\partial \ln \frac{g(\bar{\delta}_k)}{h(\bar{\delta}_k)}}{\partial \ln \delta_i}\right)\ln \delta_{i,K}$$

Using the current solution $(\bar{\delta}_k)$, the equations of [10] and [11] can be converted to a set of N+1 equations that are linear in the natural logarithms of the dual variables. New values of the dual variables can now be determined.

This dual solution technique will be demonstrated with a simple problem

minimize: $P = 5x^3 + 12x^{-2} - 2x^2$,

the dual of which is

maximize: $\Theta = \left(\frac{5}{\delta_1}\right)^{\delta_1}\left(\frac{12}{\delta_2}\right)^{\delta_2}\left(\frac{2}{\delta_3}\right)^{-\delta_3}$,

subject to: $\delta_1 + \delta_2 - \delta_3 = 1$ and $3\delta_1 - 2\delta_2 - 2\delta_3 = 0$.

The primal solution is x=1.1576, while the corresponding dual solution is $\delta_1 = .5528$, $\delta_2 = .6382$, and $\delta_3 = .1910$. The simultaneous equations that result from the partial derivatives are

$\delta_1 + \delta_2 - \delta_3 = 1$, [13]

$3\delta_1 - 2\delta_2 - 2\delta_3 = 0$, [14]

$\ln\delta_1 + \lambda_0 + 3\lambda_1 = \ln 5 - 1$, [15a]

$\ln\delta_2 + \lambda_0 - 2\lambda_1 = \ln 12 - 1$, [15b]

$\ln\delta_3 + \lambda_0 + 2\lambda_1 = \ln 2 - 1$, [15c]

Solving equations [15a] and [15b] for λ_0 and λ_1 results in:

$\lambda_0 = -.4\ln\delta - .6\ln\delta + .2\ln((12^3)(5^2)) - 1$,

$\lambda_1 = -.2\ln\delta + .2\ln\delta - .2\ln(12/5)$.

Substitution of these functional values λ_0 and λ_1 into equation [15c] gives, after collecting terms,

$$4 \ln\delta_1 + \ln\delta_2 - 5\ln\delta_3 = \ln\left(\frac{2^3}{(3)(5^4)}\right) . \qquad [15c']$$

The set of simultaneous equations has become:

$$\delta_1 + \delta_2 - \delta_3 = 1 , \qquad [13]$$

$$3\delta_1 - 2\delta_2 - 2\delta_3 = 0 , \qquad [14]$$

$$4 \ln\delta_1 + \ln\delta_2 - 5\ln\delta_3 = \ln\left(\frac{2^3}{(3)(5^4)}\right) . \qquad [15c']$$

After log-linearizing equations [13] and [14] based on the current solution, ($\bar{\delta}_k$), the set of equations are

$$\left(\frac{\delta_1}{\delta_1+\delta_2}\right)\Bigg|_{\bar{\delta}_k} \ln\delta_1 + \left(\frac{\delta_2}{\delta_2+\delta_3}\right)\Bigg|_{\bar{\delta}_k} \ln\delta_2 - \left(\frac{\delta_3}{\delta_3+1}\right)\Bigg|_{\bar{\delta}_k} \ln\delta_3 = \left(\left(\frac{\delta_1}{\delta_1+\delta_2}\right)\ln\delta_1 + \left(\frac{\delta_2}{\delta_1+\delta_2}\right)\ln\delta_2 - \left(\frac{\delta_3}{\delta_3+1}\right)\ln\delta_3 + \ln\left(\frac{\delta_3+1}{\delta_1+\delta_2}\right)\right)\Bigg|_{\bar{\delta}_k} ,$$

$$\ln\delta_1 - \left(\frac{\delta_2}{\delta_2+\delta_3}\right)\Bigg|_{\bar{\delta}_k} \ln\delta_2 - \left(\frac{\delta_3}{\delta_2+\delta_3}\right)\Bigg|_{\bar{\delta}_k} \ln\delta_3 = \left(\ln\delta_1 - \left(\frac{\delta_2}{\delta_2+\delta_3}\right)\ln\delta_2 - \left(\frac{\delta_3}{\delta_2+\delta_3}\right)\ln\delta_3 + \ln\left(\frac{2(\delta_2+\delta_3)}{3\delta_1}\right)\right)\Bigg|_{\bar{\delta}_k} ,$$

$$4 \ln\delta_1 + \ln\delta_2 - 5\ln\delta_3 = 5.4569 .$$

If, for example, $\bar{\delta}_k = (2,2,2)$, the equations are

$$1/2\ln\delta_1 + 1/2\ln\delta_2 - 2/3\ln\delta_3 = -.0566 ,$$

$$\ln\delta_1 - 1/2\ln\delta_2 - 1/2\ln\delta_3 = .2877 ,$$

$$4\ln\delta_1 + \ln\delta_2 - 5\ln\delta_3 = 5.4569;$$

and the solution is

$\ln\delta_1 = -2.8257$ and $\delta_1 = .0593$,

$\ln\delta_2 = -2.3957$ and $\delta_2 = .0911$,

$\ln\delta_3 = -2.8311$ and $\delta_3 = .0217$.

The process is continued until sufficient convergence has been achieved. Tabulated in Table 7 are the iterative solutions for this problem.

Table 7.

Iteration	δ_1	δ_2	δ_3
0	2.0	2.0	2.0
1	0.0593	0.0911	0.0217
2	0.4990	0.5790	0.1726
3	0.5524	0.6377	0.1909
4	0.5528	0.6382	0.1910
5	0.5528	0.6382	0.1910

At optimality, the dual and the primal objective functions are equal:

$$P^* = \Theta^* = \left(\frac{5}{.5528}\right)^{.5528}\left(\frac{12}{.6382}\right)^{.6382}\left(\frac{2}{.1910}\right)^{.1910}$$

$$= 14.0309 \, .$$

Using $c_i \prod_{j=1}^{N} x_j^{\alpha_{ij}} = \delta_i \, \Theta^*$, (14), to solve for the primal variable gives

$5x^3 = (14.0309)(.5528)$ and $x = 1.1576$.

This dual solution technique could have several beneficial traits. Since the equations of [10], [11] and [12a] are initially linear in either the dual variables or the natural logarithms of the dual variables, they shouldn't be highly capricious. Also, dual variables have a generally more restricted 'relevant' range than do primal variables and a starting solution is thus easier to obtain. This technique might also prove advantageous for problems with higher degrees of difficulty (defined as being equal to M-N-1) since the problems would have more static control (a larger number of equations of the type in [12a] which would not have to be approximated through log-linearization).

Conclusions And Suggestions For Further Research

Conclusions

This dissertation has presented a modification of the classical Newton's method for the solution of nonlinear equations. On the basis of limited computational evidence on the solution of signomial problems, this transformed Newton's method appears to often be superior to a direct application of Newton's method. The benefits of the transformed method over the direct method are that (a) it is less sensitive to the initial starting solution because transformed signomial equations are linearly asymptotic to the natural logarithms of the variables and (b) it provides more rapid convergence because of the transformed method's more effective approximation of signomial equations due to the use of a double nonlinear posynomial approximation as opposed to a single linear approximation as is used with a direct application of Newton's method.

Suggestions For Further Research

The transformed method should be compared to the direct method for problems involving transcendental functions. Strategies for equation decomposition could be explored for the direct inclusion of transcendental functions as well as for the inclusion of transcendental functions by Taylor's series expansion, Cebyshev series representation, and limiting techniques, (15).

Also, the dual solution method could be computer implemented and evaluated as to its convergence and accuracy properties.

Appendix A.

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     POSITIVE ROOTS OF A POLYNOMIAL USING THE TRANSFORMED NEWTONS
C         METHOD
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C     INPUT:
C           1) FORMAT(F)
C              NT=NUMBER OF TERMS
C           2) FORMAT(2F)    (ONE PER TERM)
C                 C(J)=COEFFICIENT OF TERM
C                 A(J)=EXPONENT OF TERM
C           3) FORMAT(F)
C              X=INITIAL VALUE OF VARIABLE
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      DIMENSION C(10),A(10)
      DOUBLE PRECISION S(4)
1     FORMAT(I)
2     FORMAT(2F)
3     FORMAT(1X,'ITERATION',7X,'X',20X,'F(X)',//)
4     FORMAT(1X,4X,I2,5X,E15.9,3X,E15.9)
5     FORMAT(////1X,'NEWTONS METHOD ON TRANSFORMED FUNCTION',/)
      IN=2
      READ(IN,1)NT
      DO 10 J=1,NT
10    READ(IN,2)C(J),A(J)
      READ(IN,2)X
      XI=X
      WRITE(4,5)
      WRITE(4,3)
      DO 100 IT=1,20
      DO 20 I=1,4
20    S(I)=0.0
      DO 30 J=1,NT
      VTJ=C(J)*(X**A(J))
      IR=0
      IF(VTJ.LT.0.0)IR=1
      S(IR+1)=S(IR+1)+VTJ
30    S(IR+3)=S(IR+3)+VTJ*A(J)
      XO=X
      X=X*((-S(2)/S(1))**(1./((S(3)/S(1))-(S(4)/S(2)))))
      SS=S(1)+S(2)
      WRITE(4,4)IT,X,SS
      IF(ABS(SS).LT.0.0000001)GO TO 150
100   CONTINUE
150   CALL EXIT
      END
```

Appendix B.

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       GENERAL PRIMAL GEOMETRIC PROGRAM USING THE TRANSFORMED NEWTONS
C          METHOD
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C           INPUT:
C                1) FORMAT(2I)
C                   IN=PROBLEM INPUT DEVICE NUMBER
C                2) FORMAT(2I)
C                   INV=NUMBER OF VARIABLES
C                   INC=NUMBER OF CONSTRAINTS
C                3) FORMAT(I)              (ONE PER POLYNOMIAL)
C                   KT(I)=NUMBER OF TERMS IN POLYNOMIAL I
C                4) FORMAT(20F)            (ONE PER TERM)
C                   C(I)=COEFFICIENT OF TERM I
C                   A(I,J)=EXPONENT OF VARIABLE J IN TERM I    (ONE
C                          VALUE PER VARIABLE)
C                5) FORMAT(20F)
C                   X(J)=INITIAL VALUE OF VARIABLE    (ONE PER VARIABLE)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      COMMON R(21,20),A(100,20),C(100),VT(100),X(20)
      DIMENSION IC(20),KT(20)
      DIMENSION SS(2)
1     FORMAT (20I)
2     FORMAT (20F)
3     FORMAT(1X,' RELATIVE',A5,'OF OBJECTIVE FUNCTION= ',F15.6)
4     FORMAT(1X,'X(',I2,') = ',E14.8)
5     FORMAT(1X,' NOT ENOUGH CONVERGENCE IN 10 ITERATIONS')
6     FORMAT(1X,'LINEAR MATRIX NOT INDEPENDENT--ABORT')
C     INPUT AND PREPARATION
      READ(4,1)IN
      IOUT=6
      READ(IN,1) INV,INC
      ICT=INV
      DO 25 I=1,INC+1
25    IC(I)=0
      ICC=0
      KTT=0
      DO 30 I=1,INC+1
      READ(IN,1) KT(I),IC(I)
      IF(IC(I).NE.0) ICC=ICC+1
30    KTT=KTT+KT(I)
      NT=KTT+2.*ICC
      NV=INV+INC+ICC
```

```
      DO 35 J=1,NT
      DO 35 I=INV+1,NV
35    A(J,I)=0.0
      I2=0
      DO 50 L=1,INC+1
      I1=I2+1
      I2=I1+KT(L)-1
      DO 40 I=I1,I2
      READ(IN,2)C(J),(A(I,J),J=1,INV)
      IF(L.EQ.1) GO TO 40
38    A(I,L+INV+ICC-1)=1.0
      C(I)=-C(I)
40    CONTINUE
      IF((L.EQ.1).OR.(IC(L).EQ.0)) GO TO 50
C     SLACK VARIABLES
      KT(L)=KT(L)+1
      I2=I2+1
      ICT=ICT+1
      C(I2)=1.0
      A(1,ICT)=0.0000001
      A(I2,ICT)=1.0
      A(I2,L+INV+ICC-1)=1.0
50    CONTINUE
      DO 53 I=1,INV
      READ(IN,2)X(I)
      IF(X(I).EQ.0.0) X(I)=10.0
53    WRITE(IOUT,4)I,X(I)
1505  FORMAT(1H0,I2,3X,E13.6,10F9.4)
      DO 1504 I=1,NT
 1504 WRITE(IOUT,1505)I,C(I),(A(I,J),J=1,NV)
C     INITIAL VALUES OF LAGRANGE VARIABLES
      DO 58 I=INV+1,NV
58    X(I)=1.001
C     END OF INPUT AND PREPARATION
C     EVALUATE TERMS
      DO 120 IT=1,10
      DO 60 J=1,NT
      VT(J)=C(J)
      DO 60 I=1,NV
      IF(A(J,I).EQ.0.0) GO TO 60
      VT(J)=VT(J)*(X(I)**A(J,I))
60    CONTINUE
      DO 90 IV=1,NV
      DO 65 J=1,NV+1
      R(IV,J)=0.0
65    R(IV+1,J)=0.0
      SS(1)=0.0
      SS(2)=0.0
      IXS=KT(1)
      L=1
      DO 70 J=1,NT
```

```
      IF(J.LE.IXS) GO TO 68
      L=L+1
      IXS=IXS+KT(L)
68    VTJ=VT(J)*(A(J,IV))
      IF(VTJ.EQ.Ø.Ø) GO TO 7Ø
      IM=2
      IF((IV.GT.(INV+ICC)).OR.(J.LE.KT(1))) IM=1
C     TRANSLATE LAGRANGE VARIABLES
      XT=1.Ø
      DO 7Ø II=1,IM
      IF(II.GT.1) XT=-(1.Ø/X(L+INV+ICC-1))
      IR=Ø
      IF(VTJ*XT.LT.Ø.) IR=1
      SS(IR+1)=SS(IR+1)+VTJ*XT
      DO 7Ø I=1,NV
      IF((II.GT.1).AND.(I.GT.(INV+ICC))) GO TO 7Ø
      AI=Ø.Ø
      IF(I.EQ.IV) AI=1.
      R(IV+IR,I)=R(IV+IR,I)+VTJ*(A(J,I)-AI)*XT
7Ø    CONTINUE
      IF(SS(1)*SS(2).NE.Ø.Ø) GO TO 78
      IF(SS(1).NE.Ø.Ø) GO TO 76
      SS(1)=1.Ø
      SSS=3Ø.
      GO TO 79
76    SS(2)=1.Ø
      SSS=-3Ø.
      GO TO 79
78    SSS=ALOG(-SS(2)/SS(1))
79    DO 8Ø I=1,NV
8Ø    R(IV,I)=R(IV,I)/SS(1)-R(IV+1,I)/SS(2)
9Ø    R(IV,NV+1)=SSS
C     SOLVE LINEAR SYSTEM
      NN=NV-1
      DO 19Ø K=1,NN+1
      IF(R(1,1).NE.Ø.Ø) GO TO 168
      DO 165 KK=2,NN+2-K
      IF(R(KK,1).EQ.Ø.Ø) GO TO 165
      DO 163 J=1,NN+3-K
163   R(1,J)=R(1,J)+R(KK,J)
      GO TO 168
165   CONTINUE
      WRITE(IOUT,6)
      CALL EXIT
168   PIV=R(1,1)
      DO 175 J=1,NN+2-K
175   R(NN+2,J)=R(1,J+1)/PIV
      DO 18Ø I=1,NN
      DO 18Ø J=1,NN+2-K
18Ø   R(I,J)=R(I+1,J+1)-R(I+1,1)*R(NN+2,J)
      DO 19Ø J=1,NN+2-K
```

```
190    R(NN+1,J)=R(NN+2,J)
C      DETERMINE NEW VARIABLE VALUES AND CHECK CONVERGENCE
       NGV=0
       DO 100 I=1,NV
       XX=X(I)
       X(I)=X(I)*(EXP(R(I,1)))
       IF(ABS(XX-X(I))/XX.LT.5.E-6.OR.X(I).LT.2.E-3) NGV=NGV+1
100    CONTINUE
       IF(NGV.EQ.NV) GO TO 130
120    CONTINUE
       WRITE(IOUT,5)
       GO TO 155
C      EVALUATE OBJECTIVE FUNCTION
130    VOF=0.0
       VOFT=0.0
       DO 135 J=1,KT(1)
       DO 135 I=1,INV
       IF(A(J,I).EQ.0) GO TO 135
       ITV=I
       GO TO 138
135    CONTINUE
138    DO 150 J=1,KT(1)
       VOF=VOF+VT(J)
150    VOFT=VOFT+(VT(J)*(1.01**A(J,ITV)))
       IOP=' MIN '
       IF(VOF.GT.VOFT) IOP=' MAX '
       WRITE(IOUT,3) IOP,VOF
155    DO 160 I=1,INV
       WRITE(4,4)I,X(I)
160    WRITE(IOUT,4)I,X(I)
999    CALL EXIT
       END
```

Literature Cited

1. F. S. Acton, Numerical Methods That Work. New York, N. Y.: Harper and Row, p. 51 (1970).
2. R. J. Duffin, E. L. Peterson and C. Zener, Geometric Programming--Theory And Application. New York, N. Y.: John Wiley and Sons, Inc. (1967).
3. R. J. Duffin and E. L. Peterson, "Geometric Programming With Signomials", Journal Of Optimization Theory And Application, 11, pp. 3-35 (1973).
4. C. Zener, Engineering Design By Geometric Programming. New York, N. Y.: Wiley-Interscience (1971).
5. G. A. Kochenberger, R. E. D. Woolsey and B. A. McCarl, "On The Solution Of Geometric Programs Via Separable Programming", Operations Research Quarterly, Vol. 24, No. 2 (1973).
6. M. Avriel and A. C. Williams, "An Extension Of Geometric Programming With Applications In Engineering Optimization", Journal Of Engineering Mathematics, Vol. 5, No. 3 (1971).
7. R. J. Duffin, E. L. Peterson and C. Zener, Geometric Programming--Theory And Application. New York, N. Y.: John Wiley and Sons, Inc., p. 2 (1967).
8. Ibid., p.98.

9. J. M. Henderson and R. E. Quandt, Microeconomic Theory--A Mathematical Approach. New York, N. Y.: McGraw-Hill Book, Co., p. 27 (1971).

10. S. D. Conte, Elementary Numerical Analysis. New York, N. Y.: McGraw-Hill Book, Co., p. 53 (1965).

11. D. M. Himmelblau, Applied Nonlinear Programming. New York, N. Y.: McGraw-Hill, Inc., p. 394 (1972).

12. Ibid., p. 428.

13. D. J. Wilde and C. S. Beightler, Foundations Of Optimization. Englewood Cliffs, N. J.: Prentice-Hall, Inc., pp. 99-131 (1967).

14. Ibid., p. 126.

15. R. J. Duffin, E. L. Peterson and C. Zener, Geometric Programming--Theory And Application. New York, N. Y.: John Wiley and Sons, Inc., pp. 100-101 (1967).