To the Graduate Council:

I am submitting herewith a thesis written by Oliver K. Tavakoli entitled "An Extension of Generalized Pseudo-Involutions." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Mathematics.

Randall E. Cline

Randall E. Cline, Major Professor

We have read this thesis
and recommend its acceptance:

Steven M. Serbin

[illegible]

Accepted for the Council:

[illegible]

Vice Provost
and Dean of The Graduate School

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Master's degree at The University of Tennessee, Knoxville, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made.

Permission for extensive quotation from or reproduction of this thesis may be granted by my major professor, or in his absence, by the Head of Interlibrary Services when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature Oliver Kourosh Tavakoli

Date May 7, 1985

AN EXTENSION OF GENERALIZED PSEUDO - INVOLUTIONS

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Oliver K. Tavakoli

June 1985

DEDICATION

To my mother, who instilled in me my love of education,

and to Jo Anne, who kept me going when everything seemed impossible.

ACKNOWLEDGEMENTS

The author is extremely grateful to his major professor, Dr. Randall E. Cline, without whose patience and guidance this thesis would have been almost impossible to complete.

The author would like to thank Dr. Robert M. McConnel for his close association with this project.

For his assistance and service as a committee member the author wishes to thank Dr. Steven M. Serbin.

For his special interest in the thesis the author would also like to thank Dr. Jacob Beard.

For his advice and help with writing this thesis under SCRIPT, the author wishes to thank Dr. Charles P. Pfleeger.

The author offers his special gratitude to his mother, Karin Tavakoli, who pushed him harder than anyone could, and to his father, Ardeshir Tavakoli, who stood by him every step of the way.

Finally, the author is extremely grateful to the Harriman and Treece households for making Knoxville a home away from home.

ABSTRACT

Solutions of the nonlinear matrix equation $\alpha A + \beta X = \alpha\beta AX$ are characterized in terms of the Drazin inverse of A, and are shown to provide an extension of generalized pseudo-involutions when $X = A_d$. Computational techniques are developed to construct general n by n matrices A such that $X = A_d$, and corresponding complete sets of eigenvectors and associated generalized eigenvectors. These results are illustrated by numerical examples using real matrices and also integral matrices whose elements are chosen from a finite field.

TABLE OF CONTENTS

SECTION		PAGE
1.	INTRODUCTION	1
2.	BACKGROUND	2
3.	MAIN RESULTS	6
4.	COMPUTATIONAL TECHNIQUES	20
5.	NUMERICAL EXAMPLES	27
6.	AREAS FOR FURTHER RESEARCH	32
LIST OF REFERENCES		33
APPENDICES		35
APPENDIX A.	EXPLANATION OF SUBROUTINES	36
APPENDIX B.	COMPUTER PROGRAMS	39
APPENDIX C.	NUMERICAL EXAMPLES	61
VITA		67

1. INTRODUCTION

In a monograph on involutary matrices, Cline and McConnel [3] introduced a class of nonsingular matrices which reduce to involutions over fields of characteristic p = 2. These matrices, called pseudo-involutions, are defined by a single additive relationship satisfied by a matrix and its inverse, and have the property that such matrices have only eigenvalue $\lambda = 1$. Using the Drazin inverse, it was also shown that pseudo-inverses are a special case of a more general class of matrices. The resulting matrices, called generalized pseudo-involutions, have eigenvalues $\lambda = 1$ and $\lambda = 0$, and reduce directly to pseudo-involutions for nonsingular matrices. The purpose of this thesis is to consider yet a further extension of generalized pseudo-involutions to matrices with at most two distinct nonzero eigenvalues.

In Section 2 we summarize the necessary background material on Drazin inverses and full rank factorizations. The main mathematical results are established in Section 3, and computational techniques that can be used to implement some of these results are considered in Section 4. Section 5 includes a number of small numerical examples to illustrate the material, and Section 6 provides a list of some areas that might be considered for further research. Finally, the thesis concludes with appendices that provide a description and listing of the program used to compute two of the examples in Section 5, and further numerical results.

2. BACKGROUND

Following the general usage of notation in [3], matrices will be denoted by capital Latin letters A,B,C,..., and scalars by small Greek letters $\alpha,\beta,\gamma,\ldots$, where elements of matrices and scalars are assumed to be from arbitrary fields of characteristic $p \geq 0$. For computational purposes to illustrate the resulting material, however, matrices with real elements will be used when $p = 0$, and matrices with integers reduced modulo p will be used when $p > 0$. The letters I and O will denote the identity matrix and the null matrix, respectively, with size determined by context, and eigenvalues of a matrix will be denoted by λ or λ_i, as appropriate.

Given an arbitrary square matrix A over any field, Drazin [4] showed that there exists a unique matrix X such that

(2.1) $A^k = A^{k+1}X$ for some positive integer k,

(2.2) $X = X^2A$,

(2.3) $AX = XA$.

For A=O, X=O, whereas to construct X when $A \neq O$, it was noted that A satisfies a polynomial equation of the form

$$(2.4) \qquad \sum_{i=1}^{t} \alpha_i A^i = O$$

for some scalars $\alpha_1,\ldots,\alpha_t$ with $t \leq n^2+1$ if A is n by n. Then if α_k denotes the first nonzero scalar in (2.4),

$$(2.5) \qquad A^k = A^{k+1}\left(-\alpha_k^{-1} \sum_{i=k+1}^{t} \alpha_i A^{i-k-1} \right),$$

and taking

$$(2.6) \qquad U = -\alpha_k^{-1} \sum_{i=k+1}^{t} \alpha_i A^{i-k-1}$$

the matrix X can be written as

$$(2.7) \qquad X = A^k U^{k+1}.$$

The unique matrix X is called the Drazin inverse of A and is denoted by $X=A_d$. Also, the smallest positive integer such that (2.1) holds is called the (Drazin) index of A, and it has been shown [2] that k is the smallest positive integer such that A^k and A^{k+1} have the same rank, and that $\text{rank}(A^k) = \text{rank}(A_d)$. (Clearly, if A is nonsingular, $A_d=A^{-1}$ and (2.1) reduces to $I = AA^{-1}$, in which case the observation on the rank of A^k and A^{k+1} also holds trivially with k=0 and $A^0=I$. Also, for the special case k=1, A_d is sometimes called the group inverse of A, and designated alternately by $A^{\#}$.)

For A an arbitrary m by n matrix over any field with rank $r > 0$, factorizations of the form A = EG with E an m by r matrix and G an r by n matrix are called full rank factorizations. Although such factorizations are not unique and can be constructed in various ways, they provide an alternative technique for forming A_d for square matrices A, and also for forming other types of generalized inverses of rectangular matrices. In particular, if A = EG is any factorization of an n by n matrix with rank r, $0 < r < n$, then

$$(2.8) \qquad A_d = (EG)_d = E(GE)_d^2 G.$$

Whence if A = EG is a full rank factorization, (2.8) can be used to reduce the problem of constructing A_d to that of forming $(GE)_d$ where GE is r by r.

As discussed in [2] and [3], if $GE \neq 0$ and $\text{rank}(GE) < r$, this procedure can be continued by next forming a full rank factorization of GE. In contrast, for the special case in which A has index $k = 1$, then A and A^2 have the same rank, and with $A = EG$ a full rank factorization, GE is nonsingular. Hence $(GE)_d = (GE)^{-1}$, and (2.8) becomes

(2.9) $\quad A_d = E(GE)^{-2}G.$

Returning to the defining equations for A_d, it follows using (2.3) that (2.1) and (2.2) can be written as $A^k = A^k XA$ and $X = XAX$, respectively. Thus, when $k = 1$, $X = A_d$ is the unique matrix which commutes with A and satisfies

(2.10) $\quad AXA = A$

and

(2.11) $\quad XAX = X.$

For subsequent use in the proof of Theorem 6 in Section 3, it is noted that for arbitrary rectangular matrices A and X, any two of the conditions (2.10), (2.11), and

(2.12) $\quad \text{rank}(A) = \text{rank}(X)$

imply the third. This result, established in [1] for complex matrices, is clearly seen to hold for matrices over any field by introducing full rank factorizations $A = EG$ and $X = YZ$ and noting that (2.10) or (2.11), together with (2.12), imply $GY = (ZE)^{-1}$, whereas (2.10) and (2.11) obviously imply that A and X have the same rank.

As was indicated in the Introduction, the purpose of this thesis is to consider an extension of generalized pseudo-involutions. Using the

terminology of [3], a nonsingular matrix A over any field is called a pseudo-involution if and only if

(2.13) $A + A^{-1} = 2I,$

whereas A is called a generalized pseudo-involution if and only if

(2.14) $A + A_d = 2AA_d.$

Thus, with $A_d = A^{-1}$ when A is nonsingular, (2.13) is clearly a special case of (2.14). Also, it can be shown that generalized pseudo-involutions have only eigenvalues $\lambda = 0$ and $\lambda = 1$, so if A is nonsingular, pseudo-involutions have only eigenvalue $\lambda = 1$. In both cases, techniques can be developed for constructing complete sets of associated eigenvectors where, excluding certain trivial types of matrices, some of these eigenvectors are generalized eigenvectors. The eigenvalue-eigenvector structure of these matrices will also follow from the extension of generalized pseudo-involutions to be considered in Section 3.

3. MAIN RESULTS

It is to be shown in this section that generalized pseudo-involutions are special cases of a class of matrices with cubic minimal polynomials. The first result, however, provides a direct expression for constructing A_d for any matrix A with index k = 1.

Given an arbitrary square matrix $A \neq 0$ over any field, it is seen at once that A_d can be constructed using (2.6) and (2.7) by taking the α_i in (2.4) as the coefficients of the characteristic polynomial, $f(\lambda)$, of A or the minimal polynomial, $m(\lambda)$, of A, where if $m(\lambda)=f(\lambda)$, A is sometimes said to be nonderogatory [5]. Now if A is nonsingular and the α_i are coefficients of $m(\lambda)$, (2.4) reduces to

$$(3.1) \qquad \sum_{i=1}^{t} \alpha_i A^{i-1} = 0,$$

in which case, with $A^0=I$ and $\alpha_1 \neq 0$ since the constant term in $f(\lambda)$ is nonzero, A^{-1} can be written as

$$(3.2) \qquad A^{-1} = -\alpha_1^{-1} \sum_{i=2}^{t} \alpha_i A^{i-2}.$$

That the corresponding form for A_d in Lemma 1 reduces to (3.2) when A is nonsingular is easily seen.

Lemma 1 : Let A be an arbitrary square matrix over any field with index k = 1 and minimal polynomial

$$(3.3) \qquad m(\lambda) = \alpha_1\lambda + \ldots + \alpha_t\lambda^t.$$

Then

$$(3.4) \qquad A_d = \alpha_1^{-1} \left[\alpha_2 \alpha_1^{-1} \sum_{i=2}^{t} \alpha_i A^{i-1} - \sum_{i=3}^{t} \alpha_i A^{i-2} \right].$$

Proof : A with index k=1 implies A and A^2 have the same rank. Hence, using the polynomial m(A)=0 in (2.4), (2.5) holds with k=1, so that $\alpha_1 \neq 0$. Also, using $A=A^2A_d$ from (2.1), $m(A)A_d=0$ gives

$$(3.5) \qquad \alpha_1 AA_d + \sum_{i=2}^{t} \alpha_i A^{i-1} = 0.$$

In a similar manner, with $A_d=AA_d^2$ from (2.2), since A and A_d commute, $m(A)A_d^2 = 0$ gives

$$(3.6) \qquad \alpha_1 A_d + \alpha_2 AA_d + \sum_{i=3}^{t} \alpha_i A^{i-2} = 0.$$

The expression in (3.4) now follows directly from (3.6) if $\alpha_2=0$, and by substituting AA_d from (3.5) into (3.6) if $\alpha_2 \neq 0$. □

It should be noted that if A is nonsingular, (3.2) follows directly from (3.6), where A now has minimal polynomial $\lambda^{-1}m(\lambda)$.

The following result provides an initial starting point for extending generalized pseudo-involutions.

Theorem 2 : Let A be any arbitrary nonzero square matrix over any field and let $\alpha \neq 0$ be any scalar with α^{-1} not an eigenvalue of A. For any scalar $\beta \neq 0$, there is a unique matrix X such that A and X commute and

$$(3.7) \qquad \alpha A + \beta X = \alpha\beta AX.$$

Proof : With α^{-1} not an eigenvalue of A, $\alpha A-I$ is nonsingular, as $\alpha A-I$ singular would imply that there exists a vector $y \neq 0$ such that $(\alpha A-I)y=0$ and thus $Ay=\alpha^{-1}y$. Now if

(3.8) $\beta X = I + (\alpha A - I)^{-1}$,

multiplying by $(\alpha A-I)$ on the left gives $\alpha\beta AX - \beta X = \alpha A$ so that (3.7) holds. Moreover, with $\beta X-I = (\alpha A-I)^{-1}$, $(\beta X-I)(\alpha A-I) = I$ and thus (3.7) gives $AX = XA$. Finally, if (3.7) also holds for A, α, β and some matrix Y, then $(\alpha A-I)\beta X = \alpha A = (\alpha A-I)\beta Y$ implies that $X = Y$. □

As shown in Theorem 2, the matrix X is uniquely determined by A, α and β. Also, it is seen from (3.8) that A, X and either α or β determines the other scalar uniquely so that (3.7) holds. The following theorem provides conditions to have (3.7) hold for alternative choices of α and β.

Theorem 3 : Let A, X, α and β satisfy Theorem 2 in any field of cardinality greater than two. A necessary condition to have

(3.9) $\gamma A + \delta X = \gamma\delta AX$

for some nonzero scalars $\gamma\neq\alpha$ and $\delta\neq\beta$ is that $A^2 = \eta A$, in which case $\alpha\eta\neq 1$ and

(3.10) $X = \alpha\beta^{-1}(\alpha\eta - 1)^{-1}A$.

Conversely, a sufficient condition is that $A^2 = \eta A$, where if $\eta \neq 0$, $\alpha\eta \neq 1$ and there exists a nonzero scalar $\gamma \neq \alpha$ with $\gamma\eta \neq 1$.

Proof : If A and X satisfy (3.7) and also (3.9) for some nonzero scalars $\gamma\neq\alpha$ and $\delta\neq\beta$, then (3.8) and the corresponding expression $\delta X = I + (\gamma A-I)^{-1}$ combine to give

$$\beta^{-1}\alpha A(\alpha A - I)^{-1} = \delta^{-1}\gamma A(\gamma A - I)^{-1}.$$

Therefore, since $(\alpha A - I)$ and $(\gamma A - I)$ commute,

$$\beta^{-1}\alpha A(\gamma A - I) = \delta^{-1}\gamma A(\alpha A - I)$$

so that

$$\alpha\gamma(\delta - \beta)A^2 = (\alpha\delta - \beta\gamma)A.$$

Hence, with α, γ and $(\delta - \beta)$ nonzero, $A^2 = \eta A$, where

$$\eta = (\delta - \beta)^{-1}(\gamma^{-1}\delta - \alpha^{-1}\beta).$$

Now $\alpha\eta \neq 1$, for if $\eta \neq 0$ and $\alpha\eta = 1$, then

$$\alpha\gamma^{-1}\delta - \beta = \delta - \beta$$

implies $\alpha = \gamma$, a contradiction. Consequently, with $A^2 = \eta A$ and $\alpha\eta \neq 1$, multiplying each side of (3.7) by A gives

$$\alpha\eta A + \beta AX = \alpha\beta\eta AX$$

or

$$\alpha\beta AX = \alpha^2\eta(\alpha\eta - 1)^{-1}A = \alpha A + \alpha(\alpha\eta - 1)^{-1}A.$$

Whence $\beta X = \alpha(\alpha\eta - 1)^{-1}A$, and (3.10) follows.

Conversely, if $A^2 = 0$, multiplying each side of (3.7) by A gives $\beta AX = 0$, so

$$\alpha A + \beta X = 0$$

and

$$X = -\alpha\beta^{-1}A,$$

which is (3.10) with $\eta=0$, and (3.9) holds for any γ and δ with $\gamma\delta^{-1} = \alpha\beta^{-1}$.

Finally, if $A^2 = \eta A$ with $\eta \neq 0$ and $\alpha\eta \neq 1$, X has the form in (3.10), and for any nonzero scalar $\gamma \neq \alpha$ with $\gamma\eta \neq 1$, the relation

$$\gamma A + \gamma(\gamma\eta - 1)^{-1}A = \gamma^2\eta(\gamma\eta - 1)^{-1}A$$

reduces to (3.9) with

$$\delta = \gamma(\gamma\eta - 1)^{-1}\alpha^{-1}\beta(\alpha\eta - 1).$$

Also, $\delta \neq \beta$ since $\gamma \neq \alpha$. □

Observe in Theorem 3 that if A is idempotent so $\eta = 1$ and $A = A_d$, then $X = A_d$ is obtained from (3.10) by taking $\alpha = \beta = 2$. In contrast, if $A^2 = 0$, then $A_d = 0$ but $X = -\alpha\beta^{-1}A \neq 0$. Thus it is of interest to consider conditions to have $X = A_d$, and various equivalent conditions will be

established in Theorem 6. We turn first, however, to representations for X in terms of A and A_d.

Theorem 4 : Let A, X, α and β be as in Theorem 2 where A has index $k \geq 1$. Then X can be written as

$$X = -\beta^{-1}\left[(\alpha^{-1}A_d - I)^{-1} + (I - AA_d)\sum_{i=0}^{k-1}\alpha^i A^i\right]. \tag{3.11}$$

Proof : If (3.7) holds, then

$$\alpha^2A^2 + \alpha\beta AX = \alpha^2\beta A^2X,$$

so that

$$\beta(\alpha^2A^2 - I)X = \alpha A + \alpha^2A^2.$$

Now proceeding by induction using the fact that (3.7) implies

$$\alpha^mA^m + \alpha^{m-1}\beta A^{m-1}X = \alpha^m\beta A^mX$$

for any integer $m \geq 1$, and substituting for $\alpha^{m-1}\beta A^{m-1}X$ yields

$$\beta(\alpha^mA^m - I)X = \sum_{i=1}^{m}\alpha^iA^i \tag{3.12}$$

for all positive integers m. Thus if $A^k = 0$ for some positive integer k, then

$$X = -\beta^{-1}\sum_{i=1}^{k-1}\alpha^iA^i,$$

which is (3.11) in this case since $A_d = 0$. Assuming $A^k \neq 0$ with k the index of A, then $A_d \neq 0$, and taking $m = k$ in (3.12) and using the relationship $(I - AA_d)A^k = 0$ from (2.1),

$$(3.13) \qquad -\beta(I - AA_d)X = (I - AA_d)\sum_{i=1}^{k-1} \alpha^i A^i$$

where the summation on the right hand side is the null matrix if k = 1, as then $AA_dX = X$ directly from (3.7). Also, multiplying each side of (3.7) on the left by $\alpha^{-1}A_d$ gives

$$AA_d + \alpha^{-1}\beta A_dX = \beta AA_dX,$$

since A and A_d commute, which can be used to write

$$\beta(I - AA_d)X = \beta X - AA_d - \alpha^{-1}\beta A_dX = \beta(I - \alpha^{-1}A_d)X - AA_d,$$

so (3.13) becomes

$$(3.14) \qquad \beta(\alpha^{-1}A_d - I)X = -AA_d + (I - AA_d)\sum_{i=1}^{k-1} \alpha^i A^i.$$

Moreover, $\alpha^{-1}A_d - I$ is nonsingular. For if $(\alpha^{-1}A_d - I)y=0$ for some vector $y\neq 0$, then

$$(3.15) \qquad A_dy = \alpha y.$$

Multiplying each side of (3.15) by A_dA on the left and simplifying using $A_d^2A=A_d$ gives $A_dy=\alpha A_dAy =\alpha(A_dA)^ky=\alpha A_d^kA^ky$, since A_dA is idempotent and A_d and A commute, by (2.2) and (2.3). Hence $A^ky \neq 0$ since $\alpha y \neq 0$. Also, multiplying each side of (3.15) by A^{k+1} on the left gives $A^ky = \alpha A^{k+1}y$ or $A(A^ky) = \alpha^{-1}(A^ky)$, which contradicts the assumption that α^{-1} is not an eigenvalue of A. With $\alpha^{-1}A_d - I$ nonsingular, (3.14) can then be written as

$$(3.16) \qquad \beta X = (\alpha^{-1}A_d - I)^{-1}[\, -AA_d + (I - AA_d)\sum_{i=1}^{k-1} \alpha^i A^i \,],$$

and the expression in (3.11) follows by noting that

$$(\alpha^{-1}A_d - I)AA_d = \alpha^{-1}A_d - AA_d$$

implies

$$(\alpha^{-1}A_d - I)^{-1}AA_d = (\alpha^{-1}A_d - I)^{-1}\alpha^{-1}A_d - AA_d$$
$$= I - AA_d + (\alpha^{-1}A_d - I)^{-1}$$

and thus $(\alpha^{-1}A_d - I)^{-1}(I - AA_d) = -(I - AA_d)$, and that $I - AA_d$ can be included in the summation in (3.16) by changing the lower limit to $i = 0$. □

It is noted that the form for X in (3.11) when $A^k = 0$ and $k \geq 3$ is a direct extension of (3.10) when $A^2 = 0$. It will also follow from the discussion of applications of Corollary 5 that there is a corresponding extension of (3.10) for idempotent matrices to matrices A such that $A^n = A$, $n \geq 2$.

As shown in Corollary 5, the idempotent matrix $A_dA = AA_d$ can always be represented as a sum of three nonsingular matrices if A has index $k = 1$.

Corollary 5 : Let $A \neq 0$ be an arbitrary square matrix with index $k = 1$ over any field. For any scalar $\alpha \neq 0$ with α^{-1} not an eigenvalue of A,

(3.17) $$AA_d = 2I + (\alpha A - I)^{-1} + (\alpha^{-1}A_d - I)^{-1}.$$

Proof : If $k = 1$, (3.11) reduces to

$$X = -\beta^{-1}[\,(\alpha^{-1}A_d - I)^{-1} + (I - AA_d)\,],$$

and equating this expression to the form for X in (3.8) gives (3.17). □

For computational purposes, (3.17) thus provides a method for constructing both $(\alpha A - I)^{-1}$ and $(\alpha^{-1}A_d - I)^{-1}$ using a single application of a matrix inversion subroutine. It is also noted that if A is a nonsingular matrix, $A_d = A^{-1}$, and (3.17) reduces to $-I = (\alpha A - I)^{-1} + (\alpha^{-1}A^{-1} - I)^{-1}$, which can be verified directly starting with $(\alpha A - I)(\alpha A - I)^{-1} = I$. As another application, if A is any matrix such that $A^n = A$, where $n \geq 3$, then $A_d = A^{n-2}$, and utilizing Corollary 5 we have

$$A^{n-1} = 2I + (\alpha A - I)^{-1} + (\alpha^{-1}A^{n-2} - I)^{-1}.$$

Now multiplying by $\alpha^{n-1}(\alpha^{-1}A^{n-2} - I)$ on the left gives

$$\alpha^{n-2}A^{n-2} - \alpha^{n-1}A^{n-1} = 2\alpha^{n-2}A^{n-2} - 2\alpha^{n-1}I + (\alpha^{n-2}A^{n-2} - \alpha^{n-1}I)(\alpha A - I)^{-1} + \alpha^{n-1}I$$

which can be rewritten as

$$(\alpha^{n-2}A^{n-2} - \alpha^{n-1}I)(\alpha A - I)^{-1} = \alpha^{n-1}I - \alpha^{n-2}A^{n-2} - \alpha^{n-1}A^{n-1}.$$

Hence, rearranging the terms yields

$$\alpha^{n-2}A^{n-2}(I + (\alpha A - I)^{-1}) - \alpha^{n-1}(\alpha A - I)^{-1} = \alpha^{n-1}I - \alpha^{n-1}A^{n-1}.$$

Using the fact that $\alpha A(\alpha A - I)^{-1} = I + (\alpha A - I)^{-1}$, this last expression may then be written as

$$\alpha^{n-2}A^{n-2} + \alpha^{n-3}A^{n-3} + \ldots + \alpha A + I + (\alpha A - I)^{-1} - \alpha^{n-1}(\alpha A - I)^{-1} = \alpha^{n-1}I - \alpha^{n-1}A^{n-1}.$$

Whereupon, for any α such that $\alpha^{n-1} \neq 1$,

$$(\alpha A - I)^{-1} = -I + (\alpha^{n-1} - 1)^{-1} \sum_{i=1}^{n-1} \alpha^i A^i,$$

which can be used in (3.8) to write X in this case as

$$X = \beta^{-1}(\alpha^{n-1} - 1)^{-1} \sum_{i=1}^{n-1} \alpha^i A^i,$$

thereby providing an extension of (3.10) for idempotent matrices to matrices such that $A^n = A$, $n \geq 2$.

Having established general representations for X, we next consider conditions to give $X = A_d$, and begin by relating the eigenvalue-eigenvector structure of matrices A and X satisfying (3.7).

That $\alpha\eta \neq 1$ in the proof of sufficiency in Theorem 3 is also apparent by noting that $A^2 = \eta A$ implies A has eigenvalues $\lambda = 0$ and $\lambda = \eta$, and if $\eta \neq 0$, $\alpha^{-1} \neq \eta$, by the hypothesis in Theorem 2. More generally, observe that if (3.7) holds and λ is any eigenvalue of A with y an associated eigenvector,

$$\alpha\lambda y + \beta X y = \alpha\beta\lambda X y,$$

since A and X commute. Thus $Xy = 0$ if $\lambda = 0$, and

$$Xy = \alpha\beta^{-1}\lambda(\alpha\lambda - 1)^{-1}y$$

if $\lambda \neq 0$, so that A and X have the same eigenvectors, and if $\lambda \neq 0$, the corresponding eigenvalue of X is given by $\alpha\beta^{-1}\lambda(\alpha\lambda - 1)^{-1}$. Furthermore,

$$AXy = \alpha\beta^{-1}\lambda^2(\alpha\lambda - 1)^{-1}y \tag{3.18}$$

provides a general form for eigenvalues of AX which can be used in establishing one of the implications in the following theorem on equivalent conditions to have $X = A_d$.

Theorem 6 : Let A and X be matrices and α and β be scalars as in Theorem 2. Then the following conditions are equivalent:

(i) $X = A_d$,

(ii) $X = (\alpha^2 - \alpha\beta^{-1})A - \alpha^2\beta^{-1}A^2$,

(iii) $AXA = A$,

(iv) $XAX = X$,

(v) $(AX)^2 = AX \neq 0$,

(vi) $\alpha A^3 - \alpha\beta A^2 + \beta A = 0$.

Proof : If A and X satisfy (3.7), then $\alpha A = \beta X(\alpha A - I)$ with $\alpha A - I$ nonsingular implies that A and X have the same rank. Thus if $X = A_d$, A has index $k = 1$, and (iii) and (iv) hold by the defining relations for the Drazin inverse of A. Moreover, with $X = A_d$ and $A^2A_d = A$ in this case,

multiplying each side of (3.7) on the left by A^2 gives (vi). In a similar manner, with A and X having the same rank, the previously noted implications of (2.10), (2.11) and (2.12) show that (iii) implies (iv) and (iv) implies (iii). Therefore, with AX = XA, (iii) and (iv) imply (i).

Next, if (vi) holds, then with $\beta \neq 0$, $\beta A = A^2(\alpha\beta I - \alpha A)$ implies A and A^2 have the same rank. Thus A has index k = 1, and $A = A^2 A_d$. Multiplying (vi) by A_d^2 gives

$$\alpha A - \alpha\beta A A_d + \beta A_d = 0$$

so that $\alpha A + \beta A_d = \alpha\beta A A_d$. Thus (i) holds, since the matrix X in (3.7) is unique. Also, if (vi) holds, $X = A_d$ has the form in (ii), by Lemma 1. On the other hand, if (ii) holds, then

$$\alpha\beta AX = \alpha^2(\alpha\beta - 1)A^2 - \alpha^3 A^3,$$

and (vi) follows by substituting αA, βX and $\alpha\beta AX$ into (3.7).

Now (i) implies (v), since AA_d is idempotent, and with $A \neq 0$, $AA_d \neq 0$, for otherwise $A_d = 0$. But A and A_d have the same rank, as A has index k = 1. Finally, since any idempotent matrix has only eigenvalues 0 and 1, (v) implies A has at least one eigenvalue $\lambda \neq 0$. For if all eigenvalues $\lambda = 0$, $A^j = 0$ for some integer $j > 1$ and $AX = (AX)^j = A^j X^j = 0$, as A and X commute. Using any nonzero eigenvalue λ in (3.18) then gives

$$\alpha\beta^{-1}\lambda^2(\alpha\lambda - 1)^{-1} = 1.$$

Hence

(3.19) $$\alpha\lambda^2 - \alpha\beta\lambda + \beta = 0,$$

which implies (vi). □

Corollary 7 : If $X = A_d$, then A has at most two nonzero eigenvalues.

Proof : With $X = A_d$, (3.19) holds for any nonzero eigenvalue of A. But this equation is a quadratic in λ. □

Corollary 8 : Given any square matrix $A \neq 0$ with minimum polynomial $m(\lambda) = a_3\lambda^3 + a_2\lambda^2 + a_1\lambda$ where $a_i \neq 0$, $i=1,2,3$, there exist scalars α and β such that $X = A_d$.

Proof : If λ_1 and λ_2 denote the nonzero roots of $m(\lambda) = 0$, then

$$A^3 - (\lambda_1 + \lambda_2)A^2 + \lambda_1\lambda_2 A = 0. \tag{3.20}$$

can be written as in Theorem 6(vi) with

$$\alpha = (\lambda_1 + \lambda_2)/(\lambda_1\lambda_2), \quad \beta = \lambda_1 + \lambda_2.$$

Hence $X = A_d$, by Theorem 6(i). □

It is next observed that (2.14) is a special case of (3.7) obtained by taking $\alpha = \beta = 2$ and $X = A_d$. Hence any general result that can be established for arbitrary matrices A in Theorem 6 holds for generalized pseudo-involutions. In particular, the eigenvalue-eigenvector structure now to be considered is a direct extension of the discussion in [3].

For any nonzero matrix A in a field of characteristic $p \neq 2$ such that A and A_d satisfy (3.7) for some α and β,

$$m(\lambda) = \alpha\lambda^3 - \alpha\beta\lambda^2 + \beta\lambda \tag{3.21}$$

with roots

$$\lambda = 0, \ \lambda = [\alpha\beta \pm (\alpha^2\beta^2 - 4\alpha\beta)^{\frac{1}{2}}]/2\alpha, \tag{3.22}$$

where the last two roots are nonzero since $\alpha\beta \neq 0$. Designating the nonzero roots of (3.22) as λ_1 and λ_2, then A has eigenvalues $\lambda = 0, \lambda_1, \lambda_2$. and if A^{-1} does not exist and $A \neq \lambda_i I$ and $A^2 \neq \lambda_i I$, $m(\lambda)$ is the minimal polynomial for A. Now, rewriting (3.20) in the form

$$A(A - \lambda_1 I)(A - \lambda_2 I) = 0, \tag{3.23}$$

the eigenvector structure of A can be deduced easily.

Observe first that the eigenvalues λ_1 and λ_2 are real if $\alpha\beta \geq 4$ and distinct if $\alpha\beta \neq 4$, and if A is singular, nonzero columns of the matrix

$(A - \lambda_1 I)(A - \lambda_2 I)$ are clearly eigenvectors of A corresponding to eigenvalue $\lambda = 0$. Moreover, rewriting (3.23) as

$$A[A(A - \lambda_2 I)] = \lambda_1 [A(A - \lambda_2 I)] \tag{3.24}$$

and as

$$A[A(A - \lambda_1 I)] = \lambda_2 [A(A - \lambda_1 I)] \tag{3.25}$$

implies that nonzero columns of $A(A - \lambda_2 I)$ are eigenvectors of A corresponding to eigenvalue λ_1 and, dually, nonzero columns of $A(A - \lambda_1 I)$ are eigenvectors of A corresponding to eigenvalue λ_2. Furthermore, with λ_1 and λ_2 distinct, a complete set of eigenvectors for A can be formed using columns of $(A - \lambda_1 I)(A - \lambda_2 I)$, $A(A - \lambda_1 I)$ and $A(A - \lambda_2 I)$. To establish this fact, it is sufficient to show that the partitioned matrix

$$[(A - \lambda_1 I)(A - \lambda_2 I), A(A - \lambda_1 I), A(A - \lambda_2 I)] \tag{3.27}$$

has a right inverse, and thus full row rank. Now with $\lambda_1 + \lambda_2 = \beta$ and $\lambda_1 \lambda_2 = \alpha^{-1}\beta$, the first submatrix in (3.27) can be written in the form

$$(A - \lambda_1 I)(A - \lambda_2 I) = A^2 - \beta A + \alpha^{-1}\beta I. \tag{3.28}$$

Also, with $\lambda_1 \neq \lambda_2$, the system of equations

$$\begin{bmatrix} 1 & 1 \\ \lambda_1 & \lambda_2 \end{bmatrix} \begin{bmatrix} \gamma \\ \delta \end{bmatrix} = \begin{bmatrix} 1 \\ \lambda_1 + \lambda_2 \end{bmatrix} \tag{3.29}$$

has a unique solution. Whereupon, substituting the expression from (3.28) into (3.27) and using γ and δ from (3.29), it can then be verified that the partitioned matrix

$$(3.30)\qquad \begin{bmatrix} \alpha\beta^{-1}I \\ -\alpha\beta^{-1}\gamma I \\ -\alpha\beta^{-1}\delta I \end{bmatrix}$$

is a right inverse of the matrix in (3.27).

An analogous technique for constructing a complete set of eigenvectors for A when $\alpha\beta = 4$ and $\lambda_1 = \lambda_2$ is only slightly more complicated. Indeed, with $\lambda_1 = \lambda_2$, columns of the matrix $(A - \lambda_1 I)^2$ are vectors in the null space of A, and with $A[A(A - \lambda_1 I)] = \lambda_1[A(A - \lambda_1 I)]$ from (3.24), nonzero columns of $A(A - \lambda_1 I)$ are eigenvectors of A corresponding to eigenvalue λ_1. Observe, moreover, that with

$$(3.31)\qquad AY = \lambda_1 Y + A(A - \lambda_1 I),$$

for some matrix Y, then any nonzero column of Y corresponding to a nonzero column of $A(A - \lambda_1 I)$ is a generalized eigenvector of A. Rewriting (3.31) in the form

$$(3.32)\qquad (A - \lambda_1 I)Y = A(A - \lambda_1 I),$$

however, this expression holds trivially if Y = A. Corresponding to the above argument for obtaining a complete set of eigenvectors when $\lambda_1 \neq \lambda_2$, it is next observed that the partitioned matrix

$$(3.33)\qquad [(A - \lambda_1 I)^2, A(A - \lambda_1 I), A]$$

has full row rank since it can be verified that

$$(3.34) \qquad \begin{bmatrix} \lambda_1^{-2} I \\ -\lambda_1^{-2} I \\ \lambda_1^{-1} I \end{bmatrix}$$

is a right inverse of the matrix in (3.33). Consequently, when $\lambda_1 = \lambda_2$, a complete set of linearly independent eigenvectors and associated generalized eigenvectors can be formed using the columns of the matrix in (3.33).

As will be discussed in Section 4 where computational techniques are considered, there are special cases in which a complete set of eigenvectors can be selected directly from columns of the matrix in (3.33) when $\alpha\beta=4$, but in other cases it is necessary to solve an auxiliary system of equations to obtain a complete set.

To conclude this discussion, it is noted that if A is nonsingular, then the expression in Theorem 6(vi) reduces to $\alpha A^2 - \alpha\beta A + \beta I = 0$, and the first submatrix in both (3.27) and (3.33) must be deleted. Also, (3.27) reduces to $[A - \lambda_1 I, A - \lambda_2 I]$ and (3.33) becomes $[A - \lambda_1 I, I]$, in which case the fact that these matrices have full row rank follows immediately.

4. COMPUTATIONAL TECHNIQUES

The computational techniques to be considered in this section include general procedures for constructing n by n matrices A satisfying Theorem 6(vi) for any scalars α and β, and the related question of obtaining complete sets of eigenvectors and associated generalized eigenvectors for such matrices, when $\alpha\beta=4$, from columns of the partitioned matrix in (3.33). A basic tool to be used in this discussion is the full rank factorization of arbitrary nonzero matrices as products $A = EG$.

Given an arbitrary n by n matrix A with rank r, $0 < r < n$, and any full rank factorization $A = EG$, it follows that if $\alpha A^3 - \alpha\beta A^2 + \beta A = 0$ for some nonzero scalars α and β as in Theorem 6(vi), then

$$\alpha(GE)^2 - \alpha\beta(GE) + \beta I = 0. \tag{4.1}$$

This implies, in particular, that GE is nonsingular, as is required since A and A^2 must have the same rank. Conversely, given any n by r matrix E and r by n matrix G such that (4.1) holds, then EG is a full rank factorization, since GE is nonsingular, and $A = EG$ is an n by n matrix satisfying Theorem 6(vi). Thus (4.1) is both a necessary and sufficient condition that $A = EG$ satisfies Theorem 6. This equivalence, noted in [3, Chapter 4] for generalized pseudo-involutions when $\alpha = \beta = 2$, can now be combined with direct extensions of representations for pseudo-involutions to construct such matrices A.

With GE nonsingular, (4.1) holds if and only if

$$\alpha GE + \beta(GE)^{-1} = \alpha\beta I, \tag{4.2}$$

which in the special case $\alpha=\beta=2$ gives $(GE) + (GE)^{-1} = 2I$, so that GE is a pseudo-involution. It was shown in [3, Theorem 3], moreover, that any matrix H is a pseudo-involution if and only if H can be written in the form

(4.3) $H = I - PQ$

for some matrices P and Q with QP=0, and

(4.4) $H^{-1} = I + PQ,$

where PQ = 0 is a trivial special case. To generalize (4.3) and (4.4) to obtain solutions to (4.2), suppose first that GE has the form

(4.5) $GE = \gamma I - \delta PQ$

where γ and δ are some nonzero scalars and again QP=0 but PQ≠0. Then using (4.2) with

(4.6) $(GE)^{-1} = \alpha I - \alpha\beta^{-1}GE = \alpha(1 - \beta^{-1}\gamma)I + \alpha\beta^{-1}\delta PQ$

and $(PQ)^2=0$, $GE(GE)^{-1}=I$ implies

(4.7) $\alpha\gamma(1 - \beta^{-1}\gamma) = 1$

and

(4.8) $\alpha\beta^{-1}\gamma\delta - \alpha\delta(1 - \beta^{-1}\gamma) = 0.$

Hence, with α and δ nonzero, (4.8), in any field of characteristic p≠2, gives

(4.9) $\gamma = 2^{-1}\beta,$

and to have γ also satisfy (4.7), then $\alpha\beta=4$. Thus, substituting (4.9) into (4.5), it follows for an arbitrary choice of $\delta\neq 0$ that any matrices E and G with

(4.10) $GE = 2^{-1}\beta I - \delta PQ$

will yield a matrix A = EG satisfying Theorem 6(vi) provided only that $\alpha\beta=4$.

To next consider a method for constructing matrices E and G to satisfy (4.10), let R be any r by r nonsingular matrix, and let P and Q be any nonzero matrices, r by n-r and n-r by r, respectively, with QP=0. Then the partioned matrices E and G defined by

$$(4.11) \qquad E = \begin{bmatrix} R \\ Q \end{bmatrix}, \quad G = [\ 2^{-1}\beta R^{-1}\ ,\ -\delta P\ ]$$

clearly satisfy (4.10), and for this choice of E and G, the full rank factorization A = EG has the form

$$(4.12) \qquad A = \begin{bmatrix} 2^{-1}\beta I & -\delta RP \\ 2^{-1}\beta QR^{-1} & 0 \end{bmatrix}.$$

It will next be shown that this technique for constructing matrices A to satisfy Theorem 6(iv) can be modified slightly to obtain a corresponding form when $\alpha\beta \neq 4$.

Given any matrices E and G with $GE = \gamma I - \delta PQ$ written as in (4.5), if the condition QP = 0 is replaced by QP = I, then $(PQ)^2 = PQ$ is idempotent, and forming $GE(GE)^{-1} = I$ as before, (4.7) again holds, but (4.8) now becomes

$$\alpha\beta^{-1}\gamma\delta - \alpha\delta(1 - \beta^{-1}\gamma) - \alpha\beta^{-1}\delta^2 = 0.$$

Therefore, with α and δ nonzero scalars, it follows in any field of characteristic $p \neq 2$ that

$$\delta = 2\gamma - \beta.$$

Hence δ is no longer arbitrary. Observe, moreover, that the equation in (4.7) is independent of δ, and can be written in the form $\alpha\gamma^2 - \alpha\beta\gamma + \beta = 0$ to correspond to the equation in (3.19). Consequently, with $\gamma = \lambda_i$, i=1,2, where λ_1 and λ_2 are the nonzero roots in (3.22), then $\delta = 2\lambda_i - \beta$.

(Note that $\gamma = \lambda_i$ reduces to (4.9) when $\alpha\beta = 4$.) Finally, using $\gamma = \lambda_i$ and $\delta = 2\lambda_i - \beta$ to form

$$E = \begin{bmatrix} R \\ Q \end{bmatrix}, \quad G = \begin{bmatrix} \lambda_i R^{-1} & , -\delta P \end{bmatrix},$$

analogous to (4.11), (4.5) holds, and with $QP = I$, $A = EG$ becomes

$$A = \begin{bmatrix} \lambda_i I & -\delta RP \\ \lambda_i QR^{-1} & -\delta I \end{bmatrix}.$$

This representation and (4.12) can then be used to construct n by n matrices A satisfying Theorem 6(vi) for any nonzero scalars α and β.

Before proceeding to the discussion of a method for obtaining complete sets of eigenvectors and associated generalized eigenvectors for matrices A satisfying Theorem 6(vi) with $\alpha\beta = 4$, it is noted that (4.10), (4.11) and (4.12) also hold if $\delta = 0$. In this case, moreover, with $GE = 2^{-1}\beta I$ in (4.10), $A^2 = 2^{-1}\beta A$, and the partitioned form for A in (4.12) also contains a null submatrix in the upper right hand corner. Furthermore, it is observed that if A satisfies Theorem 6(vi) for any nonzero scalars α and β, then the same expression holds for any matrix $B = S^{-1}AS$ similar to A. Thus, for computational purposes when $\alpha\beta = 4$, matrices in (4.12) can be made to appear more general, that is, eliminating the null submatrix in the lower right hand corner, by replacing A by B using any nonsingular matrix S. This initial preprocessing technique for constructing

numerical matrices A which do not contain a null submatrix is used for small examples in Section 5 and for larger examples in Appendix C.

It was shown in Section 3 that the partitioned matrix in (3.27) always contains a complete set of eigenvectors of A when $\alpha\beta \neq 4$, and such a set of vectors can be obtained using any technique for selecting a maximal linearly independent subset of columns. It will next be shown that a corresponding technique when $\alpha\beta = 4$ can be made more explicit in terms of which columns may be used from the matrix in (3.33).

The procedure now to be described for constructing eigenvectors for matrices A when $\alpha\beta = 4$ is a direct extension of the discussion in [3, Chapter 4] for generalized pseudo-involutions with $\alpha = \beta = 2$. As was shown in Section 3, if A^{-1} does not exist, then the partitioned matrix in (3.33) has full row rank, and columns of this matrix can be used to form a complete set of eigenvectors and associated generalized eigenvectors for A. To construct such a set of eigenvectors, suppose that A has rank r, $A(A-\lambda_1 I)$ has rank s and $(A-\lambda_1 I)^2$ has rank t, where $0 < s \leq r$ and $t > 0$. Next, let U denote any set of s linearly independent columns of $A(A-\lambda_1 I)$, let V denote the corresponding columns of A, and let W denote any set of t linearly independent columns of $(A-\lambda_1 I)^2$. Then it follows that the columns of the partitioned matrix [U, V, W] are linearly independent. For if

(4.13) $\quad Ux + Vy + Wz = 0,$

for some vectors x and y with s components and vector z with t components, multiplying on the left by $(A - \lambda_1 I)$ gives

(4.14) $\quad Uy - \lambda_1 Wz = 0.$

Next, multiplying (4.14) by A on the left yields

(4.15) $\lambda_1 Uy = 0$

and thus $y = 0$, as $\lambda_1 \neq 0$. Whence (4.14) reduces to $-\lambda_1 Wz = 0$, which implies $z = 0$, so $x = 0$, as (4.13) now becomes $Ux = 0$.

If A is an n by n matrix, then $2s + t \leq n$. Thus if $2s + t = n$, columns of [U, V, W] form a complete set of eigenvectors and associated generalized eigenvectors of A. As will next be shown, if $2s + t < n$, the remaining eigenvectors can be constructed using solutions of a homogeneous system of equations.

Assume that $2s + t < n$ and let $A(A - \lambda_1 I) = UY$ be a full rank factorization. For any left inverse, L, of U, let B be the partitioned matrix of the form

(4.16) $$B = \begin{bmatrix} Y \\ L \\ (A-\lambda_1 I)^2 \end{bmatrix}.$$

Now Y and L have rank s and $(A-\lambda_1 I)^2$ has rank t, so that $\text{rank}(B) \leq 2s + t$, and the dimension of the nullspace of B is at least $n - 2s - t$. Then the required eigenvectors are obtained as columns of any n by n-2s-t matrix T with full column rank such that

(4.17) $BT = 0.$

Indeed, since $YT = 0$, then $A(A-\lambda_1 I)T = 0$, and writing $(A - \lambda_1 I)^2 T = 0$ as $[A(A - \lambda_1 I) - \lambda_1(A - \lambda_1 I)]T = 0$, it follows that $AT = \lambda_1 T$. Consequently, it only remains to be shown that columns of the matrix [U, V, W, T] are linearly independent. From the choice of W and U, $A(A-\lambda_1 I)^2 = 0$ implies $AW = 0$, and $A[A(A-\lambda_1 I)] = \lambda_1 A(A-\lambda_1 I)$ so that $AU = \lambda_1 U$. Also, with columns

of V generalized eigenvectors of A associated with eigenvectors provided by columns of U, $AV = \lambda_1 V + U$. Thus if

(4.18) $\quad Ux + Vy + Wz + T\zeta = 0,$

corresponding to (4.13) where ζ is a vector with n-2s-t components, multiplying by A gives

(4.19) $\quad \lambda_1 Ux + \lambda_1 Vy + Uy + \lambda_1 T\zeta = 0.$

Next, multiplying (4.19) by $A-\lambda_1 I$,

$$\lambda_1 Uy = 0.$$

Therefore $y = 0$, and (4.19) becomes

(4.20) $\quad \lambda_1 Ux + \lambda_1 T\zeta = 0,$

which, when multiplied by L on the left yields $\lambda_1 x = 0$, so $x = 0$. Hence $\zeta = 0$, by (4.20), reducing (4.18) to $Wz = 0$ which in turn implies $z = 0$.

The importance of these observations on the eigenvector structure of matrices A satisfying Theorem 6(iv) with $\alpha\beta = 4$ is that having determined linearly independent columns of $A(A - \lambda_1 I)$ and $(A - \lambda_1 I)^2$ to form U and W, columns of A to form V are specified, and if $2s + t < n$, the remaining eigenvectors can be obtained using (4.17). It is finally noted that if A is nonsingular, then the arguments to show linear independence in (4.13) and (4.18) remain valid with Wz omitted and the steps to conclude that $z = 0$ deleted.

5. NUMERICAL EXAMPLES

The purpose of this section is to provide examples of matrices satisfying Theorem 6(vi) for the cases $\alpha\beta = 4$ and $\alpha\beta \neq 4$. As discussed in Section 4, if a matrix A satisfies Theorem 6(vi) then so does any matrix $B = S^{-1}AS$, and to eliminate the null submatrix in (4.12) it is useful to generate an arbitrary nonsingular n by n matrix S and its inverse. Moreover, to construct nonsingular matrices S, it is sufficient to generate a row vector q and a column vector p, both of size n, such that $qp=0$. Then $S = I + pq$ is a pseudo-involution and $S^{-1} = I - pq$. (All arbitrary nonsingular matrices used in the program in Appendix B to create matrices A to satisfy Theorem 6(vi) are generated in this manner.)

Let us first consider the real 4 by 4 matrix A with $\alpha = 4/3$ and $\beta = 3$, defined by

$$(5.1) \qquad A = \begin{bmatrix} 1.78 & -0.22 & -0.06 & 0.00 \\ 0.04 & -0.23 & -1.61 & -0.04 \\ -1.11 & 2.37 & 3.17 & 0.04 \\ 19.94 & -81.83 & -66.83 & -0.22 \end{bmatrix}.$$

It is easily checked that A satisfies $\alpha A^3 - \alpha\beta A^2 + \beta A = 0$ to 2 decimal places. Also, employing the techniques in Section 4, the following eigenvectors of A can be obtained where x_0 denotes an eigenvector corresponding to eigenvalue $\lambda = 0$, x_1 denotes an eigenvector of A corresponding to eigenvalue $\lambda = \beta/2 = 1.5$, and x_2 and x_3 denote an eigenvector and an associated generalized eigenvector of A corresponding to eigenvalue $\lambda = 1.5$, respectively.

$$(5.2) \qquad x_0 = \begin{bmatrix} 0.34 \\ 2.86 \\ -3.84 \\ 133.82 \end{bmatrix}, \; x_1 = \begin{bmatrix} 0.53 \\ 1.00 \\ -1.07 \\ 0.00 \end{bmatrix},$$

$$x_2 = \begin{bmatrix} 0.53 \\ 0.96 \\ -2.88 \\ 72.30 \end{bmatrix}, \; x_3 = \begin{bmatrix} 1.78 \\ 0.04 \\ -1.11 \\ 19.94 \end{bmatrix}.$$

A larger example with $\alpha\beta = 4$ is provided in Appendix C.

It should be noted that if $\alpha\beta = 4$ and β is an even integer, an integral matrix satisfying Theorem 6(vi) is easily obtained from (4.12) where the technique outlined above for forming S can again be used to generate a nonsingular integral matrix R as long as the elements of q and p are chosen as integers. If elements of A are to be chosen from a finite field of integers, the only stipulation on the value of β is that it must be an integer. Also, since the elements of all the eigenvectors obtained are rational numbers, they can easily be made into integers by simply multiplying each vector by the greatest common denominator of its elements. The following is an example of an integral matrix A satisfying Theorem 6(vi) with $\alpha = 1$ and $\beta = 4$

$$(5.3) \qquad A = \begin{bmatrix} 6 & 0 & 26 & 9 \\ -8 & 2 & -44 & -16 \\ 4 & 0 & 16 & 6 \\ -12 & 0 & -50 & -18 \end{bmatrix}.$$

The nonzero eigenvalue of this matrix is $\lambda = 2$, and with eigenvectors labelled as in the previous example,

$$(5.4) \qquad x_0 = \begin{bmatrix} 3 \\ -4 \\ 0 \\ -2 \end{bmatrix}, \; x_1 = \begin{bmatrix} -5 \\ 0 \\ -2 \\ 8 \end{bmatrix},$$

$$x_2 = \begin{bmatrix} 35 \\ -56 \\ 14 \\ -56 \end{bmatrix}, \; x_3 = \begin{bmatrix} 13 \\ -22 \\ 8 \\ -25 \end{bmatrix}.$$

If $\alpha\beta \neq 4$, using the analogous form of (4.12) for this case, it is easy to generate another matrix to satisfy Theorem 6(vi). The technique to generate q and p for this form is similar to the one outlined above for $qp = 0$, except that the n^{th} elements are now chosen so that $qp = 1$. Given $\alpha = 5$ and $\beta = 2$, one can again easily verify that the matrix

$$(5.5) \qquad A = \begin{bmatrix} 3.37 & -1.86 & 0.87 & -0.14 \\ -2.97 & 3.65 & -4.03 & 0.29 \\ -0.32 & -4.34 & -5.51 & 0.09 \\ 10.05 & -118.24 & -155.05 & 2.26 \end{bmatrix}.$$

satisfies Theorem 6(vi) and that

$$(5.6) \qquad x_0 = \begin{bmatrix} 8.83 \\ -10.74 \\ 15.15 \\ 437.63 \end{bmatrix}$$

is an eigenvector of A corresponding to eigenvalue zero. Also,

$$(5.7) \qquad x_1 = \begin{bmatrix} 26.40 \\ -32.80 \\ 43.32 \\ 1264.05 \end{bmatrix}$$

is an eigenvector of A corresponding to eigenvalue $\lambda_1 = 1.7746$. Finally, the last two eigenvectors, corresponding to eigenvalue $\lambda_2 = 0.2254$, are

$$(5.8) \qquad x_2 = \begin{bmatrix} 14.41 \\ 27.75 \\ 14.41 \\ 455.46 \end{bmatrix}, \; x_3 = \begin{bmatrix} 0.56 \\ -39.05 \\ -16.02 \\ -18.02 \end{bmatrix}.$$

A larger example for the case $\alpha\beta \neq 4$ is illustrated in Appendix C.

(The program used to generate the real examples is included in Appendix B, and was run on the DEC System 10 at the University of Tennessee. The execution time for each of these examples was approximately 0.20 seconds.)

6. AREAS FOR FURTHER RESEARCH

Although the purpose of this thesis was to obtain an extension of generalized pseudo-involutions, some of the results may provide a possible starting point for further research. As an example, given a complex matrix A as in Theorem 2, choosing α so that the Euclidean norm of αA is less than 1, then $(\alpha A - I)^{-1}$ can be expanded in a converging power series [6], and it seems of interest to consider relationships between A and approximations to X constructed by truncating this series. Other areas of interest include the possible relationships that may be obtained by equating the expressions for X in (3.8) and (3.11), without assuming A has index $k = 1$ as in Corollary 5, and the question of whether the methods for forming complete sets of eigenvectors when A satisfies Theorem 6(vi) are applicable to matrices with more general minimal polynomials.

LIST OF REFERENCES

LIST OF REFERENCES

1. Ben-Israel, A. and Greville, T.N.E. Generalized Inverses: Theory and Applications. New York: Wiley and Sons, 1974.

2. Cline, Randall E. Elements of the Theory of Generalized Inverses for Matrices. Newton, Mass.: Educational Development Center, 1979.

3. Cline, R.E. and McConnel, R.M., Lectures on Involutory Matrices, In Preparation.

4. Drazin, M.P. "Pseudo-inverses in associative rings and semi-groups." Amer. Math. Monthly 65 (1958), 506-513.

5. Perlis, Sam. Theory of Matrices. Cambridge, Mass.: Addison-Wesley Press, 1952.

6. Varga, Richard S. Matrix Iterative Analysis. Englewood Cliffs, New Jersey: Prentice Hall, 1962.

APPENDICES

APPENDIX A

EXPLANATIONS OF SUBROUTINES

EXPLANATIONS OF SUBROUTINES

The following is an alphabetically ordered listing of the subroutines used to generate some of the examples in Section 5 and Appendix C. The subroutine FRFAC is included here because it is an often helpful tool, although it was not needed for any of the examples created in this thesis.

The subroutine FRFAC is used to obtain a full rank factorization EG for any matrix A. The method used here is trapezoidal decomposition which is outlined in [2]. The subroutine PERMUT is called by this routine.

The subroutine LINVER basically obtains a left inverse of a matrix with full column rank. The routine does not check whether the inverse exists as the applications do not call for it here, but it can easily be modified to include such a check by testing the pivot element obtained in the routine. Obviously the routine can also be used to generate the inverse of a nonsingular matrix.

The subroutine MATPR is used to print out matrices and a message.

The subroutine NSPACE obtains a maximal linearly independent set of vectors in the nullspace of a matrix. This task is carried out by first row-reducing a matrix using ROWRED and then constructing the vectors by setting the arbitrary variables to a combination of ones and zeros to insure linear independence.

The subroutine PERMUT is called by the subroutine FRFAC to find possible permutations in rows and columns of E and G that would allow the full rank factorization to continue. This routine utilizes the RMULT and RSUB subroutines to carry out its task.

The subroutine RADD adds two real valued matrices together and returns the resulting matrix.

The subroutine RMULT multiplies two real matrices and returns the resulting product.

The subroutine ROWRED is designed to reduce a matrix to row-echelon form. This process is carried out by using partial pivoting and keeping track of what swaps have taken place. The resulting matrix in row-echelon form can be used to obtain a maximal linearly independent set of columns of the original matrix or to construct vectors in the nullspace of the matrix.

The subroutine RSUB simply subtracts two real matrices from each other and returns the result.

The following appendix includes listings of all of these subroutines and the drivers used to create specific examples.

APPENDIX B

COMPUTER PROGRAMS

```
C       ************************************************************************
C       *                                                                      *
C       *                          F R F A C                                   *
C       *                                                                      *
C       ************************************************************************
C       *                                                                      *
C       * PURPOSE: THE PURPOSE OF THIS ROUTINE IS TO OBTAIN A FULL RANK        *
C       *          FACTORIZATION OF A MATRIX.                                  *
C       *                                                                      *
C       ************************************************************************
C       *                                                                      *
C       * VARIABLE CONVENTION:                                                 *
C       *   A  -  ORIGINAL AS BY AS MATRIX STORED IN AD1 BY AD2 SPACE.         *
C       *   AR -  REAL VERSION OF A.                                           *
C       *   E,G-  A=E*G.                                                       *
C       *   INDEX-RANK OF MATRIX A.                                            *
C       *   RPERM,LPERM-MATRICES THAT KEEP TRACK OF RIGHT AND LEFT             *
C       *               PERMUTATONS.                                           *
C       *   RIND,LIND-KEEP TRACK OF HOW MANY PERMUTATIONS ARE TO BE PER-       *
C       *             FORMED ON EACH SIDE.                                     *
C       *   SUM,TEMP-DUMMY UTILITY VARIABLES.                                  *
C       *                                                                      *
C       ************************************************************************
C
C
        SUBROUTINE FRFAC(A,AD1,AD2,AS,E,G,INDEX)
        LOGICAL CHECK
        INTEGER AD1,AD2,AS,RIND,LIND,RPERM(10,2),LPERM(10,2)
        DOUBLE PRECISION A(AD1,AD2)
        DOUBLE PRECISION AR(10,10),E(AD1,AD2),G(AD1,AD2),SUM,TEMP
C
        INDEX=1
        RIND=0
        LIND=0
        DO 610 I=1,AS
           DO 610 J=1,AS
              AR(I,J)=A(I,J)
              E(I,J)=0.0D0
              G(I,J)=0.0D0
 610    CONTINUE
        CHECK=.TRUE.
        IF (DABS(A(1,1)).LT.1D-4) CALL PERMUT(AR,AD1,AD2,AS,E,G,INDEX,
     *                                RPERM,RIND,LPERM,LIND,CHECK)
        IF (.NOT.CHECK) THEN
           INDEX=0
           RETURN
        ENDIF
C
        E(1,1)=AR(1,1)
        G(1,1)=1.0D0
        DO 630 I=2,AS
```

```
            E(I,1)=AR(I,1)
            G(1,I)=AR(1,I)/E(1,1)
 630     CONTINUE
         INDEX=2
C
         DO WHILE ((INDEX.LE.AS).AND.CHECK)
            CALL PERMUT(AR,AD1,AD2,AS,E,G,INDEX,RPERM,RIND,LPERM,LIND,
     *                  CHECK)
            IF (CHECK) THEN
               G(INDEX,INDEX)=1.0D0
               INDM1=INDEX-1
               INDP1=INDEX+1
               DO 650 I=INDEX,AS
                  SUM=0.0D0
                  DO 643 J=1,INDM1
                     SUM=SUM+E(I,J)*G(J,INDEX)
 643              CONTINUE
                  E(I,INDEX)=AR(I,INDEX)-SUM
                  SUM=0.0D0
                  DO 647 J=1,INDM1
                     SUM=SUM+E(INDEX,J)*G(J,I)
 647              CONTINUE
                  G(INDEX,I)=(AR(INDEX,I)-SUM)/E(INDEX,INDEX)
 650           CONTINUE
               INDEX=INDEX+1
            ENDIF
         ENDDO
C
         IF (LIND.NE.0) THEN
            DO 665 I=1,LIND
               J=LIND-I+1
               DO 660 K=1,AS
                  TEMP=E(LPERM(J,1),K)
                  E(LPERM(J,1),K)=E(LPERM(J,2),K)
                  E(LPERM(J,2),K)=TEMP
 660           CONTINUE
 665        CONTINUE
         ENDIF
         IF (RIND.NE.0) THEN
            DO 675 I=1,RIND
               J=RIND-I+1
               DO 670 K=1,AS
                  TEMP=G(K,RPERM(J,1))
                  G(K,RPERM(J,1))=G(K,RPERM(J,2))
                  G(K,RPERM(J,2))=TEMP
 670           CONTINUE
 675        CONTINUE
         ENDIF
         INDEX=INDEX-1
         RETURN
         END
```

```
C       ************************************************************************
C       *                                                                      *
C       *                          L I N V E R                                 *
C       *                                                                      *
C       ************************************************************************
C       *                                                                      *
C       * PURPOSE: THE PURPOSE OF THIS SUBROUTINE IS TO OBTAIN THE LEFT        *
C       *          INVERSE OF A MATRIX WITH FULL COLUMN RANK.                  *
C       *                                                                      *
C       ************************************************************************
C       *                                                                      *
C       * VARIABLES USED:                                                      *
C       *   MAT    - ORIGINAL MATRIX WHOSE INVERSE IS TO BE OBTAINED.          *
C       *   INVER  - THE LEFT INVERSE OF MAT.                                  *
C       *   WORK   - SPACE USED TO OBTAIN THE INVERSE IN.                      *
C       *   DUMMY  - DUMMY VARIABLE USED TO SWAP COLUMNS.                      *
C       *   MAX    - VARIABLE USED TO FIND MAXIMAL ELEMENT TO PIVOT ON.        *
C       *   M,N    - DIMENSIONS OF MAT.                                        *
C       *   MAXCOL - COLUMN TO PIVOT ON.                                       *
C       *   COLUMN - NUMBER OF COLUMNS IN WORK.                                *
C       *   TRACK  - ARRAY TO KEEP TRACK OF COLUMN SWAPS.                      *
C       *   CHECK  - DUMMY VARIABLE USED TO ACTUALLY CONSTRUCT INVERSE.        *
C       *                                                                      *
C       ************************************************************************
C
        SUBROUTINE LINVER (MAT,M,N,INVER)
        DOUBLE PRECISION MAT(10,10),INVER(10,10),WORK(10,20),DUMMY,MAX
        INTEGER M,N,ROW,MAXCOL,COLUMN,TRACK(10),CHECK
C
C       INITIALIZE WORK AREA WITH TRANSPOSED MATRIX AND APPENDED IDENTITY
C       MATRIX.
        DO 410 I = 1,M
          DO 410 J = 1,N
            WORK(J,I) = MAT(I,J)
 410    CONTINUE
        DO 420 I = 1,N
          DO 415 J = 1,N
            WORK(I,J+M) = 0.0D0
 415      CONTINUE
          WORK(I,I+M) = 1.0D0
 420    CONTINUE
C
C       INITIALIZE VARIABLES AS NEEDED.
        COLUMN = M+N
        DO 425 I = 1,M
          TRACK(I) = I
 425    CONTINUE
C
C       USE PARTIAL PIVOTING TO REDUCE THE TABLE TO DESIRED FORM
        DO 460 I = 1,N
          ROW =I
```

```
          MAXCOL = I
          MAX = 0.0D0
C
C         LOCATE PIVOT ELEMENT.
          DO 430 J = ROW,M
            IF (DABS(WORK(ROW,J)).GT.MAX) THEN
              MAX = DABS(WORK(ROW,J))
              MAXCOL = J
            ENDIF
 430      CONTINUE
C
C         SWAP COLUMNS IF NECESSARY
          IF (MAXCOL.NE.ROW) THEN
            J = TRACK(ROW)
            TRACK(ROW) = TRACK(MAXCOL)
            TRACK(MAXCOL) = J
            DO 440 J = 1,N
              DUMMY = WORK(J,ROW)
              WORK(J,ROW) = WORK(J,MAXCOL)
              WORK(J,MAXCOL) = DUMMY
 440        CONTINUE
          ENDIF
C
C         UPDATE WORKTABLE BY REDUCTION.
          DO 450 J = ROW,COLUMN
            J1 = COLUMN-J+ROW
            WORK(ROW,J1) = WORK(ROW,J1)/WORK(ROW,ROW)
            DO 450 K = 1,N
              IF (K.NE.ROW) THEN
                WORK(K,J1)=WORK(K,J1)-WORK(K,ROW)*WORK(ROW,J1)
              ENDIF
 450      CONTINUE
C
 460    CONTINUE
C
C       CONSTRUCT INVERSE.
        DO 470 I = 1,N
          DO 470 J = 1,M
            CHECK = 1
            DO 465 K = 1,N
              IF (TRACK(K).EQ.J) THEN
                CHECK = 0
                INVER(I,J) = WORK(K,M+I)
              ENDIF
 465        CONTINUE
          IF (CHECK.EQ.1) THEN
            INVER(I,J) = 0.0D0
          ENDIF
 470    CONTINUE
        RETURN
        END
```

```
C     ***************************************************************************
C     *                                                                         *
C     *                          M A T P R                                      *
C     *                                                                         *
C     ***************************************************************************
C     *                                                                         *
C     * PURPOSE: THE PURPOSE OF THIS ROUTINE IS TO PRINT OUT A GIVEN            *
C     *          MATRIX ALONG WITH A MESSAGE PERTAINING TO WHAT THE             *
C     *          MATRIX IS.                                                     *
C     *                                                                         *
C     ***************************************************************************
C     *                                                                         *
C     * VARIABLE CONVENTION:                                                    *
C     *   A IS THE MATRIX TO BE PRINTED.                                        *
C     *   AD1, AD2 ARE USED TO DECLARE THE ARRAY SIZE.                          *
C     *   AS1, AS2 IS THE ACTUAL SIZE OF THE MATRIX.                            *
C     *                                                                         *
C     ***************************************************************************
C
      SUBROUTINE MATPR(MESSAG,A,AD1,AD2,AS1,AS2)
      INTEGER AD1,AD2,AS1,AS2
      DOUBLE PRECISION A(AD1,AD2)
      CHARACTER*10 MESSAG
C
      WRITE(21,530) MESSAG,(A(1,J),J=1,AS2)
  530 FORMAT(' ',A10,10F10.2)
      IF (AS1.GT.1) THEN
         DO 550 I=2,AS1
            WRITE(21,540) (A(I,J),J=1,AS2)
  540       FORMAT(' ',10X,10F10.2)
  550    CONTINUE
      ENDIF
      WRITE(21,560)
  560 FORMAT(' '//)
      RETURN
      END
```

```
C     ***************************************************************************
C     *                                                                         *
C     *                          N S P A C E                                    *
C     *                                                                         *
C     ***************************************************************************
C     *                                                                         *
C     * PURPOSE: THE PURPOSE OF THIS ROUTINE IS TO OBTAIN A MAXIMAL             *
C     *          LINEARLY INDEPENDENT SET OF VECTORS IN THE NULL                *
C     *          SPACE OF A MATRIX.                                             *
C     *                                                                         *
C     ***************************************************************************
C     *                                                                         *
C     * VARIABLES USED:                                                         *
C     *   MAT     - MATRIX WHOSE NULL SPACE IS IN QUESTION.                     *
C     *   RES     - MATRIX IN WHICH RESULTING VECTORS WILL BE STORED.           *
C     *   WORK    - USED TO ROW REDUCE THE MATRIX.                              *
C     *   D1,D2   - DIMENSIONED SIZE OF MAT IN MAIN PROGRAM.                    *
C     *   M,N     - DIMENSIONS OF MAT.                                          *
C     *   TRACK   - ALSO USED IN ROW REDUCING MATRIX.                           *
C     *   RANK    - RANK OF MAT.                                                *
C     *   NUM     - DIMENSION OF NULLSPACE OF MAT.                              *
C     *                                                                         *
C     ***************************************************************************
C
      SUBROUTINE NSPACE(MAT,D1,D2,M,N,RES,NUM)
      INTEGER M,N,NUM,TRACK(10),RANK,D1,D2
      DOUBLE PRECISION MAT(D1,D2), RES(10,10), AREA(30,10)
C
C     ROW REDUCE THE MATRIX.
      CALL ROWRED(MAT,30,10,M,N,RANK,TRACK,AREA)
C
C     CONSTRUCT NECESSARY COLUMNS.
      NUM = N - RANK
      DO 280 J = 1,NUM
        DO 280 I = 1,N
          IF (I.LE.RANK) THEN
            RES(TRACK(I),J) = - AREA(I,RANK + J)
          ELSE
            IF ((I-J).EQ.RANK) THEN
              RES(TRACK(I),J) = 1.0D0
            ELSE
              RES(TRACK(I),J) = 0.0D0
            ENDIF
          ENDIF
 280  CONTINUE
C
      RETURN
      END
```

```
C     ***************************************************************************
C     *                                                                         *
C     *                          P E R M U T                                    *
C     *                                                                         *
C     ***************************************************************************
C     *                                                                         *
C     * PURPOSE: THE PURPOSE OF THIS ROUTINE IS TO AID FRFAC IN FIND-           *
C     *          ING PERMUTATIONS OF ROWS AND COLUMNS TO AVOID ZEROS            *
C     *          ON THE MAIN DIAGONALS OF THE FACTORS E AND G.                  *
C     *                                                                         *
C     ***************************************************************************
C     *                                                                         *
C     * VARIABLE CONVENTION:                                                    *
C     *   SAME CONVENTIONS AS IN FRFAC EXCEPT FOR:                              *
C     *   RES1,RESULT-DUMMY MATRICES TO MULTIPLY AND SUBTRACT WITH.             *
C     *                                                                         *
C     ***************************************************************************
C
C
      SUBROUTINE PERMUT(AR,AD1,AD2,AS,E,G,INDEX,RPERM,RIND,LPERM,
     *                  LIND,CHECK)
      LOGICAL CHECK,BOOL
      INTEGER AD1,AD2,AS,INDEX,RIND,LIND,RPERM(10,2),LPERM(10,2)
      DOUBLE PRECISION AR(10,10),E(AD1,AD2),G(AD1,AD2),
     *                 RES1(10,10),RESULT(10,10)
C
      CHECK=.FALSE.
      BOOL=.TRUE.
      CALL RMULT(E,AD1,AD2,AS,AS,G,AD1,AD2,AS,AS,RES1,10,10)
      CALL RSUB(AR,10,10,AS,AS,RES1,10,10,RESULT,10,10)
C
      DO 710 I=INDEX,AS
         DO 710 J=INDEX,AS
            IF ((DABS(RESULT(I,J)).GT.1.0D-4).AND.BOOL) THEN
               BOOL=.FALSE.
               CHECK=.TRUE.
               IF (I.GT.INDEX) THEN
                  LIND=LIND+1
                  LPERM(LIND,1)=INDEX
                  LPERM(LIND,2)=I
                  DO 705 K=1,AS
                     TEMP=E(I,K)
                     E(I,K)=E(INDEX,K)
                     E(INDEX,K)=TEMP
                     TEMP=AR(I,K)
                     AR(I,K)=AR(INDEX,K)
                     AR(INDEX,K)=TEMP
705               CONTINUE
               ENDIF
               IF (J.GT.INDEX) THEN
                  RIND=RIND+1
```

```
                        RPERM(RIND,1)=INDEX
                        RPERM(RIND,2)=J
                        DO 707 K=1,AS
                           TEMP=G(K,J)
                           G(K,J)=G(K,INDEX)
                           G(K,INDEX)=TEMP
                           TEMP=AR(K,J)
                           AR(K,J)=AR(K,INDEX)
                           AR(K,INDEX)=TEMP
 707                    CONTINUE
                     ENDIF
                  ENDIF
 710     CONTINUE
C
         RETURN
         END
```

```
C       ***************************************************************************
C       *                                                                         *
C       *                              R A D D                                    *
C       *                                                                         *
C       ***************************************************************************
C       *                                                                         *
C       * PURPOSE: THE PURPOSE OF THIS ROUTINE IS TO ADD TWO MATRICES             *
C       *          ( OF ANY SIZE, EVEN VECTORS ) AND RETURN THE RESULT            *
C       *          TO THE MAIN PROGRAM (REAL VERSION).                            *
C       *                                                                         *
C       ***************************************************************************
C       *                                                                         *
C       * VARIABLE CONVENTION:                                                    *
C       *   A AND B ARE THE MATRICES TO BE ADDED, C STORES RESULT.                *
C       *   AD1, AD2, BD1, BD2, CD1, CD2 ARE USED TO DECLARE ARRAY SIZE.          *
C       *   AS1, AS2 ARE THE ACTUAL SIZES OF THE MATRICES.                        *
C       *                                                                         *
C       ***************************************************************************
C
C
        SUBROUTINE RADD(A,AD1,AD2,AS1,AS2,B,BD1,BD2,C,CD1,CD2)
        INTEGER AD1,AD2,BD1,BD2,CD1,CD2,AS1,AS2
        DOUBLE PRECISION A(AD1,AD2),B(BD1,BD2),C(CD1,CD2)
C
        DO 510 I=1,AS1
           DO 510 J=1,AS2
              C(I,J)=A(I,J)+B(I,J)
  510   CONTINUE
        RETURN
        END
```

```
C     ***************************************************************************
C     *                                                                         *
C     *                              R M U L T                                  *
C     *                                                                         *
C     ***************************************************************************
C     *                                                                         *
C     * PURPOSE: THE PURPOSE OF THIS ROUTINE IS TO MULTIPLY TWO MAT-            *
C     *          RICES ( OF ANY SIZE, EVEN VECTORS ) AND RETURN THE             *
C     *          RESULT TO THE MAIN PROGRAM (REAL VERSION).                     *
C     *                                                                         *
C     ***************************************************************************
C     *                                                                         *
C     * VARIABLE CONVENTION:                                                    *
C     *   A AND B ARE THE MATRICES TO BE MULTIPLIED, C STORES RESULT.           *
C     *   AD1, AD2, BD1, BD2, CD1, CD2 ARE USED TO DECLARE ARRAY SIZE.          *
C     *   AS1, AS2, BS1, BS2 ARE THE ACTUAL SIZES OF THE MATRICES.              *
C     *                                                                         *
C     ***************************************************************************
C
C
      SUBROUTINE RMULT(A,AD1,AD2,AS1,AS2,B,BD1,BD2,BS1,BS2,C,CD1,CD2)
      INTEGER AD1,AD2,BD1,BD2,CD1,CD2,AS1,AS2,BS1,BS2
      DOUBLE PRECISION A(AD1,AD2),B(BD1,BD2),C(CD1,CD2)
C
      DO 500 I=1,AS1
         DO 500 J=1,BS2
            C(I,J)=0.0D0
            DO 500 K=1,BS1
               C(I,J)=C(I,J) + A(I,K)*B(K,J)
500   CONTINUE
      RETURN
      END
```

```
C     ***************************************************************************
C     *                                                                         *
C     *                          R O W R E D                                    *
C     *                                                                         *
C     ***************************************************************************
C     *                                                                         *
C     * PURPOSE: THE PURPOSE OF THIS ROUTINE IS TO OBTAIN A MAXIMAL             *
C     *          LINEARLY INDEPENDENT SET OF COLUMNS OF A MATRIX BY             *
C     *          ROW-REDUCING THE MATRIX.                                       *
C     *                                                                         *
C     ***************************************************************************
C     *                                                                         *
C     * VARIABLES USED:                                                         *
C     *   MAT      - MATRIX TO BE ROWREDUCED.                                   *
C     *   TRACK    - ARRAY CONTAINING LIST OF LINEARLY INDEPENDENT              *
C     *              COLUMNS OF MAT.                                            *
C     *   RANK     - RANK OF MAT AND THUS NUMBER OF ELEMENTS OF TRACK.          *
C     *   M,N      - DIMENSIONS OF MAT.                                         *
C     *   MAXROW   - USED TO LOCATE PIVOT ELEMENT.                              *
C     *   MAXCOL   - USED TO LOCATE PIVOT ELEMENT.                              *
C     *   MAX      - USED TO LOCATE PIVOT ELEMENT.                              *
C     *   CHECK    - DUMMY VARIABLE.                                            *
C     *   WORK     - WORKSPACE IN WHICH MAT IS REDUCED.                         *
C     *                                                                         *
C     ***************************************************************************
C
      SUBROUTINE ROWRED(MAT,D1,D2,M,N,RANK,TRACK,WORK)
      INTEGER RANK, CHECK, MAXROW, MAXCOL, D1, D2
      INTEGER TRACK(D2),ROW
      DOUBLE PRECISION MAT(D1,D2), WORK(D1,D2), MAX
C
C     INITIALIZE WORK AREA.
      DO 320 J = 1,N
        DO 310 I = 1,M
          WORK(I,J) = MAT(I,J)
 310    CONTINUE
        TRACK(J) = J
 320  CONTINUE
C
C     INITIALIZE OTHER VARIABLES.
      ROW = 0
      CHECK = 0
C
C     LOOP UNTIL ALL DESIRED COLUMNS ARE OBTAINED.
      DO WHILE ((CHECK.EQ.0).AND.((ROW.LT.M).AND.(ROW.LT.N)))
        MAX = 0.0D0
        ROW = ROW + 1
        MAXROW = ROW
        MAXCOL = ROW
C
C       LOCATE PIVOT ELEMENT.
```

```
            DO 330 I = ROW,M
              DO 330 J = ROW,N
                IF (DABS(WORK(I,J)).GT.MAX) THEN
                  MAXROW = I
                  MAXCOL = J
                  MAX = DABS(WORK(I,J))
                ENDIF
 330        CONTINUE
C
C           SWAP ROWS IF NECESSARY
            IF (MAXROW.NE.ROW) THEN
              DO 340 J = 1,N
                DUMMY = WORK(ROW,J)
                WORK(ROW,J) = WORK(MAXROW,J)
                WORK(MAXROW,J) = DUMMY
 340          CONTINUE
            ENDIF
C
C           SWAP COLUMNS IF NECESSARY AND KEEP TRACK OF COLUMN SWITCH.
            IF (MAXCOL.NE.ROW) THEN
              J = TRACK(ROW)
              TRACK(ROW) = TRACK(MAXCOL)
              TRACK(MAXCOL) = J
              DO 350 I = 1,M
                DUMMY = WORK(I,ROW)
                WORK(I,ROW) = WORK(I,MAXCOL)
                WORK(I,MAXCOL) = DUMMY
 350          CONTINUE
            ENDIF
C
C           CHECK WHETHER PIVOT ELEMENT IS ZERO.
            IF (DABS(WORK(ROW,ROW)).LT.10D-5) CHECK = 1
C
C           REDUCE THE TABLE IF NECESSARY.
            IF (CHECK.EQ.0) THEN
              DO 360 J = ROW,N
                J1 = N-J+ROW
                WORK(ROW,J1) = WORK(ROW,J1)/WORK(ROW,ROW)
                DO 360 K = 1,M
                  IF (K.NE.ROW) THEN
                    WORK(K,J1) = WORK(K,J1)-WORK(K,ROW)*WORK(ROW,J1)
                  ENDIF
 360          CONTINUE
            ENDIF
C
          ENDDO
C
          IF (CHECK.EQ.0) RANK = ROW
          IF (CHECK.EQ.1) RANK = ROW - 1
          RETURN
          END
```

```
C       ***************************************************************************
C       *                                                                         *
C       *                              R S U B                                    *
C       *                                                                         *
C       ***************************************************************************
C       *                                                                         *
C       * PURPOSE: THE PURPOSE OF THIS ROUTINE IS TO SUBTRACT TWO MAT-            *
C       *          RICES ( OF ANY SIZE, EVEN VECTORS ) AND RETURN THE             *
C       *          RESULT TO THE MAIN PROGRAM (REAL VERSION).                     *
C       *                                                                         *
C       ***************************************************************************
C       *                                                                         *
C       * VARIABLE CONVENTION:                                                    *
C       *   A AND B ARE THE MATRICES TO BE SUBTRACTED, C STORES RESULT.           *
C       *   AD1, AD2, BD1, BD2, CD1, CD2 ARE USED TO DECLARE ARRAY SIZE.          *
C       *   AS1, AS2 ARE THE ACTUAL SIZES OF THE MATRICES.                        *
C       *                                                                         *
C       ***************************************************************************
C
C
      SUBROUTINE RSUB(A,AD1,AD2,AS1,AS2,B,BD1,BD2,C,CD1,CD2)
      INTEGER AD1,AD2,BD1,BD2,CD1,CD2,AS1,AS2
      DOUBLE PRECISION A(AD1,AD2),B(BD1,BD2),C(CD1,CD2)
C
      DO 520 I=1,AS1
         DO 520 J=1,AS2
            C(I,J)=A(I,J)-B(I,J)
  520 CONTINUE
      RETURN
      END
```

The following main program was used to create the first example in Section 5 and the first example in Appendix C.

```
C     ************************************************************************
C     *                                                                      *
C     *                              M A I N                                 *
C     *                                                                      *
C     ************************************************************************
C
      DOUBLE PRECISION ID(10,10),Q(1,10),P(10,1),RINV(10,10),
     *     PQ(10,10),E(10,10),G(10,10),ASQR(10,10),ACUB(10,10),
     *     RESULT(10,10), ALPHA,BETA,DELTA,MAT(10,10),MAT1(10,10),
     *     A(10,10),R(10,10),AMLI(10,10),MAT2(10,10),LAM,B(30,10),
     *     U(10,10),L(10,10),Y(10,10)
      INTEGER RANK,NUM,SEED,DIM,TRACK1(10),TRACK2(10),S,T
C
C     READ DATA AND INITIALIZE IDENTITY MATRIX AND OTHER VARIABLES.
      READ (22,*) DIM,SEED,BETA,DELTA
      WRITE(21,*) ' BETA = ',BETA
      N = DIM - 1
      DO 20 I = 1,DIM
        DO 10 J = 1,DIM
          ID(I,J) = 0.0D0
 10     CONTINUE
        ID(I,I) = 1.0D0
 20   CONTINUE
C
C     RANDOMLY GENERATE A NONSINGULAR MATRIX R.
      DO 25 I = 1,SEED
        J = RAN(SEED)
 25   CONTINUE
      SUM = 0.0D0
      DO 30 I = 1,N
        J = 6.0D0*RAN(SEED) - 3.0D0
        Q(1,I) = J
        J = 6.0D0*RAN(SEED) - 3.0D0
        P(I,1) = J
        SUM = SUM + Q(1,I)*P(I,1)
 30   CONTINUE
      SUM = SUM - Q(1,N)*P(N,1)
      P(N,1) = -2.0D0
      Q(1,N) = SUM/2.0D0
      CALL RMULT(P,10,1,N,1,Q,1,10,1,N,PQ,10,10)
      CALL RSUB(ID,10,10,N,N,PQ,10,10,R,10,10)
      CALL RADD(ID,10,10,N,N,PQ,10,10,RINV,10,10)
C
C     GENERATE E AND G IN (4.11).
      DO 40 I = 1,N
        DO 40 J = 1,N
          E(I,J) = R(I,J)
```

```
            G(I,J) = (BETA/2.0D0)*RINV(I,J)
 40       CONTINUE
C
          DO 50 I = 1,N
            E(DIM,I) = Q(1,I)
            G(I,DIM) = -DELTA*P(I,1)
 50       CONTINUE
C
C         GENERATE ANOTHER NONSINGULAR MATRIX TO GENERALIZE A.
          CALL RMULT(E,10,10,DIM,N,G,10,10,N,DIM,MAT,10,10)
          SUM = 0.0D0
          DO 55 I = 1,DIM
            Q(1,I) = 4.0D0*RAN(SEED) - 2.0D0
            P(I,1) = 4.0D0*RAN(SEED) - 2.0D0
            SUM = SUM + Q(1,I)*P(I,1)
 55       CONTINUE
          SUM = SUM - Q(1,DIM)*P(DIM,1)
          P(DIM,1) = -2.0D0
          Q(1,DIM) = SUM/2.0D0
          CALL RMULT(P,10,1,DIM,1,Q,1,10,1,DIM,PQ,10,10)
          CALL RSUB(ID,10,10,DIM,DIM,PQ,10,10,R,10,10)
          CALL RADD(ID,10,10,DIM,DIM,PQ,10,10,RINV,10,10)
          CALL RMULT(RINV,10,10,DIM,DIM,MAT,10,10,DIM,DIM,MAT1,10,10)
          CALL RMULT(MAT1,10,10,DIM,DIM,R,10,10,DIM,DIM,A,10,10)
          CALL MATPR('    A =     ',A,10,10,DIM,DIM)
C         CHECK WHETHER A SATISFIES THEOREM 6(IV).
          ALPHA = 4.0D0/BETA
          CALL RMULT(A,10,10,DIM,DIM,A,10,10,DIM,DIM,ASQR,10,10)
          CALL RMULT(A,10,10,DIM,DIM,ASQR,10,10,DIM,DIM,ACUB,10,10)
          DO 60 I = 1,DIM
            DO 60 J = 1,DIM
              RESULT(I,J) = ALPHA*ACUB(I,J) - ALPHA*BETA*ASQR(I,J)
     *                    + BETA*A(I,J)
 60       CONTINUE
          CALL MATPR('RESULT =  ',RESULT,10,10,DIM,DIM)
C
C         GENERATE A - LAMBDA*I.
          LAM = BETA/2.0D0
          DO 70 I = 1,DIM
            DO 70 J = 1,DIM
              AMLI(I,J) = A(I,J) - LAM*ID(I,J)
 70       CONTINUE
          CALL RMULT(A,10,10,DIM,DIM,AMLI,10,10,DIM,DIM,MAT,10,10)
          CALL RMULT(AMLI,10,10,DIM,DIM,AMLI,10,10,DIM,DIM,MAT1,10,10)
C
C         GENERATE CANDIDATES FOR EIGENVECTORS OF A.
          CALL ROWRED(MAT,10,10,DIM,DIM,S,TRACK1,MAT2)
          CALL ROWRED(MAT1,10,10,DIM,DIM,T,TRACK2,MAT2)
C
C         PRINT OUT THE EIGENVECTORS.
          IF (T.GT.0) THEN
```

```
          WRITE(21,*) 'COLUMNS OF THE FOLLOWING MATRIX ARE EIGENVECTOR',
     *                'S OF A CORRESPONDING'
          WRITE(21,*) 'EIGENVALUE ZERO:'
          DO 80 I = 1,DIM
            WRITE(21,75) (MAT1(I,TRACK2(J)),J=1,T)
 75         FORMAT(10X,10F10.2)
 80       CONTINUE
          WRITE(21,*) '  '
        ENDIF
C
        IF (S.GT.0) THEN
          WRITE(21,*) 'COLUMNS OF THE FOLLOWING MATRIX ARE EIGENVECTOR',
     *                'S OF A CORRESPONDING'
          WRITE(21,*) 'TO EIGENVALUE LAMBDA = ',LAM
          DO 90 I = 1,DIM
            WRITE(21,75) (MAT(I,TRACK1(J)),J=1,S)
 90       CONTINUE
          WRITE(21,*) ' '
          WRITE(21,*) 'COLUMNS OF THE FOLLOWING MATRIX ARE GENERALIZED',
     *                ' EIGENVECTORS OF A'
          WRITE(21,*) 'ASSOCIATED WITH THE EIGENVECTORS IN THE PREVIOU',
     *                'S MATRIX:'
          DO 100 I = 1,DIM
            WRITE(21,75) (A(I,TRACK1(J)),J=1,S)
 100      CONTINUE
          WRITE(21,*) ' '
        ENDIF
C
C       USE THIS NEXT STEP ONLY IF 2S+T < DIM TO FIND REMAINING
C       EIGENVECTORS.
        IF ((2*S+T).LT.DIM) THEN
          DO 110 I = 1,DIM
             DO 110 J = 1,S
              U(I,J) = MAT(I,TRACK1(J))
 110      CONTINUE
          CALL LINVER(U,DIM,S,L)
          CALL RMULT(L,10,10,DIM,DIM,MAT,10,10,DIM,DIM,Y,10,10)
          K = 2*S + DIM
          DO 120 I = 1,K
            DO 120 J = 1,DIM
              IF (I.LE.S) THEN
                B(I,J) = Y(I,J)
              ELSE
                IF (I.LE.2*S) THEN
                  B(I,J) = L(I-S,J)
                ELSE
                  B(I,J) = MAT1(I-2*S,J)
                ENDIF
              ENDIF
 120      CONTINUE
          CALL NSPACE(B,30,10,K,DIM,MAT2,NUM)
```

```
            N = DIM-2*S-T
            WRITE(21,*) 'COLUMNS OF THE FOLLOWING MATRIX ARE SIMPLE EIGE',
     *                  'NVECTORS OF A '
            WRITE(21,*) 'CORRESPONDING TO EIGENVALUE LAMBDA MENTIONED ',
     *                  'ABOVE'
            CALL MATPR('           ',MAT2,10,10,DIM,N)
          ENDIF
C
          STOP
          END
```

The following main program was used to create the third example in Section 5 and the second example in Appendix C.

```
C       ***************************************************************************
C       *                                                                         *
C       *                              M A I N                                    *
C       *                                                                         *
C       ***************************************************************************
C
        DOUBLE PRECISION ID(10,10),Q(1,10),P(10,1),R(10,10),RINV(10,10),
     *       A(10,10),AML1I(10,10),AML2I(10,10),LAM1,LAM2,
     *       PQ(10,10),E(10,10),G(10,10),ASQR(10,10),ACUB(10,10),
     *       RESULT(10,10),ALPHA,BETA,DELTA,MAT(10,10),
     *       WORK(10,30),TOT(10,30),MAT2(10,10),MAT1(10,10)
        INTEGER RANK,TRACK(30),NUM,SEED,DIM
C
C       READ INITIAL DATA, CALCULATE EIGENVALUES AND INITIALIZE THE
C       IDENTITY MATRIX.
        READ (22,*) DIM,SEED,ALPHA,BETA
        WRITE(21,*) 'ALPHA = ',ALPHA,' BETA = ',BETA
        WRITE(21,*) ' '
        N = DIM - 1
        LAM1=(ALPHA*BETA+DSQRT((ALPHA*BETA)**2-4.0D0*ALPHA*BETA))/
     *       (2.0D0*ALPHA)
        LAM2=(ALPHA*BETA-DSQRT((ALPHA*BETA)**2-4.0D0*ALPHA*BETA))/
     *       (2.0D0*ALPHA)
C
        DO 20 I = 1,DIM
          DO 10 J = 1,DIM
            ID(I,J) = 0.0D0
 10       CONTINUE
          ID(I,I) = 1.0D0
 20     CONTINUE
C
C       CREATE THE NONSINGULAR MATRIX R AND ITS INVERSE.
        DO 25 I = 1,SEED
          J = RAN(SEED)
 25     CONTINUE
        SUM = 0.0D0
        DO 30 I = 1,N
          Q(1,I) = 10.0D0*RAN(SEED) - 5.0D0
          P(I,1) = 10.0D0*RAN(SEED) - 5.0D0
          SUM = SUM + Q(1,I)*P(I,1)
 30     CONTINUE
        SUM = SUM - Q(1,N)*P(N,1)
        P(N,1) = -2.0D0
        Q(1,N) = SUM/2.0D0
        CALL RMULT(P,10,1,N,1,Q,1,10,1,N,PQ,10,10)
        CALL RSUB(ID,10,10,N,N,PQ,10,10,R,10,10)
```

```
      CALL RADD(ID,10,10,N,N,PQ,10,10,RINV,10,10)
C
C     CREATE E AND G ANALAGOUS TO (4.12).
      Q(1,N) = SUM/2.0D0 - 0.5D0
      DO 40 I = 1,N
        DO 40 J = 1,N
          E(I,J) = R(I,J)
          G(I,J) = LAM1*RINV(I,J)
 40   CONTINUE
C
      DELTA = 2.0D0*LAM1 - BETA
      DO 50 I = 1,N
        E(DIM,I) = Q(1,I)
        G(I,DIM) = -DELTA*P(I,1)
 50   CONTINUE
C
C     CREATE ANOTHER NONSINGULAR MATRIX TO GENERALIZE A.
      CALL RMULT(E,10,10,DIM,N,G,10,10,N,DIM,MAT,10,10)
      SUM = 0.0D0
      DO 55 I = 1,DIM
        Q(1,I) = 4.0D0*RAN(SEED) - 2.0D0
        P(I,1) = 4.0D0*RAN(SEED) - 2.0D0
        SUM = SUM + Q(1,I)*P(I,1)
 55   CONTINUE
      SUM = SUM - Q(1,DIM)*P(DIM,1)
      P(DIM,1) = -2.0D0
      Q(1,DIM) = SUM/2.0D0
      CALL RMULT(P,10,1,DIM,1,Q,1,10,1,DIM,PQ,10,10)
      CALL RSUB(ID,10,10,DIM,DIM,PQ,10,10,R,10,10)
      CALL RADD(ID,10,10,DIM,DIM,PQ,10,10,RINV,10,10)
      CALL RMULT(RINV,10,10,DIM,DIM,MAT,10,10,DIM,DIM,MAT1,10,10)
      CALL RMULT(MAT1,10,10,DIM,DIM,R,10,10,DIM,DIM,A,10,10)
      CALL MATPR('    A =     ',A,10,10,DIM,DIM)
C     CHECK WHETHER A SATISFIES THEOREM 6(IV).
      CALL RMULT(A,10,10,DIM,DIM,A,10,10,DIM,DIM,ASQR,10,10)
      CALL RMULT(A,10,10,DIM,DIM,ASQR,10,10,DIM,DIM,ACUB,10,10)
      DO 60 I = 1,DIM
        DO 60 J = 1,DIM
          RESULT(I,J) = ALPHA*ACUB(I,J) - ALPHA*BETA*ASQR(I,J)
     *                  + BETA*A(I,J)
 60   CONTINUE
      CALL MATPR('RESULT =  ',RESULT,10,10,DIM,DIM)
C
C     CALCULATE A - LAMBDA1*I AND A - LAMBDA2*I.
      DO 70 I = 1,DIM
        DO 70 J = 1,DIM
          AML1I(I,J) = A(I,J) - LAM1*ID(I,J)
          AML2I(I,J) = A(I,J) - LAM2*ID(I,J)
 70   CONTINUE
C
C     CONSTRUCT MATRIX IN (3.27).
```

```
      CALL RMULT(AML1I,10,10,DIM,DIM,AML2I,10,10,DIM,DIM,TOT,30,10)
      CALL RMULT(A,10,10,DIM,DIM,AML1I,10,10,DIM,DIM,MAT1,10,10)
      CALL RMULT(A,10,10,DIM,DIM,AML2I,10,10,DIM,DIM,MAT2,10,10)
      K = 3*DIM
      DIMP1 = DIM + 1
      DO 80 I = 1,DIM
        DO 80 J = DIMP1,K
          IF (J.LE.2*DIM) THEN
            TOT(I,J) = MAT1(I,J-DIM)
          ELSE
            TOT(I,J) = MAT2(I,J-2*DIM)
          ENDIF
 80   CONTINUE
C
C     FIND LINEARLY INDEPENDENTS SET OF COLUMNS.
      CALL ROWRED(TOT,10,30,DIM,K,RANK,TRACK,WORK)
      I0 = 0
      I1 = 0
      I2 = 0
      DO 90 I = 1,RANK
        IF (TRACK(I).LE.DIM) THEN
          I0 = I0 + 1
          DO 81 J = 1,DIM
            MAT(J,I0) = TOT(J,TRACK(I))
81        CONTINUE
        ELSE
          IF (TRACK(I).LE.2*DIM) THEN
            I1 = I1 + 1
            DO 82 J = 1,DIM
              MAT1(J,I1) = TOT(J,TRACK(I))
 82         CONTINUE
          ELSE
            I2 = I2 + 1
            DO 83 J = 1,DIM
              MAT2(I,J) = TOT(J,TRACK(I))
 83         CONTINUE
          ENDIF
        ENDIF
C
 90   CONTINUE
C
C     PRINT OUT THE EIGENVECTORS.
      IF (I0.NE.0) THEN
      WRITE(21,*) 'COLUMNS OF THE FOLLOWING MATRIX ARE EIGENVECTORS',
     *            ' CORRESPONDING TO'
      WRITE(21,*) 'EIGENVALUE ZERO:'
      CALL MATPR('          ',MAT,10,10,DIM,I0)
      ENDIF
C
      IF (I1.NE.0) THEN
      WRITE(21,*) 'COLUMNS OF THE FOLLOWING MATRIX ARE EIGENVECTORS',
```

```
     *              ' CORRESPONDING TO'
            WRITE(21,*) 'EIGENVALUE LAMBDA1 = ',LAM1
            CALL MATPR('          ',MAT1,10,10,DIM,I1)
            ENDIF
C
            IF (I2.NE.0) THEN
            WRITE(21,*) 'COLUMNS OF THE FOLLOWING MATRIX ARE EIGENVECTORS',
     *              ' CORRESPONDING TO'
            WRITE(21,*) 'EIGENVALUE LAMBDA2 = ',LAM2
            CALL MATPR('          ',MAT2,10,10,DIM,I2)
            ENDIF
C
            STOP
            END
```

APPENDIX C

NUMERICAL EXAMPLES

NUMERICAL EXAMPLES

The following is an example of a 7 by 7 matrix A satisfying Theorem 6(iv) with $\beta = 1$ and $\alpha = 4$, and its eigenvector structure. The matrix entitled "RESULT" is simply a check of whether $\alpha A^3 - \alpha\beta A^2 + \beta A = 0$.

BETA = 1.00000

A =	37.78	-12.85	13.10	9.65	-16.18	4.50	-43.76
	-6.59	3.27	-3.54	-2.08	2.05	-0.25	9.20
	25.71	-8.67	9.07	6.51	-11.47	3.31	-29.63
	29.44	-9.93	9.82	7.96	-13.13	3.79	-33.93
	15.68	-5.29	5.23	3.97	-6.49	2.02	-18.07
	-4.78	2.16	-2.94	-1.62	1.24	0.48	7.11
	41.92	-14.57	15.03	10.94	-18.00	4.92	-49.07

RESULT =	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.00	0.00

COLUMNS OF THE FOLLOWING MATRIX ARE EIGENVECTORS OF A CORRESPONDING EIGENVALUE ZERO:

28.40

-5.43

19.43

22.25

11.85

-4.06

32.03

COLUMNS OF THE FOLLOWING MATRIX ARE EIGENVECTORS OF A CORRESPONDING TO EIGENVALUE LAMBDA = 0.50000

-8.28

1.06

-5.86

-6.71

-3.58

0.65

-9.21

COLUMNS OF THE FOLLOWING MATRIX ARE GENERALIZED EIGENVECTORS OF A ASSOCIATED WITH THE EIGENVECTORS IN THE PREVIOUS MATRIX:

-16.18

2.05

-11.47

-13.13

-6.49

1.24

-18.00

COLUMNS OF THE FOLLOWING MATRIX ARE SIMPLE EIGENVECTORS OF A

CORRESPONDING TO EIGENVALUE LAMBDA MENTIONED ABOVE

1.00	0.00	0.00	0.00
0.00	1.00	0.00	0.00
-1.00	0.61	0.17	-0.46
0.00	0.00	0.00	1.00
1.50	-0.30	0.42	0.23
0.00	0.00	1.00	0.00
0.00	0.00	0.00	0.00

The following is an example of a 7 by 7 matrix A satisfying Theorem 6(iv) with $\beta = 1$ and $\alpha = 6$, and its eigenvector structure. The matrix "RESULT" fulfills the same role as in the previous example.

ALPHA = 6.00000 BETA = 1.00000

A =

8.79	2.46	-0.78	4.92	-19.21	7.72	-0.42
10.67	4.61	-2.28	5.96	-25.40	9.36	-0.28
4.65	1.35	0.53	2.95	-11.19	4.63	-0.29
3.11	0.89	-0.15	2.78	-7.50	3.12	-0.20
7.42	1.50	1.10	5.45	-17.35	8.55	-0.80
4.72	1.05	0.48	3.36	-11.50	6.06	-0.46
16.53	2.81	3.67	12.73	-40.62	19.97	-1.26

RESULT =

0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00

COLUMNS OF THE FOLLOWING MATRIX ARE EIGENVECTORS CORRESPONDING TO EIGENVALUE ZERO:

-31.85

-43.21

-18.40

-12.30

-28.50

-18.26

-62.78

COLUMNS OF THE FOLLOWING MATRIX ARE EIGENVECTORS CORRESPONDING TO EIGENVALUE LAMBDA1 = 0.78868

7.54

10.29

4.35

2.90

6.66

4.28

14.61

COLUMNS OF THE FOLLOWING MATRIX ARE EIGENVECTORS CORRESPONDING TO EIGENVALUE LAMBDA2 = 0.21132

7.30	10.13	4.18	2.79	6.20
-0.26	19.87	-45.01	-22.08	8.54
5.92	8.54	3.34	2.22	4.99
0.37	-0.26	-0.03	-0.02	0.13
-15.86	-22.08	-9.07	-5.60	-13.36
-32.47	-45.01	-18.15	-12.42	-27.63
0.44	26.04	-59.89	-28.86	9.48

VITA

Oliver K. Tavakoli was born in London, England on November 29, 1963. At the age of four his family moved to Tehran, Iran, where he attended elementary and middle school. In 1978 he moved to Knoxville, Tennessee and attended Farragut High School. He was graduated in January 1981 and entered the University of Tennessee in March 1981. In December 1983 he received a Bachelor of Art degree with a double major in Mathematics and Computer Science. He accepted a graduate teaching assistantship in the Mathematics Department in January 1984 and began study toward a Master's degree. This degree was awarded in June 1985. Due to his international background, Mr. Tavakoli holds citizenships in Germany, England and Iran.

The author is a member of the American Mathematical Society and Phi Beta Kappa and Phi Kappa Phi national honor societies. He will be employed by IBM in Sterling Forest, New York after graduation.