Global Journal of Pure and Applied Mathematics.
ISSN 0973-1768 Volume 13, Number 9 (2017), pp. 4959-4967

http://www.ripublication.com

Iterative Algorithm to Find the Solution of Multiparameter Matrix Eigenvalue Problem

N. Bora[1] and **A. K. Baruah**[2]

[1]*Department of Mathematics, Dibrugarh University Institute of Engineering and Technology, Dibrugarh University, Dibrugarh, Assam-786004, India.*
[2]*Department of Mathematics, Dibrugarh University, Dibrugarh, Assam-786004, India.*

Abstract

In this paper, an iterative algorithm based on Newton's method is examined for finding the approximate eigenvalues of Multiparameter matrix eigenvalue problem (MMEP). Three different iterative algorithms are presented to calculate the partial derivative of matrix determinant appears in this Method. This approach provides us an elegant numerical procedure to find numerical solution of original MMEPs, but the computational works are done by considering three-parameter case to relax computational cost. MATLAB code has been developed for each algorithm.

Keywords: Tensor Product Space, Newton's Method, Implicit Determinant Algorithm.

AMS Subject Classification: 35PXX, 65FXX, 65F15, 35A35 (2010).

1. INTRODUCTION:

The abstract settings of MMEPs to be studied is

$$\left(A_i - \sum_{j=1}^{k} \lambda_j B_{ij}\right) x_i = 0, \quad i = 1, 2, \ldots\ldots, k \tag{1.1}$$

where the problem is to find k-tuple of values $\lambda = (\lambda_1, \lambda_2, \lambda_3, \lambda_4 \ldots\ldots\ldots\ldots \lambda_k) \epsilon C^k$ for non-zero vector x_i. The operators $A_{i,}\, B_{ij}$ are

self-adjoint, bounded and linear that act on seperable Hilbert Spaces H_i, $x_i \in H_i$. The k-tuple $\lambda \epsilon C^k$ is called an eigenvalue and the decomposible tensor product $x = x_1 \otimes x_2 \otimes x_3 \dots\dots\dots\otimes x_k$ is the corresponding (right) eigenvector. Similarly left eigenvector can also be defined. The usual approach to solve (1.1) suggested by Atkinson [2], [3] are by transforming it into a system of generalized eigenvalue problem given by

$$\Delta_{\mathrm{i}} \mathrm{z} = \lambda_{\mathrm{i}} \Delta_0 \mathrm{z} \tag{1.2}$$

where Δ_{i} are defined by

$$\Delta_{\mathrm{i}} = \begin{vmatrix} \mathrm{B}_{11}^{+} & \dots & \mathrm{B}_{1,\mathrm{i}-1}^{+} & \mathrm{A}_{1}^{+} & \mathrm{B}_{1,\mathrm{i}+1}^{+} & \dots & \mathrm{B}_{1\mathrm{k}}^{+} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathrm{B}_{\mathrm{k}1}^{+} & \dots & \mathrm{B}_{\mathrm{k},\mathrm{i}-1}^{+} & \mathrm{A}_{\mathrm{k}}^{+} & \mathrm{B}_{\mathrm{k},\mathrm{i}+1}^{+} & \dots & \mathrm{B}_{\mathrm{kk}}^{+} \end{vmatrix} \tag{1.3}$$

and $\Delta_0 = \otimes \left[\mathrm{B}_{\mathrm{ij}}\right]$, where the linear transformation $\mathrm{B}_{\mathrm{ij}}^{+}$ on H are induced by B_{ij} and are defined by

$$\mathrm{B}_{\mathrm{ij}}^{+}(x_1 \otimes x_2 \dots\dots\dots\otimes x_k) = x_1 \otimes x_2 \otimes \dots\dots \otimes x_{i-1} \otimes \mathrm{B}_{\mathrm{ij}} x_i \otimes x_{i+1} \otimes \dots\otimes x_k$$

On the decomposable tensor $x_1 \otimes x_2 \dots\dots\dots\otimes x_k$ where $x_i \in H_i$, extended to H by linearity.

If Δ_0 is positive definite, then the matrices ${\Delta_0}^{-1}\Delta_{\mathrm{i}}, i = 1,2, \dots\dots\dots\dots, k$ commute and all the eigenvalues of (1.1) coincide with the eigenvalues of (1.2). This approach is suitable only for the matrices of small orders, as the computational cost and time is more to calculate the operator determinants Δ_{i} which is of order n^n if the corresponding matrices are of order n. Hence it necessary to exploit iterative methods to compute eigenvalues closes to a given target. In this paper we will study a numerical algorithm based on Newton's Method to find the value of λ that yields a non trivial solution of x.

2. NEWTON'S METHOD:

The eigenvalues are solution of the following system of equations

$$\begin{gathered} f_1 \equiv det(A_1 - \lambda_1 B_{11} - \lambda_2 B_{12} - \lambda_{\mathrm{k}} B_{1k}) = 0 \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ f_1 \equiv det(A_k - \lambda_1 B_{k1} - \lambda_2 B_{k2} - \lambda_{\mathrm{k}} B_{kk}) = 0 \end{gathered} \tag{2.1}$$

On the above systems an iterative process will be presented to calculate the eigenvalues of the system (1.1) by using Newton's Method as a tool without expanding the determinants in (2.1). In this process the left-hand sides in system (2.1) are not calculated explicitly, which will be replaced by an algorithm to calculate the functions $f_i(\lambda_1, \lambda_2, \lambda_3, \dots\dots\dots\dots \lambda_k)$, as well as the entries of the Jacobian matrix of system (2.1), for any fixed values of λ. Assume that $(\lambda_1^0, \lambda_2^0, \dots\dots\dots, \lambda_k^0)$ be an approximation of λ, then the iteration of the Newton's method can be written as

$$\lambda_1^{(n+1)} = \lambda_1^n + \Delta\lambda_1^n;\ \lambda_2^{(n+1)} = \lambda_2^n + \Delta\lambda_2^n; \dots \lambda_k^{(n+1)} = \lambda_k^n + \Delta\lambda_k^n\ ;\ n = 1,2, \dots. \quad (2.2)$$

The correction $\Delta\lambda_1^n, \Delta\lambda_2^n, \dots \dots, \Delta\lambda_k^n$ can be obtained by solving the following equations

$$\frac{\partial f_1}{\partial \lambda_1}\Delta\lambda_1^n + \frac{\partial f_1}{\partial \lambda_2}\Delta\lambda_2^n + \dots \dots + \frac{\partial f_1}{\partial \lambda_k}\Delta\lambda_k^n = f_1$$

$$\dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots$$

$$\frac{\partial f_k}{\partial \lambda_1}\Delta\lambda_1^n + \frac{\partial f_k}{\partial \lambda_2}\Delta\lambda_2^n + \dots \dots + \frac{\partial f_k}{\partial \lambda_k}\Delta\lambda_k^n = f_k \quad (2.3)$$

where all coefficients (i.e functions f_i and $\frac{\partial f_i}{\partial \lambda_i}$ are calculated at $(\lambda_1^n, \lambda_2^n, \dots \dots \dots, \lambda_k^n)$.

3. ALGORITHM FOR DERIVATIVE OF MATRIX DETERMINANT:

To apply Newton's iterative scheme, we need an efficient formula to find the partial derivatives $\frac{\partial f_i}{\partial \lambda_i}$ of matrix determinant where $i = 1, 2, \dots \dots, k$. We use the following Jacobi's formula based on trace of matrix

$$\frac{\partial f_i}{\partial \lambda_i}(\lambda_1, \lambda_2, \dots \dots \lambda_k) = f_i(\lambda_1, \lambda_2, \dots \dots \dots \lambda_k).\, trace([T_i(\lambda_1, \lambda_2, \dots \dots \dots \lambda_k)]^{-1}\frac{\partial T_i}{\partial \lambda_i}\ (\lambda_1, \lambda_2, \dots \dots \dots \lambda_k)) \quad (3.1)$$

Using this above formula the partial derivative $\frac{\partial f_i}{\partial \lambda_i}$ can be calculated as follows $\frac{\partial f_1}{\partial \lambda_1} = f_1.\, trace[(A_1 - \lambda_1 B_{11} - \lambda_2 B_{12} - \lambda_k B_{1k})^{-1}(-B_{11})]$; provided T_1 is non-singular. In a similar way the other partial derivative appears in (2.3) can be calculated. This scheme is presented in Algorithm 4:1.

Algorithm 4.1

- Step 1: Calculate the matrix $T_i = A_i - \Sigma\lambda_i B_{ij}$
- Step 2: Find all partial derivatives $\frac{\partial f_i}{\partial \lambda_i}(\lambda_1, \lambda_2, \dots \dots \lambda_k)$ with the help of (3.1).
- Step 3: Form the Jacobian matrix from the co-efficient $\frac{\partial f_i}{\partial \lambda_1}(\lambda_1, \lambda_2, \dots \dots \lambda_k); \dots \dots \dots \dots \frac{\partial f_i}{\partial \lambda_k}(\lambda_1, \lambda_2, \dots \dots \lambda_k)$ and solve system (2.3) with respect to $\Delta\lambda_1^k, \Delta\lambda_2^k, \dots \dots, \Delta\lambda_k^k$
- Step 4: Calculate the next approximation of $\lambda_1, \lambda_2, \dots \dots \lambda_k$ using iterative formula (2.2).

Next we present another approach based on the LU-decomposition of the matrix. The following theorem from [5] will be required to calculate the matrix decomposition.

Theorem 3.1: If the elements of a square matrix $A(\lambda)$ are differentiable functions of λ then the derivatives of the matrix determinant $detA(\lambda) = f(\lambda)$ of the matrix $A(\lambda)$ satisfy the following relations

$$f(\lambda) = (-1)^q \Pi\, u_{ii};\; f(\lambda)' = \boldsymbol{\Sigma_{k=1}^{n}}\, v_{kk} \Pi_{1=i\neq k}^{n}\, u_{ii} \tag{3.2}$$

Where $u_{ii}(\lambda)$ and $v_{ii}(\lambda)$ are the elements of the upper triangular matrices $U(\lambda)$ and $V(\lambda)$ in the decompositions

$$A(\lambda) = L(\lambda)\, U(\lambda);\; B(\lambda) = M(\lambda)\, U(\lambda) + L(\lambda)\, V(\lambda) \tag{3.3}$$

and $L(\lambda)$ is a unit lower triangular matrix.

Bothe [4] developed an algorithm to find the entries of the matrices appearing in decomposition (3.3) as follows
$\forall\ r = 1, 2, 3, \ldots\ldots\ldots\ldots, n;\ k = r, \ldots\ldots\ldots, n \text{ and } i = r+1, \ldots\ldots\ldots\ldots\ldots\ldots n$

$$u_{rk} = a_{rk} - \sum_{1}^{r-1} l_{rj} u_{jk}$$

$$l_{ir} = \left(a_{ir} - \sum_{j=1}^{r-1} l_{ij} u_{jr} \right) \Big/ u_{rr}$$

$$V_{rk} = b_{rk} - \sum_{j=1}^{r-1} (m_{rj} u_{jk} - l_{rj} V_{jk})$$

$$m_{ir} = \left[b_{ir} - \sum_{j=1}^{r-1} \left(m_{ij} u_{jr} - l_{ij} V_{jr} \right) - l_{ir} V_{rr} \right] \Big/ u_{rr}$$

The above algorithm can be applied to find $f(\lambda)$ and $f'(\lambda)$ i.e the coefficients of the system (2.3) as shown below

Algorithm 4.2

- Step 1: Calculate the matrix $A = A_1 - \Sigma\lambda_{\mathrm{i}} B_{1i}$
- Step 2 : 1. Take B_{11} as the matrix B; thus, $B = -B_{11}$. Using (3.2), calculate $f_1(\lambda_1, \lambda_2, \ldots\ldots\lambda_k)$ and $\frac{\partial f_1}{\partial \lambda_1}(\lambda_1, \lambda_2, \ldots\ldots\lambda_k)$.
- Step 2 : 2: Take B_{11} as the matrix B; thus, $B = -B_{12}$. Using (3.2), calculate $f_1(\lambda_1, \lambda_2, \ldots\ldots\lambda_k)$ and $\frac{\partial f_1}{\partial \lambda_2}(\lambda_1, \lambda_2, \ldots\ldots\lambda_k)$.

  . . . . . .
- Step 2 : n: Take B_{1k} as the matrix B; thus, $B = -B_{1k}$. Using (3.2), calculate $f_1(\lambda_1, \lambda_2, \ldots\ldots\lambda_k)$ and $\frac{\partial f_1}{\partial \lambda_k}(\lambda_1, \lambda_2, \ldots\ldots\lambda_k)$.

The co-efficient of the system (2.3) can be calculated in a similar pattern. Now it remains to perform another two steps prescribed by Newton's Method.

- Step 3: Form the Jacobian matrix from the co-efficients $\frac{\partial f_i}{\partial \lambda_1}(\lambda_1, \lambda_2, \dots \dots \lambda_k)$; $\dots \dots \dots \dots . \frac{\partial f_i}{\partial \lambda_k}(\lambda_1, \lambda_2, \dots \dots \lambda_k)$ and solve system (2.3) with respect to $\Delta\lambda_1^k, \Delta\lambda_2^k, \dots \dots, \Delta\lambda_k^k$.
- Step 4: Calculate the next approximation of $\lambda_1, \lambda_2, \dots \dots \lambda_k$ using iterative formula (2.2).

The following algorithm based on the implicit determinant algorithm, which is found in [1] for one parameter non linear eigenvalue problems can be used as an alternate method to calculate the partial derivatives.

Definition 3.2: *An eigenvalue* $(\lambda_1, \lambda_2, \dots \dots \lambda_k)$ *of the MMEP is said to be (geometrically) simple if and only if* $Dim\, Ker\left(A_i - \Sigma\lambda_i B_{ij}\right) = 1$, *where* $i = 1,2, \dots ., k$.

Lemma (3.3) from [6] is required to find non singularity conditions of bordered matrix.

Lemma 3.3: ABCD: Given an $n \times n$ matrix A with $Rank\,(A) = n - 1$ and $b, c \in C^n$ and $d \in C$, then the bordered matrix M of order $(n+1) \times (n+1)$ such that $M = \begin{bmatrix} A & b \\ c^T & d \end{bmatrix}$ is non singular if and only if $\Psi^T b \neq 0;\ \forall\, \Psi \in Ker(A^T) - \{0\}$ and $c^T\theta \neq 0;\ \forall\, \theta \in Ker(A) - \{0\}$.

Theorem 3.4: Let $(\lambda_1^*, \lambda_2^*, \dots \dots \dots, \lambda_k^*)$ be an algebraically simple eigenvalue of the MMEPs (1.1) and let $(x_1 \otimes x_2 \otimes x_3 \dots \dots \dots \otimes x_k)$ and $(y_1 \otimes y_2 \otimes y_3 \dots \dots \dots \otimes y_k)$ be the corresponding right and left eigenvectors. If vectors u_i and v_i are such that ${y_i}^T u_i \neq 0;\ \forall\, y_i \in Ker(T_i)^T - \{0\}$ and ${v_i}^T x_i \neq 0;\ \forall\, x_i \in KerT_i - \{0\}$, then the bordered matrix

$$M_i\,(\lambda_1, \lambda_2, \dots \dots \lambda_k) = \begin{bmatrix} T_i\,(\lambda_1, \lambda_2, \dots \dots \lambda_k) & u_i \\ {v_i}^T & 0 \end{bmatrix} \tag{3.4}$$

is non singular at $(\lambda_1^*, \lambda_2^*, \dots \dots \dots, \lambda_k^*)$ at $i = 1,2, \dots, k$.

Proof: If $(\lambda_1^*, \lambda_2^*, \dots \dots \dots, \lambda_k^*)$ be an algebraically simple eigenvalue of the MMEPs (1.1), then
$Dim\, KerT_i\,(\lambda_1, \lambda_2, \dots \dots \lambda_k) = 1$ at $(\lambda_1^*, \lambda_2^*, \dots \dots \dots, \lambda_k^*)$.
By Rank-Nulity theorem of linear algebra
$Rank\, T_i\,(\lambda_1, \lambda_2, \dots \dots \lambda_k) + Dim\, KerT_i\,(\lambda_1, \lambda_2, \dots \dots \lambda_k) = n$
$Rank\, T_i\,(\lambda_1, \lambda_2, \dots \dots \lambda_k) = n - 1$
Using ABCD Lemma above the bodered matrix M_i is non singular if and only if ${y_i}^T u_i \neq 0;\ \forall\, y_i \in Ker(T_i)^T - \{0\}$ and ${v_i}^T x_i \neq 0;\ \forall\, x_i \in KerT_i - \{0\}$. ■

Near to $(\lambda_1^*, \lambda_2^*, \ldots\ldots\ldots, \lambda_k^*)$ we de_ne the vector $x_i\ (\lambda_1, \lambda_2, \ldots\ldots \lambda_k)$ and scalars $g_i\ (\lambda_1, \lambda_2, \ldots\ldots \lambda_k)$ as the solution of the system

$$M_i\ (\lambda_1, \lambda_2, \ldots\ldots \lambda_k) \begin{bmatrix} x_i\ (\lambda_1, \lambda_2, \ldots\ldots \lambda_k) \\ g_i\ (\lambda_1, \lambda_2, \ldots\ldots \lambda_k) \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.5}$$

By Cramer's rule $g_i\ (\lambda_1, \lambda_2, \ldots\ldots \lambda_k) = \frac{det T_i\ (\lambda_1, \lambda_2, \ldots\ldots \lambda_k)}{det\ M_i\ (\lambda_1, \lambda_2, \ldots\ldots \lambda_k)}$ and $f_i\ (\lambda_1, \lambda_2, \ldots\ldots \lambda_k) = 0$ has the root at $(\lambda_1^*, \lambda_2^*, \ldots\ldots\ldots, \lambda_k^*)$ and $g_i\ (\lambda_1, \lambda_2, \ldots\ldots \lambda_k) = 0$. Differentiating equation (3.5) we get the following linear system

$$M_i\ (\lambda_1, \lambda_2, \ldots\ldots \lambda_k) \begin{bmatrix} \frac{\partial x_i}{\partial \lambda_1} & \ldots & \frac{\partial x_i}{\partial \lambda_k} \\ \frac{\partial g_i}{\partial \lambda_1} & \ldots & \frac{\partial g_i}{\partial \lambda_k} \end{bmatrix} = \begin{bmatrix} \frac{\partial T_i}{\partial \lambda_1}.x_i & \ldots & \frac{\partial T_i}{\partial \lambda_k}.x_i \\ 0 & \ldots & 0 \end{bmatrix} \tag{3.6}$$

Solving the (3.6) system for partial derivatives $\frac{\partial g_i}{\partial \lambda_k}$ we get the necessary Newton's updates required in (2.2). The following Algorithm reflects the whole method.

Algorithm 4.3

- Step1: Start with $(\lambda_1^0, \lambda_2^0, \ldots\ldots\ldots, \lambda_k^0)$ and non zero vectors u_i and v_i where $i = 1, 2, \ldots\ldots\ldots$
- Step 2: For $k = 1, 2, \ldots\ldots\ldots$ until convergence do
- Step 3: Solve $\begin{bmatrix} T_i\ (\lambda_1, \lambda_2, \ldots\ldots \lambda_k) & u_i \\ {v_i}^T & 0 \end{bmatrix} \begin{bmatrix} x_i \\ \gamma_i \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$
- Step 4: Solve $\begin{bmatrix} T_i\ (\lambda_1, \lambda_2, \ldots\ldots \lambda_k) & u_i \\ {v_i}^T & 0 \end{bmatrix} \begin{bmatrix} z_i^1 & \ldots & z_i^k \\ a_i^1 & \ldots & a_i^k \end{bmatrix} = \begin{bmatrix} \frac{\partial T_i}{\partial \lambda_1}.x_i & \ldots & \frac{\partial T_i}{\partial \lambda_k}.x_i \\ 0 & \ldots & 0 \end{bmatrix}.$
- Step 5: Solve $\begin{bmatrix} a_1^1 & \cdots & a_1^k \\ \vdots & \ddots & \vdots \\ a_i^1 & \cdots & a_i^k \end{bmatrix} \begin{bmatrix} \Delta\lambda_1 \\ \vdots \\ \Delta\lambda_k \end{bmatrix} = -\begin{bmatrix} \gamma_1 \\ \vdots \\ \gamma_k \end{bmatrix}$
- Step 6: Upgrade $\lambda_1^{(n+1)} = \lambda_1^n + \Delta\lambda_1^n;\ \lambda_2^{(n+1)} = \lambda_2^n + \Delta\lambda_2^n; \ldots\ldots\ldots\ldots \lambda_k^{(n+1)} = \lambda_k^n + \Delta\lambda_k^n$

4. ANALYSIS OF THE ALGORITHMS:

The algorithm 4:1 can be applied if the initial approximation of eigenvalues is chosen in such a way that operators $T_i(\lambda_1, \lambda_2, \ldots\ldots\ldots\ldots \lambda_k)$ are non singular, ∀= 1, 2,….., k. Again it requires the concept of inversion of matrix. If some principal minors of the matrix A of order $j \leq n-1$ vanishes, then decompositions (3.3) does not always exist or, it is multi-valued if it exists. Thus the algorithm 4:2 is numerically unstable from step 2:1 to step 2: n if u_{rr} for some r. But if the matrix in decomposition (3.3) are non singular, then the algorithm can be made numerically stable. This can be done by using well known technique of pivoting, which uses a series of permutations of rows (and / or columns) of the matrix A with choosing its principal element in decomposition part. Then the decomposition takes the form PA = LU, PB = MU +

LV, where P is the permutation matrix and $det(P) = (-1)^q$, q is the number of permutations (e.g. rows). The equation (3.2) takes the form

$$f(\lambda) = (-1)^q \Pi\, u_{ii};\ f(\lambda)' = \boldsymbol{\Sigma_{k=1}^{n}}\, v_{kk} \Pi_{1=i\neq k}^{n}\, u_{ii} \qquad (4.1)$$

In algorithm 4:3 initial approximations for both eigenvalues and component vectors u_i and v_i are required. This algorithm depends on the selection of the vectors u_i and v_i. The optimal selection for u_i and v_i are left and right eigenvector components $u_i = y$ and $u_i = x_i$ where $i = 1, 2, \ldots\ldots, k$ so that bordered matrix must be non singular. The algorithm can be applied if the eigenvalues of the problem are algebraically simple.

5. NUMERICAL ILLUSTRATION:

For numerical illustration we will the following three-parameter problems

$$\begin{pmatrix}1&3&0\\4&4&4\\5&7&6\end{pmatrix}u_1 = \lambda_1\begin{pmatrix}25&2&6\\0&18&0\\1&0&20\end{pmatrix} + \lambda_2\begin{pmatrix}2&0&2\\2&4&0\\0&0&5\end{pmatrix} + \lambda_3\begin{pmatrix}7&5&0\\5&8&0\\4&0&2\end{pmatrix}$$

$$\begin{pmatrix}9&8&0\\7&1&7\\0&3&1\end{pmatrix} = \lambda_1\begin{pmatrix}4&3&3\\0&4&0\\3&0&4\end{pmatrix} + \lambda_2\begin{pmatrix}55&0&1\\0&53&7\\2&0&45\end{pmatrix} + \lambda_3\begin{pmatrix}3&0&9\\3&4&0\\0&0&3\end{pmatrix}$$

$$\begin{pmatrix}5&2&3\\7&5&0\\0&0&6\end{pmatrix}u_3 = \lambda_1\begin{pmatrix}8&4&5\\0&5&0\\9&0&1\end{pmatrix} + \lambda_2\begin{pmatrix}6&4&8\\4&4&0\\1&0&9\end{pmatrix} + \lambda_3\begin{pmatrix}54&6&0\\0&44&0\\8&0&33\end{pmatrix}$$

The exact number of common 3-tuples of the above system will be $3^3 = 27$. In table 1and 2 approximate eigenvalues with three different initial guesses has been calculated using algorithm presented above.

Table 1: Approximate eigenvalue by Algorithm 4.1 and 4.2

Initial Guess	**Iterations**	**Eigenvalues by algorithm 4 : 1**	**Eigenvalues by algorithm 4 : 1**
$\lambda = (1, 1, 1)$	1	(0.7190, 0.6883, 0.7032)	(0.6810, 0.5564, 0.6557)
	.....	.....	.....
	10	(0.4152, 0.1972, 0.1218)	(-0.1748, 0.2755, 0.2114)
	11	(0.4152, 0.1972, 0.1218)	(-0.2556, 0.3220, 0.2194)
$\lambda = (1, 2, 0.5)$	1	(0.7170, 1.3526, 0.3656)	(0.7389, 1.0193, 0.3585)
	.....	.....	.....
	9	(0.4152, 0.1972, 0.1218)	(0.1343, 0.2403, 0.1375)
	10	(0.4152, 0.1972, 0.1218)	(0.2474, 0.2957, 0.0769)

$\lambda = (-1, -1, -1)$	1	(-0.6386, -0.6556, -0.6283)	(-0.6618, -0.3971, -0.5955)
	.....	.....	.....
	9	..	(-0.1999, 0.3056, -0.0600)
	10	(-0.0636, -0.1138, 0.0390)	(-0.1478, 0.3468, -0.0109)
		(-0.0636, -0.1138, 0.0390)	

Table 2: Approximate eigenvalue by Algorithm 4.3

Initial Guess	**No of Iteration**	**Eigenvalues by algorithm 4 : 3**
$\lambda = (1, 1, 1)$ $u_i = (1\ \ 1\ \ 1)^T$ $v_i = (1\ \ 1\ \ 1)^T$	1 … 19 20	(0.5890, 0.5192, 0.5345) ….. (-0.0273, 0.2381, 0.1777) (-0.0273, 0.2381, 0.1777)
$\lambda = (-1, 3, -1)$ $u_1 = (1\ \ 2\ \ 1)^T$ $u_2 = (1\ \ 1\ \ 1)^T$ $u_3 = (3\ \ 2\ \ 1)^T$ $v_1 = (2\ \ 2\ \ 1)^T$ $v_2 = (2\ \ 1\ \ 2)^T$ $3 = (1\ \ 1\ \ 1)^T$	1 … 29 30	(-1.2189, 1.0325, -0:5020) ….. (-0.0273, 0.2381, 0.1777) (-0:0273, 0.2381, 0.1777)

6. CONCLUSION:

All the algorithms presented here can be easily extended to n parameter case. These algorithms can be used to find the initial approximate eigenvalue required for the other Methods and also useful to find partial derivatives of matrix determinant arises in numerous scientific problems.

REFERENCES:

[1] Akinola, R. O., Freitag, M. A., and Spence, A., 2014, "The computation of Jordan blocks in parameter-dependent matrices", IMA Journal of Numerical Analysis," 34, pp.955-976.

[2] Atkinson, F. V., 1972, "Multiparameter Eigenvalue Problems, Vol. I, (Matrices and Compact Operators)", Academic Press, New York.

[3] Atkinson, F. V., 1968, "Multiparameter Spectral theory", Bull. Amer. Math. Soc., 74, pp. 1-27.

[4] Bothe, Z., 1981, "Calculation of the derivative of the determinant", Obzornik mat. Fiz. 28, pp.33-50.

[5] Podlevskyi, B. M. and Yaroshko, O. S., 2013, "Newton's method for the solution of inverse spectral problems", Journal of Mathematical Sciences, 194 (2), pp. 156-165.

[6] Spence, A and Graham, I. G., 2002, "Numerical Methods for Bifurcation problems" University of Bath.