

Determining Matrix Eigenvalues and Eigenvectors by Jacobi Algorithm

Mark Pedley and Michael Stanley

1 Introduction

1.1 Summary

This application note documents the *Jacobi rotation* eigenanalysis algorithm in the [Freescal Sensor Fusion Library](#) software and implemented by the two functions `eigencompute10` and `eigencompute4` in the file `matrix.c`.

The two functions are identical except for their function headers which specify 10x10 and 4x4 input matrices respectively. This is for software portability with early C standards where functions cannot be defined to handle arrays with variable numbers of columns.

Contents

1	Introduction.....	
1.1	Summary	
1.2	Terminology.....	
1.3	Software Functions	
2	Eigenanalysis by Jacobi Algorithm	
2.1	Introduction.....	
2.2	Givens Rotation Matrix	
2.3	Determining the Givens Rotation Angle ..	
2.4	The Jacobi Algorithm	
3	Revision History.....	

1.2 Terminology

Symbol	Definition
A	General square matrix
A^T	Transpose of matrix A
A^{-1}	Inverse of matrix A
R_i	i -th Givens rotation matrix
R_{pq}	Givens rotation matrix with non-zero elements in row p and column q
X	Matrix of column eigenvectors
β_i	i -th eigenvector
λ_i	i -th eigenvalue
Λ	Diagonal eigenvalue matrix
ϕ	Jacobi rotation angle

1.3 Software Functions

Table 1. Sensor Fusion Library software functions

Function	Description	Section
<code>void eigencompute10 (float A[][10], float eigval[], float eigvec[][10], int8 n)</code>	Computes the eigenvalues and eigenvectors of a square matrix stored in the upper left of a 10x10 array.	2
<code>void eigencompute4 (float A[][4], float eigval[], float eigvec[][4], int8 n)</code>	Computes the eigenvalues and eigenvectors of a square matrix stored in the upper left of a 4x4 array.	2

2 Eigenanalysis by Jacobi Algorithm

2.1 Introduction

The i^{th} column eigenvector β_i and eigenvalue λ_i of any square $N \times N$ matrix A are defined as satisfying:

$$A\beta_i = \lambda_i\beta_i \quad (1)$$

The $N \times N$ matrix X formed from the N individual column eigenvectors β_i is:

$$X = (\beta_0 \quad \beta_1 \quad \dots \quad \beta_{N-1}) \quad (2)$$

Equation (1) can then be written in the form:

$$AX = X\Lambda \quad (3)$$

where Λ is the $N \times N$ matrix formed from the eigenvalues lying on the diagonal:

$$\mathbf{\Lambda} = \begin{pmatrix} \lambda_0 & 0 & \dots & 0 \\ 0 & \lambda_1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \lambda_{N-1} \end{pmatrix} \quad (4)$$

If the inverse matrix \mathbf{X}^{-1} exists then equation (3) implies:

$$\mathbf{A} = \mathbf{X}\mathbf{\Lambda}\mathbf{X}^{-1} \quad (5)$$

$$\mathbf{\Lambda} = \mathbf{X}^{-1}\mathbf{A}\mathbf{X} \quad (6)$$

A diagonal matrix is unaffected by pre- and post-multiplication by any rotation matrix. Pre- and post-multiplying equation (6) by any inverse and forward rotation matrix \mathbf{R} therefore gives:

$$\mathbf{R}^{-1}\mathbf{\Lambda}\mathbf{R} = \mathbf{\Lambda} = \mathbf{R}^{-1}\mathbf{X}^{-1}\mathbf{A}\mathbf{X}\mathbf{R} \quad (7)$$

The Jacobi algorithm underlying the functions `eigencompute10` and `eigencompute4` computes the eigenvalues and eigenvectors of a symmetric (and therefore square) $N \times N$ matrix \mathbf{A} by successive pre- and post-multiplication by inverse and forward two-dimensional plane rotation matrices \mathbf{R}_i termed Givens rotation matrices designed to obtain the diagonal matrix $\mathbf{\Lambda}$:

$$\mathbf{\Lambda} = \mathbf{R}_N^{-1} \dots \mathbf{R}_2^{-1} \mathbf{R}_1^{-1} \mathbf{A} \mathbf{R}_1 \mathbf{R}_2 \dots \mathbf{R}_N \quad (8)$$

$$\Rightarrow \mathbf{\Lambda} = \mathbf{R}_N^{-1} \dots \mathbf{R}_2^{-1} \mathbf{R}_1^{-1} (\mathbf{X}\mathbf{\Lambda}\mathbf{X}^{-1}) \mathbf{R}_1 \mathbf{R}_2 \dots \mathbf{R}_N \quad (9)$$

$$\Rightarrow \mathbf{X} = \mathbf{R}_1 \mathbf{R}_2 \dots \mathbf{R}_N \quad (10)$$

The eigenvectors of a symmetric matrix are orthogonal and another way of interpreting equations (8) through (10) is that the sequence of Givens rotations matrices rotates the matrix of eigenvectors to be aligned with the coordinate system.

The eigenvalues of the matrix \mathbf{A} are then the elements of the diagonal matrix $\mathbf{\Lambda}$ derived by zeroing off-diagonal elements in \mathbf{A} and the matrix of eigenvectors \mathbf{X} is the product of the sequence of matrices \mathbf{R}_i used to perform the diagonalization.

2.2 Givens Rotation Matrix

The Givens matrix \mathbf{R}_{pq} for rotation angle ϕ has form:

$$\mathbf{R}_{pq} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \dots & \dots & \dots & \dots & \dots & 0 \\ 0 & \dots & \cos\phi & \dots & \sin\phi & \dots & 0 \\ 0 & \dots & \dots & \dots & \dots & \dots & 0 \\ 0 & \dots & -\sin\phi & \dots & \cos\phi & \dots & 0 \\ 0 & \dots & \dots & \dots & \dots & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad (11)$$

All the diagonal elements are zero except for the two elements at positions p, p and q, q . All off-diagonal elements are zero except for the two elements at positions p, q and q, p . The Givens rotation matrix is orthonormal as required for a rotation matrix.

A general matrix \mathbf{A} is transformed by pre- and post-multiplication by the Givens rotation matrix with non-zero elements in rows p and q as:

$$\mathbf{A}' = \mathbf{R}_{pq}^T \mathbf{A} \mathbf{R}_{pq} \quad (12)$$

The elements in \mathbf{A} changed by this operation are:

$$\mathbf{A}' = \begin{pmatrix} \dots & \dots & a'_{0,p} & \dots & a'_{0,q} & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a'_{p,0} & \dots & a'_{p,p} & \dots & a'_{p,q} & \dots & a'_{p,n-1} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a'_{q,0} & \dots & a'_{q,p} & \dots & a'_{q,q} & \dots & a'_{q,n-1} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & a'_{n-1,p} & \dots & a'_{n-1,q} & \dots & \dots \end{pmatrix} \quad (13)$$

The changed elements of equation (13) are:

$$a'_{r,p} = a_{r,p} \cos \phi - a_{r,q} \sin \phi \quad (r \neq p, r \neq q) \quad (14)$$

$$a'_{r,q} = a_{r,q} \cos \phi + a_{r,p} \sin \phi \quad (r \neq p, r \neq q) \quad (15)$$

$$a'_{p,p} = a_{p,p} \cos^2 \phi + a_{q,q} \sin^2 \phi - 2a_{p,q} \sin \phi \cos \phi \quad (16)$$

$$a'_{q,q} = a_{p,p} \sin^2 \phi + a_{q,q} \cos^2 \phi + 2a_{p,q} \sin \phi \cos \phi \quad (17)$$

$$a'_{p,q} = a_{p,q} (\cos^2 \phi - \sin^2 \phi) + (a_{p,p} - a_{q,q}) \sin \phi \cos \phi \quad (18)$$

2.3 Determining the Givens Rotation Angle

The required rotation angle ϕ is that which zeroes out element $a'_{p,q}$ in equation (18):

$$a'_{p,q} = 0 \Rightarrow \frac{(\cos^2 \phi - \sin^2 \phi)}{\sin \phi \cos \phi} = \frac{(a_{q,q} - a_{p,p})}{a_{p,q}} \quad (19)$$

Standard trigonometry identities allow the cotangent of twice the rotation angle (2ϕ) to be written as:

$$\cot(2\phi) = \frac{\cos(2\phi)}{\sin(2\phi)} = \frac{\cos^2 \phi - \sin^2 \phi}{2 \sin \phi \cos \phi} \quad (20)$$

Combining equations (19) and (20) defines the rotation angle ϕ as:

$$\cot(2\phi) = \frac{a_{q,q} - a_{p,p}}{2a_{p,q}} \quad (21)$$

$$\cot(2\phi) = \frac{\cos^2 \phi - \sin^2 \phi}{2 \sin \phi \cos \phi} = \frac{1 - \tan^2 \phi}{2 \tan \phi} \Rightarrow \tan^2 \phi + 2 \cot(2\phi) \tan \phi - 1 = 0 \quad (22)$$

$$\Rightarrow \tan \phi = -\cot(2\phi) \pm \sqrt{\cot^2(2\phi) + 1} \quad (23)$$

Taking the positive square root for $\tan \phi$ gives:

$$\tan \phi = -\cot(2\phi) + \sqrt{\cot^2(2\phi) + 1} \quad (24)$$

$$= \frac{(-\cot(2\phi) + \sqrt{\cot^2(2\phi) + 1})(-\cot(2\phi) - \sqrt{\cot^2(2\phi) + 1})}{(-\cot(2\phi) - \sqrt{\cot^2(2\phi) + 1})} = \frac{-1}{-\cot(2\phi) - \sqrt{\cot^2(2\phi) + 1}} \quad (25)$$

Taking the negative square root gives:

$$\tan \phi = -\cot(2\phi) - \sqrt{\cot^2(2\phi) + 1} \quad (26)$$

$$= \frac{(-\cot(2\phi) - \sqrt{\cot^2(2\phi) + 1})(-\cot(2\phi) + \sqrt{\cot^2(2\phi) + 1})}{(-\cot(2\phi) + \sqrt{\cot^2(2\phi) + 1})} = \frac{-1}{-\cot(2\phi) + \sqrt{\cot^2(2\phi) + 1}} \quad (27)$$

For θ negative, the smaller magnitude of the two solutions is:

$$\tan \phi = \frac{-1}{(-\cot(2\phi) + \sqrt{\cot^2(2\phi) + 1})} = \frac{\text{sgn}(\cot(2\phi))}{(|\cot(2\phi)| + \sqrt{\cot^2(2\phi) + 1})} \quad (28)$$

For θ positive, the smaller magnitude of the two solutions is:

$$\tan \phi = \frac{1}{(\cot(2\phi) + \sqrt{\cot^2(2\phi) + 1})} = \frac{\text{sgn}(\cot(2\phi))}{(|\cot(2\phi)| + \sqrt{\cot^2(2\phi) + 1})} \quad (29)$$

In both cases:

$$\tan \phi = \frac{\text{sgn}(\cot(2\phi))}{(|\cot(2\phi)| + \sqrt{\cot^2(2\phi) + 1})} \quad (30)$$

If θ is so large that $\cot(2\phi)$ squared would overflow, the alternative is:

$$\tan \phi = \frac{\text{sgn}(\cot(2\phi))}{2|\cot(2\phi)|} = \frac{-1}{2\cot(2\phi)} \quad (31)$$

A trigonometric identity used later is:

$$\frac{\sin\phi}{1 + \cos\phi} = \frac{2\sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\phi}{2}\right)}{2\cos^2\left(\frac{\phi}{2}\right)} = \tan\left(\frac{\phi}{2}\right) \quad (32)$$

2.4 The Jacobi Algorithm

To avoid roundoff error, iterative updates are used for equations (14) to (19).

Equation (19):

By definition, the rotation angle ϕ is selected so that equation (19) results in zero $a'_{p,q}$.

$$a'_{p,q} = 0 \Rightarrow a_{p,q}(\cos^2\phi - \sin^2\phi) + (a_{p,p} - a_{q,q})\sin\phi\cos\phi = 0 \quad (33)$$

$$\Rightarrow a_{q,q}\sin^2\phi = \frac{\sin\phi\{a_{p,q}(\cos^2\phi - \sin^2\phi) + a_{p,p}\sin\phi\cos\phi\}}{\cos\phi} \quad (34)$$

Equation (16):

Substituting equation (34) into equation (16) gives:

$$a'_{p,p} = a_{p,p}\cos^2\phi + \sin\phi\left\{\frac{a_{p,q}(\cos^2\phi - \sin^2\phi) + a_{p,p}\sin\phi\cos\phi}{\cos\phi}\right\} - 2a_{p,q}\sin\phi\cos\phi \quad (35)$$

$$= a_{p,p} + \left(\frac{a_{p,q}\sin\phi(\cos^2\phi - \sin^2\phi) - 2a_{p,q}\sin\phi\cos^2\phi}{\cos\phi}\right) \quad (36)$$

$$\Rightarrow a'_{p,p} = a_{p,p} - a_{p,q}\sin\phi\left(\frac{\sin^2\phi + \cos^2\phi}{\cos\phi}\right) = a_{p,p} - a_{p,q}\tan\phi \quad (37)$$

Equation (17):

Since the rotation angle ϕ is selected to zero $a'_{p,q}$, equation (17) can be written as:

$$a_{p,q}(\cos^2\phi - \sin^2\phi) + (a_{p,p} - a_{q,q})\sin\phi\cos\phi = 0 \quad (38)$$

$$\Rightarrow a_{p,p} = a_{q,q} - \left\{\frac{(\cos^2\phi - \sin^2\phi)a_{p,q}}{\sin\phi\cos\phi}\right\} \quad (39)$$

Substituting into equation ((17) gives:

$$a'_{q,q} = \sin^2\phi\left\{a_{q,q} - \left\{\frac{(\cos^2\phi - \sin^2\phi)a_{p,q}}{\sin\phi\cos\phi}\right\}\right\} + a_{q,q}\cos^2\phi + 2a_{p,q}\sin\phi\cos\phi \quad (40)$$

$$\Rightarrow a'_{q,q} = a_{q,q} + \sin\phi \left(\frac{\sin^2\phi + \cos^2\phi}{\cos\phi} \right) a_{p,q} = a_{q,q} + a_{p,q} \tan\phi \quad (41)$$

Equation ((14):

With trivial manipulation, equation (14) can be written as:

$$a'_{r,p} = a_{r,p} - a_{r,p}(1 - \cos\phi) - a_{r,q}\sin\phi \quad (42)$$

$$= a_{r,p} - \sin\phi \left\{ a_{r,q} + \frac{(1 - \cos\phi)(1 + \cos\phi)a_{r,p}}{\sin\phi(1 + \cos\phi)} \right\} = a_{r,p} - \sin\phi \left\{ a_{r,q} + \frac{a_{r,p}\sin^2\phi}{\sin\phi(1 + \cos\phi)} \right\} \quad (43)$$

Substituting equation (32):

$$a'_{r,p} = a_{r,p} - \sin\phi \left(a_{r,q} + a_{r,p}\tan\left(\frac{\phi}{2}\right) \right) \quad (44)$$

Equation (16):

With trivial manipulation, equation (15) can be written as:

$$a'_{r,q} = a_{r,q} - a_{r,q}(1 - \cos\phi) + a_{r,p}\sin\phi \quad (45)$$

$$= a_{r,q} + \sin\phi \left\{ a_{r,p} - \frac{(1 - \cos\phi)(1 + \cos\phi)a_{r,q}}{\sin\phi(1 + \cos\phi)} \right\} = a_{r,q} + \sin\phi \left\{ a_{r,p} - \frac{a_{r,q}\sin^2\phi}{\sin\phi(1 + \cos\phi)} \right\} \quad (46)$$

Substituting equation (32) gives:

$$a'_{r,q} = a_{r,q} + \sin\phi \left(a_{r,p} - a_{r,q}\tan\left(\frac{\phi}{2}\right) \right) \quad (47)$$

3 Revision History

Table 2. Revision history

Rev. No.	Date	Description
1	9/2015	Initial release

How to Reach Us:

Home Page:

freescale.com

Web Support:

freescale.com/support

Information in this document is provided solely to enable system and software implementers to use Freescale products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. Freescale reserves the right to make changes without further notice to any products herein.

Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address:
freescale.com/SalesTermsandConditions.

Freescale and the Freescale logo, are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. All other product or service names are the property of their respective owners.

© 2015 Freescale Semiconductor, Inc.