This algorithm was used to compute the eigenvalues of the 2D MHD system in:

H. Nishikawa, P. L. Roe, Y. Suzuki, B. van Leer,
A General Theory of Local Preconditioning and Its Application to the 2D Ideal MHD Equations, AIAA Paper 2003-3704,
16th AIAA Computational Fluid Dynamics Conference, Orlando, June 2003.

# Solving the Quartic Equations by Newton's Method

Hiroaki Nishikawa

January, 2003

In this note, I will describe an efficient way to solve a quartic equation

$$a_4\lambda^4 + a_3\lambda^3 + a_2\lambda^2 + a_1\lambda + a_0 = 0 \tag{1}$$

using Newton's method.

## 1 Algorithm

We assume that there exist two real solutions to the quartic, which applies to the 2D MHD equations (slow wave). We begin by computing one of the two real solutions by Newton's method with the initial guess taken to be the low-Mach number approximation

$$\lambda_0 = \frac{b\,\sin\alpha}{(b\,\cos\alpha + M\sqrt{1+b^2}) - \frac{M^3 b^2 \sin^2\alpha}{2\sqrt{1+b^2}\left\{b^2\sin^2\alpha + (b\,\cos\alpha + M\sqrt{1+b^2})^2\right\}}}. \tag{2}$$

This is sufficiently accurate for small Mach numbers and also a good initial guess for large Mach numbers. It has been confirmed that only a few Newton iterations are required to obtain the solution comparable to the one obtained by the exact formula.

The second step is to find another real root. We use Newton's method again, but we do not solve the quartic because it may converge to the same root $\lambda_1$ that we have already found. In order to avoid such a possibility, we apply the method to the cubic equation

$$a_4\lambda^3 + (a_3 + \lambda_1 a_4)\lambda^2 + \{a_2 + \lambda_1(a_3 + \lambda_1 a_4)\}\,\lambda + [a_1 + \lambda_1\{a_2 + \lambda_1(a_3 + \lambda_1 a_4)\}] = 0 \tag{3}$$

which is derived from dividing (1) by $(\lambda - \lambda_1)$, thus excluding the root already found. The initial guess is the other low-Mach number approximation

$$\lambda_0 = \frac{b\,\sin\alpha}{(b\,\cos\alpha - M\sqrt{1+b^2}) + \frac{M^3 b^2 \sin^2\alpha}{2\sqrt{1+b^2}\left\{b^2\sin^2\alpha + (b\,\cos\alpha - M\sqrt{1+b^2})^2\right\}}}. \tag{4}$$

This Newton iteration has also been found to converge very quickly, i.e. only a few iterations.

Now that we have found two real roots, $\lambda_1$, $\lambda_2$, we only need to solve a quadratic equation to find two remaining roots. The quadratic equation can be found by dividing (3) by $(\lambda - \lambda_1)(\lambda - \lambda_2)$. The reault is

$$a_4\lambda^2 + (a_3 + \alpha a_4)\lambda + \left\{a_2 + \alpha a_3 + (\alpha^2 - \beta)a_4\right\} = 0, \tag{5}$$

where

$$\alpha = \lambda_1 + \lambda_2, \quad \beta = \lambda_1 \lambda_2, \tag{6}$$

which is easily solved by the quadratic formula.

## 2 Remarks

This algorithm is not only much faster than the exact formula (4 - 7 times faster in CPU time), but also robust.

## A Finding Complex Roots by Newton Iterations

It is possible to compute complex roots by Newton's method. For complex roots, we break the equation into a 2x2 system and solve them for real and imaginary parts of the complex root. Let $\lambda_R$ and $\lambda_I$ be the real and imaginary parts of the root we wish to find. Then, substituting $\lambda = \lambda_R + i\lambda_I$ into (1), we get

$$f_R + i f_I = 0 \tag{7}$$

whence

$$\begin{aligned} f_R &\equiv a_4(\lambda_R^2 - 2\lambda_R\lambda_I - \lambda_I^2)(\lambda_R^2 + 2\lambda_R\lambda_I - \lambda_I^2) + a_3\lambda_I(3\lambda_R^2 - \lambda_I^2) + a_2(\lambda_R^2 - \lambda_I^2) + a_1\lambda_R + a_0 = 0 \\ f_I &\equiv 4a_4\lambda_R\lambda_I(\lambda_R^2 - \lambda_I^2)a_3\lambda_I(3\lambda_R^2 - \lambda_I^2) + 2a_2\lambda_R\lambda_I + a_1\lambda_I = 0 \end{aligned}$$

which are to be solved for $[\lambda_R, \lambda_I]$. Using the vector notation, we may write the equations as

$$\mathbf{f}(\lambda) = 0 \quad \text{where } \mathbf{f} = [f_R, f_I] \text{ and } \lambda = [\lambda_R, \lambda_I]. \tag{8}$$

Newton's method gives

$$\lambda^{k+1} = \lambda^k - \left(\frac{\partial \mathbf{f}}{\partial \lambda}\right)^{-1} \mathbf{f}(\lambda^k). \tag{9}$$

This works well.

 