Proceedings of the
American Control Conference
Baltimore, Maryland • June 1994

A continuous time power iteration method for computing a lower bound of μ^*

Yi Cheng Bart De Moor†
ESAT - Department of Electrical Engineering
Katholieke Universiteit Leuven
Kardinaal Mercierlaan 94 B-3001 Heverlee Belgium
email: cheng.yi@esat.kuleuven.ac.be, bart.demoor@esat.kuleuven.ac.be

Abstract

A lower bound of μ may be calculated via power iteration method[1]. However this algorithm sometimes suffers from a limit cycling problem. In this short paper, a continuous time power iteration scheme is proposed. The scheme is based on building a continuous time dynamic system for power iteration, and then solving it by automatic step integration methods. Numerical examples show that convergence properties are promising.

1 Introduction

For computing a lower bound of the structure singular value μ, a power iteration scheme was developed for the purely complex case in [1] (we will refer to it as the standard power iteration or SPI). The iteration scheme appears to have good convergence properties, and each iteration step of the scheme is very cheap, so that the resulting lower bound algorithm is very fast. However, the lower bound power iteration is not always guaranteed to converge because of limit cycling. In this short paper, a continuous time power iteration scheme is proposed. We construct a continuous dynamic system for the power iteration, such that if the continuous dynamic system is stable for some initial values, that is, the continuous power iteration is convergent, then the solution can be found from the steady states.

2 Characterization of lower bound of μ

Let a matrix $M \in \mathbb{C}^{n\times n}$ be given, and let the block structure contain two blocks: a repeated scalar block($s = 1$) and a full block ($f = 1$)with dimensions r and m ($r + m = n$) respectively, then the lower bound of $\mu(M)$(see [1] for the definition of $\mu(M)$), β, satisfies the following equations[1]:

$$\beta a = Mb,\ z_1 = \frac{w_1^* a_1}{|w_1^* a_1|} w_1,\ z_2 = \frac{\|w_2\|}{\|a_2\|} a_2 \quad (1.a)$$

$$\beta w = M^* z,\ b_1 = \frac{a_1^* w_1}{|a_1^* w_1|} a_1,\ b_2 = \frac{\|a_2\|}{\|w_2\|} w_2 \quad (1.b)$$

where $a = [a_1^t\ a_2^t]^t$, $b = [b_1^t\ b_2^t]^t$, $w = [w_1^t\ w_2^t]^t$ and $z = [z_1^t\ z_2^t]^t$ with nonzero vectors z_1, w_1, b_1, $a_1 \in \mathbb{C}^r$ and z_2, w_2, b_2, $a_2 \in \mathbb{C}^m$, and $w_1^* a_1 \neq 0$. Now by eliminating z and b, we can rewrite (1) as:

$$\beta \begin{bmatrix} a \\ w \end{bmatrix} = \begin{bmatrix} 0 & M \\ M^* & 0 \end{bmatrix} \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} \begin{bmatrix} a \\ w \end{bmatrix} \quad (2)$$

where $d_{11} = \mathrm{diag}(0_{r\times r}, \frac{\|w_2\|}{\|a_2\|} I_m)$, $d_{12} = \mathrm{diag}(\frac{w_1^* a_1}{|w_1^* a_1|} I_r, 0_{m\times m})$, $d_{21} = \mathrm{diag}(\frac{a_1^* w_1}{|a_1^* w_1|} I_r, 0_{m\times m})$, $d_{22} = \mathrm{diag}(0_{r\times r}, \frac{\|a_2\|}{\|w_2\|} I_m)$. By separating real and imaginary parts, it can easily be checked that (2) is equivalent to the following equation:

$$\beta \begin{bmatrix} a_r \\ a_i \\ w_r \\ w_i \end{bmatrix} = AD \begin{bmatrix} a_r \\ a_i \\ w_r \\ w_i \end{bmatrix} \quad (3)$$

where $a_r = \mathrm{real}(a)$, $a_i = \mathrm{imag}(a)$, $w_r = \mathrm{real}(w)$, $w_i = \mathrm{imag}(w)$ and

$$A = \begin{bmatrix} 0 & 0 & M_r & -M_i \\ 0 & 0 & M_i & M_r \\ M_r^t & M_i^t & 0 & 0 \\ -M_i^t & M_r^t & 0 & 0 \end{bmatrix},\ D = \begin{bmatrix} d_{11} & 0 & d_{12}^r & -d_{12}^i \\ 0 & d_{11} & d_{12}^i & d_{12}^r \\ d_{21}^r & -d_{21}^i & d_{22} & 0 \\ d_{21}^i & d_{21}^r & 0 & d_{22} \end{bmatrix}$$

with $M_r = \mathrm{real}(M)$, $M_i = \mathrm{imag}(M)$ and $d_{12}^r = \mathrm{real}(d_{12})$, $d_{12}^i = \mathrm{imag}(d_{12})$, $d_{21}^r = \mathrm{real}(d_{21})$, $d_{21}^i = \mathrm{imag}(d_{21})$. Now letting $x = [a_r^t\ a_i^t\ w_r^t\ w_i^t]^t$ and $D(x)$ emphasize the fact that D is a function of x, we obtain:

$$AD(x)x = \beta x \quad (4)$$

Basically there is no difference between (1) and (4). However the matrices and vectors are real in (4), while they are complex in (1). This formulation can be easily generalized to the cases of $s > 1$ and $f > 1$. It can be done by modifying the matrices d_{11}, d_{12}, d_{21} and d_{22}. The matrix $D(x)$ here contains all the structure information.

3 Continuous time model of power iteration

Now we consider the following continuous time dynamic system:

$$\frac{dx}{dt} = AD(x)x - \beta x, \beta = x^t AD(x)x \quad (5)$$

*This text presents research results obtained within the framework of the Belgian programme on interuniversity attraction poles initiated by the Belgian State - Prime Minister's Office - Science Policy Programming (IUAP-17 & IUAP-50). The scientific responsibility is assumed by its authors.

†Bart De Moor is a senior research associate of the N.F.W.O. (Belgian National Fund for Scientific Research).

here x is a vector function of the time t. For this dynamic system, one property is that the vector x has a constant norm of 1 if the initial value of the norm is 1. We can see this by taking the derivative of $\|x\|^2$: $d\|x\|^2/dt = 2x^t\frac{dx}{dt} = 2(1-x^tx)(x^tADx)$, so we have $d\|x\|^2/dt|_{\|x\|=1} = 0$. Thus if we take the initial value $\|x(0)\| = 1$, $\|x\|$ is constant and equal to 1. The second property is that in any equilibrium point where $dx/dt = 0$, (5)is the same as (4), and thus β is a lower bound of μ

Now we can see that a power iteration can be formed by discretizing the continuous dynamic system of (5) plus normalizing the norm of the state vector in each step. Let (5) is discretized as:

$$\frac{x_{k+1}-x_k}{\Delta t_k} = AD(x_k)x_k - x_k(x_k^tAD(x_k)x_k) \qquad (6)$$

where Δt_k is the integration step at time t_k and x_k is the vector x at time t_k. Suppose $x_k^tAD(x_k)x_k > 0$ and let $\Delta t_k = 1/(x_k^tAD(x_k)x_k)$, then by normalizing the norm of x_{k+1} to 1, (6) can be arranged as

$$x_{k+1} = \frac{AD(x_k)x_k}{\|AD(x_k)x_k\|} \qquad (7)$$

The power iteration (7) is actually the same as SPI (see [1] for details of SPI), if some intermediate results in each iteration are not used and if $\beta_{k+1} = \|AD(x_k)x_k\|$. Note that if $x_k^tAD(x_k)x_k < 0$ the above explanation does not hold. However it can be proved that if the continuous time power iteration is convergent, $x^tAD(x)x > 0$ always near an equilibrium point. Though numerical simulation of the continuous time dynamic system is also discrete, it is different from SPI, as the former can choose integration steps according to the desired accuracy by, for example, automatic step Runge-Kutta integration, the latter only uses a special integration law of (6), for which the integration steps can be possibly very large. One result is that limit cycling in SPI could be eliminated in the continuous time power iteration, which will be shown by an example later.

4 Numerical examples

We simulated the continuous dynamic system of (5) using MATLAB function 'ode23', which uses the automatic step Runge-Kutta-Fehlberg integration method. The desired accuracy of the solution is set to 10^{-5}. For most of examples we tested with nonzero initial vectors a_r, a_i, w_r and w_i, the simulations were convergent, by which we mean that both the state vector x and the scalar β evolve to constant values, even in the case where SPI has limit cycling. Here is one of the examples. The matrix M is generated in random, by MATLAB 'rand' with format 'normal', as:

$$M = \begin{bmatrix} 1.0308 & 0.7611 & -0.3225 \\ -0.7599 & -0.1659 & -0.3684 \\ 0.8741 & 0.3009 & 1.1479 \end{bmatrix}$$

the block structure consists of one repeated scalar block with dimension $r = 2$ and one full block with dimension $s = 1$. We first use SPI to calculate the lower bound of μ with the MATLAB function 'mu' [2]. The initial values of the vectors b and w are the same and generated randomly as: $-0.7896 - 0.0098i$, $0.5648 - 0.0515i$, $0.0321 - 0.2317i$. For this special case, limit cycling appears during SPI, as shown in the upper

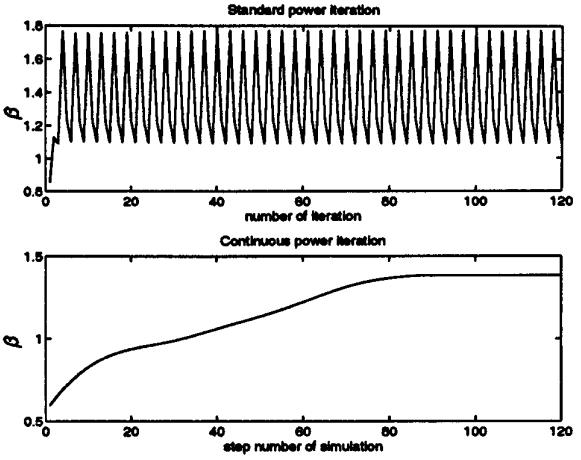

Figure 1: Upper plot: Standard power iteration. Limit cycling occurs in iterations. Bottom plot: Continuous power iteration. β evolves to an equilibrium point 1.384 in 102 steps

plot of Fig.1. Now we calculate β by solving the differential equation (5) with the MATLAB function 'ode23'. The matrix M is exactly the same as above. The initial value of the vector x is calculated from initial values of the vectors b and w via (1). That is, we use the same initial values as those for the power iteration as shown above. We do this for comparison of the two methods. The bottom plot of Fig. 1 shows the simulation result. β now evolves rather smoothly and converges to an equilibrium value 1.384 in 102 steps (the number of all forward and backward steps are 306).

5 Conclusion

From numerical experiences, we found that the convergence properties of the continuous time power iteration are good, even in the case where the limit cycling happens in SPI. However the results presented here are very preliminary. The stability property of the continuous time dynamic system of the power iteration is not proved yet for a general block structure. It is a subject of current ongoing research. Actually we have found that in a special case where the Jacobian matrix of (5) at a certain equilibrium point has purely imaginary eigenvalues, then limit cycling could happen for some initial values. The continuous time power iteration is in general slower than SPI (if SPI is convergent), so it is suggested only to use the continuous time power iteration in the case where SPI is not convergent or is very slow.

References

[1] Andy Pachard, Michael K.H.Fan and John Doyle, "A power method for the structured singular value", *Proc. 27th Conf. Decision Contr.*, Austin, Texas, Dec. 1988, pp. 2132-2137.

[2] G.J. Balas, J.C. Doyle, K.Glover, A. Packard and R. Smith, *μ-Analysis and Synthesis Toolbox*, The Math Works, Inc. 1991.