Pole Assignment Method Based on Numerical Computation with Guaranteed Accuracy

著者	Yano Kentaro, Koga Masanobu
journal or publication title	SICE 2007 proceedings
page range	2960-2965
year	2008-01-07
URL	http://hdl.handle.net/10228/00006398

doi: info:doi/10.1109/SICE.2007.4421498

Pole Assignment Method Based on Numerical Computation with Guaranteed Accuracy

Kentaro Yano[1] and Masanobu Koga[1]

[1]Kyushu Institute of Technology, 680-4 Kawazu Iizuka Fukuoka, Japan
(Tel : +81-948-29-7726; yano@mk.ces.kyutech.ac.jp, koga@ces.kyutech.ac.jp)

Abstract: This paper presents a pole assignment method based on a numerical computation with guaranteed accuracy. By using the proposed methods, it is possible to guarantee a numerical quality about design of control system. This paper also proposes a problem which finds the best controller from a set of solutions solved by verified pole assignment method. That problem is solved by using Genetic Algorithms.

Keywords: Design of control system, numerical computation with guaranteed accuracy, pole assignment problem, Genetic Algorithms

1. INTRODUCTION

A computer is used for various areas now, but it was devised that a real number is approximated to a floating point number and computed to compute fast. So, a numerical computation using a computer cannot solve a mathematical strict solution because of a round error etc. Many mathematicians and researchers took notice of this error, but it was rarely estimated how much an error is big correctly until recently.

But recently, **Numerical Computation with Guaranteed Accuracy**[1] which computes an error occurred by a numerical computation using a computer, and proves how much a computational result is correct was appeared. In the area which is wanted to compute solutions closely, there are great hopes that apply this computational technique to it.

A developing of a control system is divided into 5 phases which are modeling of a plant, characteristics analysis, design of a controller, simulation, and experiment. After modeling, a controller is designed by using that model, and design of a controller is also computed using a computer. For a reason of failing a design, it is thought of as the model is wrong, the design method is wrong, or there is a numerical problem like error etc. If modeling or design methods is wrong, it may be possible to get expected results by remodeling or redesign. Changing an algorithm etc. is thought of as an answer in an issue of a numerical computation like an error, however design methods with guaranteed numerical quality has not been built up.

Recently, methods are proposed which apply a numerical computation with guaranteed accuracy to design of a control system to solve this issue [2], [3]. For example, a method which applies a computation with guaranteed accuracy using rational numbers to design of a control system, and solve H_2 design problem for a single input single output control plant was proposed[3]. However, there are problems about a method using rational numbers shown below.

1. It cannot support irrational numbers.
2. There are impossible arithmetic operations since the result of an arithmetic must be closed under an arithmetic by rational numbers.
3. There is a problem about a computational speed and a memory size since an arithmetic of rational numbers may explode.

The purpose of this research is apply a numerical computation with guaranteed accuracy to a computation about design of a control system, and design a control system from the viewpoint of a guarantee of a numerical quality. We guaranteed the quality about a computation of design of control system, especially a numerical result about pole assignment problem.

In Section 2, the principle of a numerical computation with guaranteed accuracy, and a creation of an interval by the switching the CPU rounding mode to compute a fast numerical computation with guaranteed accuracy using floating point numbers is explained. In Section 3, we propose a pole assignment method based on a numerical computation with guaranteed accuracy. In Section 4, **JCGA(Java Computing Guaranteed Accuracy)** which is the developed Java numerical computation with guaranteed accuracy package is described. In Section 5, an example of pole assignment method based on numerical computation with guaranteed accuracy using JCGA is explained. Finally, in Section 6, conclusions are described.

2. NUMERICAL COMPUTATION WITH GUARANTEED ACCURACY

2.1 Principle of numerical computation with guaranteed accuracy

A general form of the principle and the method to solve a problem with guaranteed is explained here[4].

Now, a problem is given as

Find x which satisfies $f(x) = 0$

If an accuracy of an approximation is good, it is hoped that the true solution exists near by the approximate solution of the above problem obtained by a numerical computation. Then, consider a method to check whether the set(candidate set) which contains the approximate solution contains the true solution. If this method is obtained, it is possible to try to specify a set which contains the true solution by the procedure like following.

1. By using some method, resolve a candidate set which is expected that it contains the true solution .
2. Verify if the candidate set actually contains the true solution.
3. If it doesn't contain the true solution, create other candidate set and go back to the previous step.

If a set is specified, the size of the set gives the accuracy of the approximate solution.

The most popular mathematical tool to check whether a set contains a solution of the problem $f(x) = 0$ is an equivalent theorem about an existence of the solution of fixed point equation $x = F(x)$, that is fixed point theorem. There are several types about fixed point theorem, so a generic representation of fixed point theorem is shown below.

Fixed point theorem

Consider a F(U) as

$$F(U) = \{v \mid v = F(u), u \in U\}$$

where U is a set which satisfies some condition. Then, if set U satisfies

$$F(U) \in U$$

or, U satisfies a similar condition, the true solution of fixed point equation $x = F(x)$ exists within the U (especially exists within the $F(U)$).

2.2 Fast creating of interval by switching CPU rounding mode

A method to search a candidate set by the switching CPU rounding mode is explained here.

The CPU rounding mode which is called **round to nearest**(rounds to the floating point number that is the nearest to the real number r) is traditionally used when we compute using a computer. Other rounding modes are **round downward**(round towards the floating point number that is the biggest number fewer than the real number r), and **round upward**(round towards the floating point number that is the smallest number larger than the real number r) etc[5].

In general, it is impossible to solve the true solution itself in a numerical computation using floating point numbers. Then, a numerical computation with guaranteed accuracy switches the CPU rounding mode, computes the infimum and the supremum of the true solution, and creates the interval[6][1] as shown in Fig. 1. And, it wraps around the true solution by the interval which has the floating point number as both ends. By using this method, it is possible to search the interval which contains the true solution. Numerical computation with guaranteed accuracy is executed by the simple idea that an interval wraps around a true solution of a problem using an interval.

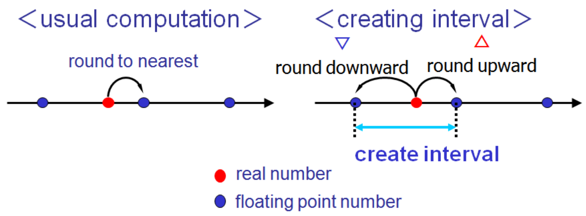

Fig. 1 Difference between traditional computation and numerical computation with guaranteed accuracy

3. POLE ASSIGNMENT METHOD BASED ON NUMERICAL COMPUTATION WITH GUARANTEED ACCURACY

3.1 Verified Pole Assignment method

Whether poles of a closed loop which use an obtained feedback gain correspond with specified poles is the most important for pole assignment problem[7]. Some problems assign poles far from specified poles, so the verifying where poles are assigned is needed. Then, we apply guaranteed accuracy of eigen value[8] to pole assignment problem and propose **verified pole assignment method** which verifies the solution obtained by pole assignment method. This method computes the eigen value of $A-BF$ with guaranteed accuracy about obtained feedback gain F. At this time, eigen values are solved as an circular complex interval. If specified poles exist within that guaranteed accuracy of eigen value, that is to say, specified poles exist within the circular disk which consists of the radius and the center about eigen value of $A - BF$ shown like Fig. 2(a). Then eigen values with guaranteed accuracy obtained by that feedback gain contain specified poles. And it is possible to solve the feedback gain with guaranteed the existence area of specified poles. This feedback gain is called **feedback gain with guarantee of quality**. And if it doesn't contain shown like Fig. 2(b), it is impossible to specify the existence area, so that solution is the solution which doesn't verify the existence area. That feedback gain is called **feedback gain without guarantee**.

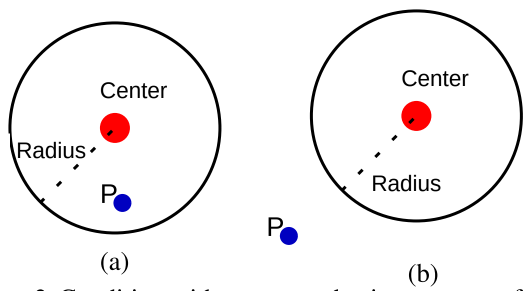

Fig. 2 Condition with guaranteed existence area of specified pole and Condition without guarantee

3.2 Verified Best Pole Assignment Problem

In the previous section, the method that verifies the existence area of specified poles about the obtained feedback gain was shown. After executing verified pole assignment method, if the obtained feedback gain is a feedback gain without guarantee, how to find a feedback gain with guarantee of quality? In this section, a method to find feedback gain with guarantee of quality, **verified best pole assignment problem** is described. Verified best pole assignment problem is the problem that pulls out a set of feedback gain with guarantee of quality(it has n feedback gains) from the interval which contains the true value of the feedback gain, and finds the feedback gain which makes a value of evaluation formula

$$error_n = \max_{1 \leqq i \leqq l} \left(\frac{|P_i - \mathrm{Mid}(E)_i| + \mathrm{Rad}(E)_i}{|P_i|} \right) \quad (1)$$

the smallest. Where, P is a specified pole, $\mathrm{Mid}(E), \mathrm{Rad}(E)$ is the center and the radius of the eigen value. Also, here are l specified poles. The part of the numerator of this evaluation formula computes the sum of the distance from the center of the eigen value to the specified pole, and the radius. The ideal state of pole assignment problem that the pole assigned by the obtained feedback gain completely correspond with the specified pole, so if parts of the numerator is small, it is thought of as the good solution is obtained. And, the denominator is the specified pole. The case that a pole is far from the imaginary axis, the pole which is near the imaginary axis is important since the convergence of the state is fast. Then, it makes a weighting by normalization at the origin. The biggest value from the obtained value is the evaluated value of the feedback gain. And, the feedback gain which makes the supremum of this evaluated value the smallest is the **best feedback gain with guarantee of quality**.

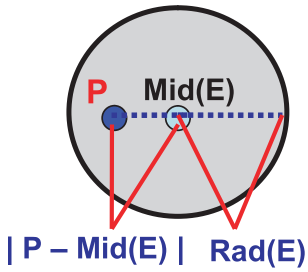

Fig. 3 Schematic of numerator of evaluation formula for verified best pole assignment problem

A solve procedures of verified best pole assignment problem is shown below.

Solve procedures of verified best pole assignment problem

Step1 Compute the interval which contains the true value of a feedback gain, and pull out from a candidate set of the feedback gain with guarantee of quality

Step2 Execute verified pole assignment method

Step3 If it is a feedback gain with guarantee of quality in Step 2, compute the evaluated value by the evaluation formula(Eq.(1)).

Step4 Make the feedback gain that supremum of the evaluated value the smallest is the verified best feedback gain.

3.3 Creating a candidate set by using GA

This paper searches the set which contains the best feedback gain from the interval which contains the true value of the feedback gain by using genetic algorithm(GA)[9].

Since an interval feedback gain is an interval matrix, it is necessary to select a floating point from an interval of all elements of a matrix to create an element of a candidate set. So, there is a difficult case to full search because of the width of an interval or element counts of an matrix. Then, this paper proposes that searching a candidate set by using GA which is a one of heuristic.

This paper proposes that make a floating point in the interval feedback gain as individual, use the Eq.(1) for the computation of fitness, use UNDX method for the crossover, and use a method by uniform random number for mutation. It is possible to search a candidate set from an interval feedback gain in a systematic way by using GA.

4. IMPLEMENTATION OF THE PROPOSED METHOD

4.1 Numerical computation with Guaranteed Accuracy package JCGA

We developed JCGA(Java Computing Guaranteed Accuracy) as a Java package of a numerical computation with guaranteed accuracy. JCGA is able to compute interval arithmetic, interval matrix arithmetic[1][6], linear equation[10][6], eigen value problem[8], polynomial[10], differential[1][10][11], and nonlinear equation[1][6][11] of numerical computation with guaranteed accuracy. And, JCGA use NFC[12] which is a Java numerical computation package for basic operations like matrix operation, or complex number operation etc. Java only use round to nearest mode[13], and cannot control the CPU rounding mode directly. So, JCGA controls the CPU rounding mode by the calling code of the rounding control written in C language by using JNI[14].

4.2 Architecture of JCGA

Figure 4 shows the architecture of JCGA. The architecture of JCGA consists of JCGA component which is the main component and executes a computation with guaranteed accuracy, NFC component which computes internal basic operations of JCGA, and libFPUNative.so(for Linux, or FPUNative.dll for Windows) component which controls the CPU rounding mode.

JCGA has round package to switch the CPU rounding mode, interval package to carry out interval arithmetic and interval matrix arithmetic, linear, eigen, polynomial, derivative, nonlinear package to compute each problem with guaranteed accuracy, and pset[15], [16] package to manage a processor on a multiprocessor environment. pset package is used to restrain a switch of CPU rounding mode by other process during a computation with guaranteed accuracy.

Classes shown in Table 1 are prepared as main classes to compute with guaranteed accuracy.

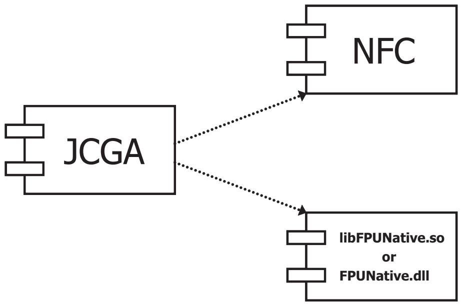

Fig. 4 Architecture of JCGA

Table 1 Classes of numerical computation with guaranteed accuracy

Numerical computation with guaranteed accuracy	Class
Interval arithmetic	Interval
Interval matrix arithmetic	IntervalMatrix
Linear equation	LinearVerifier
Eigen value problem	EigenVerifier
Polynomial	HornerVerifier
Differential value	IntervalDerivative
Nonlinear equation	NonLinearVerifier

4.3 Numerical example using JCGA

This section shows a numerical computation with guaranteed accuracy of eigen value as a numerical example using JCGA.

JCGA provides EigenVerifier class to execute a numerical computation with guaranteed accuracy of eigen value. By using this class, computes guaranteed accuracy of eingen value of the following matrix.

$$A = \begin{bmatrix} -4 & 17 & 60 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

The true value of eigen value is $\lambda = 4, -3, -5$.

The program and the computation result are shown below.

Example program of eigen value with guaranteed accuracy

```
public class EigenVerifierSample {
  public static void main(String[] args) {
    //Matrix A
    Matrix A = new RealMatrix(new double[][]{
      {-4, 17, 60},
      { 1,  0,  0},
      { 0,  1,  0}});
    //Creates a numerical computation with
    //guaranteed accuracy object
    //of eigen value
    EigenVerifier ev = new EigenVerifier();
    //Computes eigen value
    //with guaranteed accuracy
    IntervalMatrix ans = ev.solve(A)[0];
    //Display of the center and the radius
    ans.printMidRad("Eigen value");
  }
}
```

Result of eigen value with guaranteed accuracy

```
=== Center of eigen value(3 x 1) CoMatrix ===
        [ (  1)-Real ]
(  1) 4.000000000000000000E+00
(  2)-3.000000000000000400E+00
(  3)-5.000000000000000000E+00
        [ (  1)-Imag ]
(  1) 0.000000000000000000E+00
(  2) 0.000000000000000000E+00
(  3) 0.000000000000000000E+00
=== Radius of eigen value(3 x 1) Matrix ===
            (  1)
(  1)2.329383478607396300E-16
(  2)1.437330003186217900E-15
(  3)1.258978188088645200E-15
```

The above result shows that true solution exist within the circular disk which consists of the radius and the center. And, the existence interval of the solution is computed by using the above result.

Existence interval of eigen value

```
=== Infimum (3 x 1) Matrix ===
            (  1)
 (  1)  3.999999999999999600E+00
 (  2) -3.000000000000002000E+00
 (  3) -5.000000000000002000E+00
=== Supremum (3 x 1) Matrix ===
            (  1)
 (  1)  4.000000000000001000E+00
 (  2) -2.999999999999998700E+00
 (  3) -4.999999999999998000E+00
```

It is confirmed that the true solution is wrapped around between the infimum and the supremum by this result, so the guaranteed accuracy result is validate.

5. NUMERICAL EXAMPLE OF VERIFIED POLE ASSIGNMENT PROBLEM

Consider a following system, and specified poles are $P = -1, -3$.

$$A = \begin{bmatrix} 4 & -3 \\ 2 & -1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

And, verify the approximate value of the feedback gain about this problem.

The approximate value of the feedback gain is

```
 ===  F  (   1  x   2)  Matrix  ===
            (  1)
(  1)-9.666666666666666000E+00
            (  2)
(  1) 7.000000000000000000E+00
```

Then, computes the eigen value of $A - BF$ with guaranteed accuracy.

```
=== Center of eigen value(2 x 1) CoMatrix ===
        [ (  1)-Real ]
(  1)-9.999999999999987000E-01
(  2)-2.999999999999999600E+00
        [ (  1)-Imag ]
(  1) 0.000000000000000000E+00
(  2) 0.000000000000000000E+00
=== Radius of eigen value(2 x 1) Matrix ===
            (  1)
 (  1)  1.093957329412127000E-14
 (  2)  1.209323584016177600E-14
```

And, shows the result by the interval of the infimum and the supremum.

```
=== Infimum ( 2 x 1) Matrix ===
          (  1)
(  1) -1.000000000000009800E+00
(  2) -3.000000000000012000E+00
=== Supremum ( 2 x 1) Matrix ===
          (  1)
(  1) -9.999999999999877000E-01
(  2) -2.999999999999987000E+00
```

The above result shows that specified poles are wrapped around the area of the guaranteed accuracy of eigen value of $A - BF$ by pole assignment by this feedback gain.

Thus, this feedback gain verifies the existence area of specified poles, and it is a feedback gain with guarantee of quality.

Next, consider a following system, and specified poles are $P = -1, -5, -10, -15, -20, -25$.

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 50000 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Find a high-quality feedback gain of this problem. This problem is a very sensitive problem since the condition number of A is 50000.

First, computes the feedback gain by the usual approximate computation[1].

```
===Approximate solution of F(1 x 6) Matrix===
          (  1)
(  1)-9.639806218498292000E-01
          (  2)
(  1) 8.509152983176330000E+00
          (  3)
(  1)-2.418811769229065300E+01
          (  4)
(  1) 2.785936334204669000E+01
          (  5)
(  1)-1.128626546904882800E+01
          (  6)
(  1) 5.060212104148602000E+04
```

The eigen value of $A - BF$ using this approximate feedback gain is shown below.

```
=== Eigen value (6 x 1) CoMatrix ===
        [ (  1)-Real ]
(  1) 5.556514690020053500E+00
(  2) 5.556514690020053500E+00
(  3)-4.736678870134883000E-01
(  4)-6.430407250185999000E-01
(  5)-6.430407250185999000E-01
(  6)-5.964044740710399000E+02
        [ (  1)-Imag ]
(  1) 1.015862498610436800E+01
(  2)-1.015862498610436800E+01
(  3) 0.000000000000000000E+00
(  4) 2.810712219574160300E+00
(  5)-2.810712219574160300E+00
(  6) 0.000000000000000000E+00
```

The above result doesn't correspond to specified poles at all. So, we cannot use this feedback gain computed approximately.

Then, solve the verified best pole assignment problem about this problem.

[1]It uses a method from controllability companion form shown in [7] etc. And, it doesn't use any special optimization.

Step1：Computes the feedback gain of this problem, and the interval of the feed back gain is

Interval of feedback gain

```
=== Infimum ( 1 x 6) Matrix ===
          (  1)
(  1) -9.609792195860576000E-01
          (  2)
(  1)  8.482659306355298000E+00
          (  3)
(  1) -2.411280676849047200E+01
          (  4)
(  1)  2.777262180964545400E+01
          (  5)
(  1) -1.125112511255948200E+01
          (  6)
(  1)  5.009106962998430000E+04
=== supremum ( 1 x 6) Matrix ===
          (  1)
(  1) -9.609792195827667000E-01
          (  2)
(  1)  8.482659306389033000E+00
          (  3)
(  1) -2.411280676832195300E+01
          (  4)
(  1)  2.777262180984378800E+01
          (  5)
(  1) -1.125112511246289800E+01
          (  6)
(  1)  5.009106962998450000E+04
```

And, create a candidate set by using GA. We set parameters as population size is 60, number of generation is 100, number of elite is 10, crossover rate is 100%, value of SD is 0.5, mutation rate is 25%, value of M is $\frac{1}{10}$ of numbers of pieces of point within interval. Then, compute the eigen value of $A - BF$ with guaranteed accuracy using that feedback gain.

Step2：There are 7 feedback gains with guaranteed accuracy.

Step3：We compute the evaluated value about 7 feedback gains based on the evaluation formula.

Step4：The below feedback gain makes the evaluated value the smallest.

Best feedback gain with guarantee of quality

```
=== F ( 1 x 6) Matrix ===
          (  1)
(  1) -9.609792195844106000E-01
          (  2)
(  1)  8.482659306372607000E+00
          (  3)
(  1) -2.411280676840988600E+01
          (  4)
(  1)  2.777262180975247000E+01
          (  5)
(  1) -1.125112511251553000E+01
          (  6)
(  1)  5.009106962998439000E+04
```

And, the smallest evaluated value is

$$error = [5.432948863478505E{-}10, 5.432948863478507E{-}10]$$

The eigen value of the best feedback gain with guarantee of quality is

```
=== Center of eigen value(6 x 1) CoMatrix ===
           [ (   1)-Real ]
(   1)-1.000000000002854600E+00
(   2)-5.000000000033347000E+00
(   3)-9.999999999778545000E+00
(   4)-1.500000000168764200E+01
(   5)-1.999999999991897000E+01
(   6)-2.499999999954850000E+01
           [(   1)-Imag ]
(   1) 0.000000000000000000E+00
(   2) 0.000000000000000000E+00
(   3) 0.000000000000000000E+00
(   4) 0.000000000000000000E+00
(   5) 0.000000000000000000E+00
(   6) 0.000000000000000000E+00
=== Radius of eigen value  (6 x 1) Matrix ===
                (   1)
(   1)  7.627531467764463000E-12
(   2)  3.508607539019007000E-10
(   3)  3.432007441453436400E-09
(   4)  6.461781269090835500E-09
(   5)  1.066656583505956800E-08
(   6)  4.605143259111369500E-09
```

The infimum and the supuremum of the interval is

```
=== Infimum of eigen value(6 x 1) Matrix ===
                (   1)
(   1) -1.000000000010482300E+00
(   2) -5.000000000384208000E+00
(   3) -1.000000000321055400E+01
(   4) -1.500000000814942400E+01
(   5) -2.000000001058553800E+01
(   6) -2.500000000415364500E+01
=== Supremum of eigen value(6 x 1) Matrix ===
                (   1)
(   1) -9.999999999952270000E-01
(   2) -4.999999999682485000E+00
(   3) -9.999999996346537000E+00
(   4) -1.499999999522586000E+01
(   5) -1.999999998925240000E+01
(   6) -2.499999999494335500E+01
```

The above result shows that interval of the eigen value of $A - BF$ contains specified poles. The time to solve this problem is shown as Table 2. Computation environment is OS: Windows XP Professional, CPU: Pentium D 945(3.4GHz), Memory: 1024MB.

Table 2 Time to compute example

Step	Time(ms)
Step1(Computation of interval F + select candidate of best F)	151273(156+151117)
Step2(Verified pole assignment method)	1070
Step3(Evaluation of best feedback gain with guarantee of quality F)	32
Total	152375

6. CONCLUSIONS

This paper developed JCGA which is a Java numerical computation with guaranteed accuracy package, and applied numerical computation with guaranteed accuracy to computational problems about design of control system, especially guarantee a quality about pole assignment problem. It is possible to compute the computational result with the significantly-improved numerical quality by applied numerical computation with guaranteed accuracy to a sensitive problem which is difficult to solve by an approximate computation. We would like to go on to propose design methods for another design problems like LQR control[7] etc. from the viewpoint of the guarantee of numerical quality.

REFERENCES

[1] Shinichi Oishi. *Verified Numerical Computation*. CORONA PUBLISHING CO., LTD., 2000.

[2] Masanobu Koga and Akinori Hiroki. Control System Design based on Numerical Computation with Guaranteed Accuracy, 2005. SICE 5th Annual Conference on Control Systems pp.707-710.

[3] Masaaki Kanno and Shinji Hara. Guaranteed Accuracy Algorithm in H2 Optimal Tracking Controller Synthesis. *Transactions of the Society of Instrument and Control Engineers*, Vol. 43, No. 2, 2007.

[4] Mitsuhiro Nakao and Nobito Yamamoto. *Nmerical Computation with Guaranteed Accuracy*. NIPPON HYORONSHA CO.,LTD.PURBLISHERS, 1998.

[5] ANSI/IEEE. IEEE Standard for Binary Floating Point Arithmetic, 1985. Std 754-1985 edition, New York.

[6] G.I.Hargreaves. *Interval Analysis in MATLAB*, 2002. Numerical Analysis Report No.416.

[7] William L. Brogan. *Modern Control Theory Third Edition*. Prentice-Hall International, Inc., 1981.

[8] Siegried M. Rump. Computational error bounds for multiple or nearly multiple eigenvalues. *Linear Algebra and its Applications*, Vol. 324, pp. 209–226, 2001.

[9] Juraj Hromkovic. *Algorithmics for Hard Computing Problems:Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics*. Springer, 2001.

[10] Ohishi Shinichi. *Numerical Calculation*. SHOKABO PUBLISHING Co., Ltd., 1999.

[11] Shinichi Oishi. *Tools for Numerical Computation on Linux*. CORONA PUBLISHING CO., LTD., 2000.

[12] Masanobu Koga and Takeshi Matsuki. Development of OS-Neutral Numerical Foundation Class Library and its Application to Control System, 2003. SICE 3th Annual Conference on Control Systems pp.725-728.

[13] Tim Lindholm and Frank Yellin. *The Java Virtual Machine Specification, Second Edition*. Addison-Wesley Pub, 1999.

[14] Rob Gordon. *Essential JNI: Java Native Interface*. Prentice Hall Ptr, 1998.

[15] Masanobu Koga and Kiminori Takei. Parallel Computation guaranteed Accuracy using Processor Set. 21th Simulation Technology Conference, pp.199-202 (2002).

[16] Masanobu Koga, Satoko Tsuneoka, and Akinori Hiroki. Parallel Computing Guaranteed Accuracy using Java. FIT2003 3rd Forum on Information Technology, pp.37-39 (2003).