

# AN ALGORITHM FOR EXTRACTION OF BOUNDARY RECTANGLES FROM INTERVAL TEMPLATES

P. S. V. NATARAJ, R. SATYANARAYAN, AND S. SHEELA

*Systems and Control Engineering  
IIT Bombay 400 076 India  
Email: nataraj@ee.iitb.ernet.in*

**ABSTRACT.** We present an algorithm for extracting the boundary rectangles from interval templates. We test and compare the performance of the proposed algorithm versus those of three boundary extraction algorithms available in the QFT literature. We perform the testing on a benchmark suite of eleven transfer function examples. In terms of computational time and effort (*flops*), the test results show the proposed algorithm to be the most efficient in every example. On an average, the improvement in terms of these measures is by 2 – 3 orders of magnitude.

## 1. INTRODUCTION

Over the last few decades, Horowitz's quantitative feedback theory (QFT) approach [5] to robust control system design has been gaining popularity among control researchers. The QFT approach comprises of a collection of techniques for dealing with several classes of uncertain plants: linear and nonlinear, time-invariant and time-varying, lumped and distributed, single input-output and multi input-output, single-loop and multiple-loop, etc.

The first step in the QFT procedure is to generate the *template* of the given plant at each design frequency. A template is a set of points in the angle-magnitude plane (i.e., in the Nichols chart) representing the response of the system at the given frequency. More precisely, consider a system represented by the transfer function  $g(s, \lambda)$ , where  $\lambda = \{\lambda_1, \dots, \lambda_n\}$  is a real vector of the system parameters and  $s$  is the Laplace variable. Suppose the parameters  $\lambda_i$  vary independently over given real intervals  $\Lambda_i^0$ , so that we have a box  $\Lambda^0 = \{\Lambda_1^0, \dots, \Lambda_n^0\}$  of system parameters. Denote the phase angle and magnitude functions of  $g(s, \lambda)$  as  $f_{ang}(\omega, \lambda) = \arg g(j\omega, \lambda)$ ;  $f_{mag}(\omega, \lambda) = |g(j\omega, \lambda)|$ , where  $\omega$  is a given frequency. Define the angle-magnitude function  $f$  as  $f(\omega, \lambda) = (f_{ang}(\omega, \lambda), f_{mag}(\omega, \lambda))$ . Then, the set  $\mathcal{G}(\omega) := \{f(\omega, \lambda), \lambda \in \Lambda^0\}$  defines a region in the angle-magnitude plane, called the template of  $g(s, \lambda)$  at the given  $\omega$ . We assume in this work that the template is closed and simply connected. Several methods are available for generating the templates, see, for example, [4], [9] and the references cited therein.

In this work we consider the so-called *interval* templates. An interval template is generated with interval analysis tools [8], using what are called as *interval extensions* of the magnitude and phase functions. A simple to use interval extension is the *natural* interval extension. A natural interval extension is obtained for each of these functions by replacing *real* variables with *interval* variables and replacing *real arithmetic* with *interval arithmetic*. We denote the natural interval extension of  $f(\omega, \lambda)$  to  $\Lambda$  as  $F(\omega, \Lambda)$ .

One can compute  $F(\omega, \Lambda)$  and obtain with a single evaluation of  $F$ , an interval template comprising of a single angle-magnitude rectangle. By inclusion property of natural interval extensions [8, Theorem 3.1], the interval template encloses the actual template  $\mathcal{G}(\omega)$ . However, this interval template with just one angle-magnitude rectangle usually has a width that considerably exceeds the prescribed accuracy (in the form of the specified rectangle width  $\varepsilon$ ). Therefore, we may repeatedly subdivide (or partition) the parameter box, find the evaluations of  $F$  over the subboxes using interval arithmetic, and take the union of the results to get interval templates comprising of smaller and smaller angle-magnitude rectangles which give increasingly accurate information about the actual phase-magnitude values. It is a fundamental result in interval analysis that as the partition of the parameter box is refined, these interval templates will converge to the actual template. The partition or subdivision process can be stopped when the widths of all the angle-magnitude rectangles in the interval template is less than the prescribed accuracy  $\varepsilon$ .

There are several interval template generation algorithms based on these ideas, see [10] for a survey. The interval template generation algorithms provides several guarantees: the generated interval template is (a) guaranteed to be of prescribed accuracy, (b) guaranteed to be reliable - that is, an assurance of correctness of the computed numerical results is provided, in face of all kinds of computational errors, such as round-off, truncation,

and approximation, (c) guaranteed to enclose all actual template points, thereby avoiding any loss of robustness due to template approximation errors.

The number of points in the plant template dictates the computation burden of the QFT design technique. In QFT designs, only the boundary of a simply connected template is necessary (see [1], [2], [7]), to which only the boundary rectangles of an interval template contribute. The aim of this note is to present an algorithm that can efficiently extract all the boundary rectangles from a given interval template, without introducing any kind of boundary approximations.

## 2. A BOUNDARY EXTRACTION ALGORITHM

We can obtain the set of boundary rectangles of a given interval template as the union of upper, lower, left and the right boundary rectangles in the Nichols chart. By upper boundary rectangles, we mean the template rectangles with the maximum magnitude at each phase, and likewise for the lower, left and the right boundary rectangles. We describe the method for extracting the upper boundary rectangles.

### Algorithm for extraction of upper boundary rectangles

- Inputs : The interval template whose boundary rectangles are to be extracted.
- Output: The set of all upper boundary rectangles of the given interval template.

#### BEGIN Algorithm

1. Mark the rectangle forming the left end of the template as a boundary rectangle. Set it as *current* rectangle.
2. Journey along the top edge of the current rectangle towards its right end. Does an upward jump to a higher magnitude occur on this journey ?
  - (a) **IF yes**, mark the rectangle associated with the jump as a boundary rectangle. Set it as *current* rectangle. Take the jump along the left side of the current rectangle to reach its top edge. Then, go back to Step 2.
  - (b) **IF no**, complete the journey to reach the right end of the top edge. If the right end of the entire template itself is thereby reached, go to step 3, else journey downward along the right side of current rectangle to the next lower magnitude rectangle, mark the latter rectangle as a boundary rectangle, and set it as *current* rectangle. Go back to Step 2.
3. Output all the boundary rectangles obtained. Exit algorithm.

#### END Algorithm.

The algorithm for extraction of lower boundary rectangles is identical to that for upper boundary rectangles, except that while journeying from left to right of the current rectangle, a check for any downward jump to a lower magnitude is made. Similarly, we can extract the left and right boundary rectangles, simply by interchanging the roles of magnitude and phase in an obvious way in the above procedures. Finally, we can obtain the set of all boundary rectangles of the interval template by taking the union of all these upper, lower, left and right boundary rectangles.

## 3. TEST RESULTS

First, to check the capability of the proposed algorithm, we consider the two arbitrary interval template shapes in Figure 3.1a. The boundary results obtained with the proposed algorithm are also plotted in the same figure. The extracted lower and upper boundaries are shown as starred lines in Figures 3.1b and 3.1c, respectively. For both template shapes, we see that the exact template boundaries are indeed extracted by the proposed algorithm.

Next, we conduct a more elaborate test of the proposed algorithm. For this purpose, we construct a benchmark suite of *eleven* examples taken from the QFT literature. This suite of examples is described in the Appendix. We carry out all computations on a PC/Pentium-III 550 MHz machine with 256 MB RAM using INTLAB [11]. We generate the interval template for each transfer function example to an accuracy of 1 deg and 1 dB, using the template generation algorithm in [13]. We then compare the performance of the proposed algorithm versus those of three boundary extraction algorithms available in the QFT literature: the algorithms of Lasky and Ravani [7], Boje [2], and Agamennoni *et al.* [1].

We proceed to apply the various boundary extraction algorithms to extract the boundary in each example. Table 3.1 presents the obtained performance of the various algorithms. For each example, the Table lists the number of uncertain parameters  $n$  in the transfer function, the execution time (in seconds), and the number of floating point operations (*flops*) required for the extraction. The numbers given in parenthesis are the ratios w.r.t. the proposed algorithm. A entry with star indicates that the respective algorithm failed to generate the boundary for that example, due to excessive memory or time requirements.

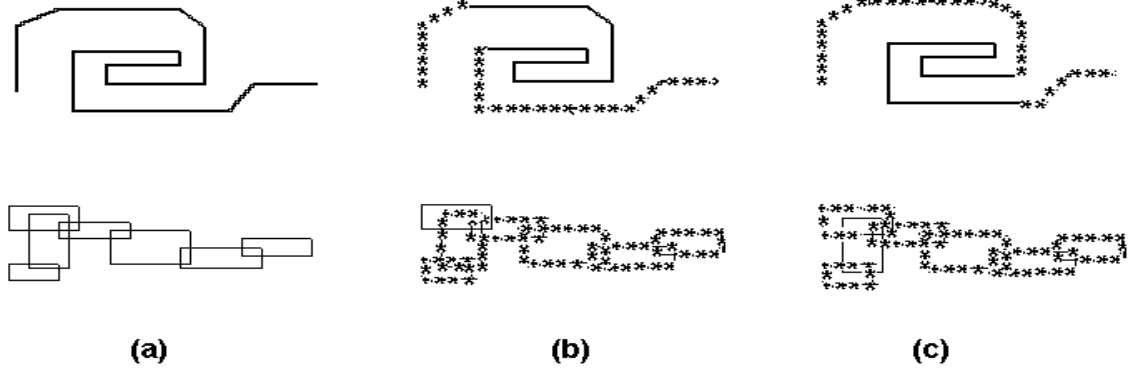


FIGURE 3.1. Application of the proposed algorithm to two arbitrary shaped interval templates. (a) original templates (b) extracted lower boundary shown in starred lines (c) extracted upper boundary shown in starred lines.

TABLE 3.1. Performances of various boundary extraction algorithms on a suite of eleven transfer function examples.

Example	$n$	Solution	Boundary Extraction using Algorithm of			
			Proposed	Lasky - Ravani [7]	Boje [2]	Agamennoni <i>et al.</i> [1]
1. Under-damped	2	time(s)	16	96.67 (6)	4890 (306)	1030 (64)
		flops	1607	474,960 (265)	199,722 (124)	73,515,618 (45,747)
2. DC Motor	2	time(s)	0.4	4.17 (10)	21.14 (53)	90.96 (227)
		flops	408	117,612 (288)	19,891 (49)	7,114,338 (17,437)
3. Simple Poles	3	time(s)	0.08	0.38 (4)	0.38 (4)	16.92 (212)
		flops	214	9340 (44)	3789 (18)	1,322,658 (6181)
4. NMP	3	time(s)	0.08	0.33 (4)	0.49 (6)	18.62 (233)
		flops	227	14,873 (66)	4161 (18)	1,452,258 (6,398)
5. Nonrational	3	time(s)	2.9	7.63 (3)	285.83 (99)	252.88 (87)
		flops	899	51,869 (58)	54,041 (60)	19,466,658 (21,654)
6. Electro-mechanical	3	time(s)	84	1360 (16)	*	4928 (59)
		flops	2886	8320 (3)	*	243,262,818 (84,291)
7. Vehicle clutch	3	time(s)	0.6	1.04 (2)	24.82 (41)	97.38 (162)
		flops	495	6013 (12)	21,260 (43)	7,615,458 (15,385)
8. Multiple lags	4	time(s)	70	1940 (28)	*	*
		flops	2665	12,773 (5)	*	*
9. Mechanical system	5	time(s)	0.8	0.82 (1)	7.08 (9)	64.42 (81)
		flops	939	21,233 (23)	14,122 (15)	5,043,618 (5371)
10. Aircraft	5	time(s)	0.1	138.96 (1,389)	6240 (62,400)	2450 (24,500)
		flops	237	112,579 (475)	225,830 (953)	81,245,538 (342,808)
11. Inverted pendulum	7	time(s)	0.04	1.37 (34)	0.05 (1)	2.36 (59)
		flops	170	128,469 (756)	672 (4)	179,298 (1055)

TABLE 3.2. Average ratios of time taken and computational effort (flops) of the three existing algorithms w.r.t. the proposed algorithm. The average is taken over all the eleven examples considered.

Average ratio	Boundary Extraction Algorithm of		
	Lasky - Ravani [7]	Boje [2]	Agamennoni <i>et al.</i> [1]
Time	136	127	2568
Flops	184	142	54,633

Table 3.1 shows the proposed algorithm to be clearly the fastest in every example. From Table 3.2, on an average, the proposed algorithm is 2–3 orders of magnitude faster than existing boundary extraction algorithms. Moreover, Table 3.1 shows the proposed algorithm as needing the least computational effort (in terms of *flops*) in every example. From Table 3.2, on an average the computational effort with the proposed algorithm is 2–4 orders of magnitude less than with the existing boundary extraction algorithms. Lastly, we see from the plots that the proposed algorithm extracts all the boundary rectangles without any approximations (due to space limitations, we regret that we are unable to provide here any plots of the templates and their boundaries; however, these are given in [12] which can be requested from the first author).

## REFERENCES

- [1] O. Agamennoni, J. L. Figueroa, and A. Palazoglu. Robust controller design under highly structured uncertainty. *International Journal of Control*, 70(5):721–733, 1998.
- [2] E. Boje. Finding nonconvex hulls of QFT templates. *Trans. of the ASME Journal of Dynamic Systems, Measurement and Control*, 122:230–231, 2000.
- [3] Y. Chait, C. Borghesani, and Y. Zheng. Single loop QFT design for robust performance in the presence of non-parametric uncertainties. *Trans. of the ASME Journal of Dynamic Systems, Measurement and Control*, 117:420–424, 1995.
- [4] P. O. Gutman, C. Baril, and L. Neumann. An algorithm for computing value sets of uncertain transfer functions in factored real form. volume 39, pages 1268–1273, 1994.
- [5] I. M. Horowitz. Survey of quantitative feedback theory (QFT). *International Journal of Control*, 53(2):255–291, 1991.
- [6] I. M. Horowitz. *Quantitative feedback design theory (QFT)*. QFT Publications, Boulder, Colorado, 1993.
- [7] T. A. Lasky and B. Ravani. Use of convex hulls for plant template approximation in QFT design. *Trans. of the ASME, Journal of Dynamic Systems, Measurement and Control*, 119(3):598–600, 1997.
- [8] R. E. Moore. *Methods and applications of interval analysis*. SIAM, Philadelphia, 1979.
- [9] P. S. V. Nataraj and G. Sardar. Template generation for continuous transfer functions using interval analysis. *Automatica*, 36:111–119, 2000.
- [10] P. S. V. Nataraj, S. Sheela, and A. K. Prakash. Interval QFT: A mathematical and computational enhancement of QFT. In M. Garcia-Sanz, editor, *Proc. 5th Int. Symposium on QFT and Robust Frequency Domain Methods*, pages 177–192, Pamplona, Spain, August 2001.
- [11] S. M. Rump. INTLAB - interval laboratory. In T. Csendes, editor, *Developments in reliable computing*. Kluwer Academic Publishers, 1999.
- [12] R. Satyanarayana. An algorithm for extraction of boundary rectangles from interval templates. Second stage M.Tech progress report, Systems and Control Engg. Group, EE Dept., IIT Bombay, September 2001.
- [13] S. Sheela and P. S. V. Nataraj. A reliable parallel-adaptive algorithm with back-tracking for QFT template generation. *IETE Technical Review*, 18, 2001.

## APPENDIX A. BENCHMARK SUITE OF EXAMPLES

The following is a benchmark suite of transfer function examples referred to in section 3. The test problems are taken from the QFT literature. The problem names reflect the general type of the system.

**Example 1 Underdamped second order system:** The transfer function for a system occurring in active noise and vibration control with highly underdamped resonances is

$$g(s) = \frac{\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2}, \quad \omega_n \in [0.75, 1.25], \quad \xi \in [0.02, 0.06], \quad \omega = 1.$$

**Example 2 DC Motor:** The DC Motor drives a viscously damped inertial load. The transfer function between the torque and armature voltage is

$$\begin{aligned} g(s) &= \frac{K(J_l s + B_l)}{(Ls + R)(J_m s + J_l s + B_m + B_l) + K^2}, \quad K \in [0.2, 0.6], \quad J_l \in [1e-5, 3e-5], \quad J_m = 2e-3 \\ B_m &= 2e-5, \quad L = 1e-2H, \quad R = 1\Omega, \quad B_l = B_m, \quad \omega = 20. \end{aligned}$$

**Example 3** *Simple poles*: The transfer function for a stable second order system with real poles is

$$g(s) = \frac{k}{(s+a)(s+b)}, \quad a \in [1, 5], \quad b \in [20, 30], \quad k \in [1, 10], \quad \omega = 1.$$

**Example 4** *Non-minimum phase*: The transfer function for a non-minimum phase system with real poles and zeros is

$$g(s) = K \frac{(1 - Ds)}{s(1 + Bs)}, \quad B \in [0.3, 1], \quad D \in [0.05, 0.1], \quad K \in [1, 3], \quad \omega = 1.$$

**Example 5** *Non-rational*: The transfer function for a non-rational system is

$$g(s, T, a, b) = \frac{e^{-sT}}{1 + be^{-as}}, \quad a \in [1, 2], \quad b \in [0.4, 0.6], \quad T \in [0.01, 0.02], \quad \omega = 2.$$

**Example 6** *Electro-mechanical*: The transfer function between control torque to motor speed of an electro-mechanical system is

$$\begin{aligned} g(s) &= \frac{J_l s^2 + ds + k}{J_l J_m s^3 + (J_l + J_m) ds^2 + (J_l + J_m) ks}, \quad J_m = 0.4, \quad J_l \in [5.6, 8], \quad d \in [30, 300], \quad k \in [5880, 5900], \\ \omega &= 10\pi. \end{aligned}$$

**Example 7** *Vehicle clutch system*: The transfer function between the input clutch position to the output transmission speed is

$$\begin{aligned} g(s) &= \frac{k_c(s^2 + \frac{c_s}{j_v}s + \frac{k_s}{j_v})}{J_e s \left( s^2 + \left( \frac{c_s}{j_v} + \frac{c_s}{g r^2 j_c} \right) s + \left( \frac{k_s}{j_v} + \frac{k_s}{g r^2 j_c} \right) \right)}, \quad J_v \in [1400, 11000], \quad k_s \in [5800, 115000], \\ k_c &\in [100, 800], \quad J_c = 0.09, \quad j_e = 3.07, \quad c_s = 377, \quad c_e = 18.5, \quad g_r = 27.0, \quad \omega = 10. \end{aligned}$$

**Example 8** *Multiple transport lags*: The transfer function for a system with multiple transport lags is

$$\begin{aligned} g(s) &= \frac{(T + \tau_1)e^{-sT_1}}{\log_{10} T_1 - \frac{e^{-sT}}{1 + \cos(\tau_1)s}} + \frac{1 - e^{-sT_r}}{sT_r \left( 1 + \left( \frac{sT_r}{4\pi} \right)^2 \right)}, \quad T_1 \in [3, 5], \quad T_r \in [0.5, 0.7], \quad T \in [9.25, 9.35], \\ \tau_1 &\in [0.49, 0.5], \quad \omega = 0.5. \end{aligned}$$

**Example 9** *Mechanical system*: The transfer function for a mechanical system is

$$g(s) = \frac{km}{s(s^2 fm^2 + bms + c)}, \quad f \in [1, 2], \quad m^2 \in [1, 10], \quad b \in [0.5, 1], \quad c \in [2, 3], \quad k \in [0.5, 2], \quad \omega = 8.$$

**Example 10** *Aircraft, longitudinal Motion*: The transfer function for the longitudinal motion of an aircraft is (from aerodynamic data, the ranges for the uncertain parameters are identified)

$$g(s) = \frac{k(1 + \frac{s}{z})}{s(1 + \frac{s}{p})(1 + \frac{2\xi}{\omega_n}s + \frac{s^2}{\omega_n^2})}, \quad z \in [0.5, 0.75], \quad p \in [1, 10], \quad \xi \in [0.8, 0.9], \quad \omega_n \in [5, 6], \quad k \in [0.2, 2], \quad \omega = 0.1.$$

**Example 11** *Inverted pendulum*: The transfer function between pendulum angle to the cart's motor current is

$$\begin{aligned} g(s) &= \frac{K\alpha\omega_n^2(1/L)s^2e^{-s\tau}}{(s(s + \alpha) - K_{gf}K\alpha)(s^2 + 2\xi\omega_ns + \omega_n^2)(s^2 - g/L)}, \quad L \in [0.3, 0.45], \quad K \in [1.5, 1.7], \quad \alpha \in [15, 17], \\ \omega_n &\in [50, 70], \quad \xi \in [0.001, 0.02], \quad \tau \in [0.014, 0.015], \quad K_{gf} \in [0.005, 0.15], \quad g = 9.81, \quad \omega = 1. \end{aligned}$$