Pergamon

International Journal of Machine Tools & Manufacture 42 (2002) 1479–1486

INTERNATIONAL JOURNAL OF
MACHINE TOOLS
& MANUFACTURE
DESIGN, RESEARCH AND APPLICATION

Automatic trimming of machining stability lobes

Tony L. Schmitz

Manufacturing Metrology Division, National Institute of Standards and Technology, MS 8220, Gaithersburg, MD 20899, USA

Received 15 May 2001; received in revised form 8 February 2002; accepted 15 March 2002

Abstract

This paper details an automatic method for the organization of the stability information included in the multiple parameterized lobes that make up analytic stability lobe diagrams into a single pair of vectors that describe the continuous relationship between spindle speed and allowable chip width. Additionally, the method describes the trimming of adjacent lobes and the removal of the spurious loops that are contained in each lobe for multiple-mode tool point frequency response functions. Example MATLAB code is provided which implements the steps to perform the necessary trimming operations in AutoCAD using an executable script file. The trimmed lobe coordinate information is then exported from AutoCAD using the drawing interchange file (.dxf) format and read back into MATLAB for plotting and analysis. Examples for the development of trimmed stability lobe diagrams from tool point dynamic data that exhibits multiple modes, as well as directly from experimental frequency response measurements, are provided. Published by Elsevier Science Ltd.

Keywords: Stability lobes; Chatter; Lobe trimming

1. Introduction

Numerous researchers have investigated methods for chatter avoidance in machining operations. One powerful graphical tool that has been developed is the so-called 'stability lobe diagram' that plots the boundary between stable and unstable cuts as a function of spindle speed and chip width, as shown in Fig. 1. These diagrams provide a means of selecting favorable combinations of spindle speed and axial depth of cut in end milling, for example, for increased material removal rates (MRR). Early work in the analytic prediction of stability lobes, based on simplified models, was performed by Tobias and Fishwick [1,2], Tlusty and colleagues [3,4], and Merrit [5]. Subsequent improvements to this early research include efforts by Minus et al. [6], concerned with an iterative analytic solution of the mathematical milling model given by Sridhar et al [7], and Altintas and Budak [8], which implemented a Fourier series expansion of the time-varying milling force coefficients. Analytic stability formulations have also been developed by Stépán and colleagues [19–21]. Recent research by Budak and Altintas [9] describes a general formulation for analytic stability using multiple degree-of-freedom models for the cutting tool and workpiece, including effects in the axial direction. Additionally, analytic stability formulations for variable pitch cutters [10], ball end mills [11], and face mills [12] have been detailed in the literature.

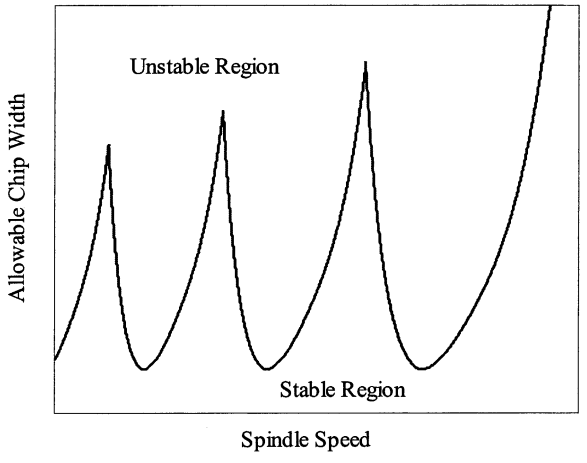

Fig. 1. Sample stability lobe diagram.

In general, these diagrams consist of a number of individual lobes that each span a limited range of spindle speeds and vertically separate stable (axial depths below the lobe) and unstable (axial depths above the lobe) cuts. Multiple lobes are produced by incrementing the integer number of vibration waves, k, imprinted on the cut sur-

0890-6955/02/$ - see front matter Published by Elsevier Science Ltd.
PII: S0890-6955(02)00034-2

face and solving the analytic formulation for each k. The $k = 0$ lobe appears at the right of the diagram (i.e. highest spindle speeds) and all subsequent lobes occur sequentially to the left (see Fig. 2). It is then the convolution of these multiple lobes that provides the continuous boundary between stable/unstable depths of cut.

The algorithms described in the literature for the calculation of analytic stability lobes do not address techniques to trim the intersections of adjacent lobes, nor do they contain methods to convert the multiple parameterized pairs of spindle speed and allowable depth of cut vectors (corresponding to each lobe) into a single pair of vectors describing the continuous relationship between spindle speed and stable depth over the selected spindle speed range. This continuous map of spindle speed versus allowable axial depth may not be critical in general situations, where the stability lobe diagram is simply examined visually to determine best spindle speeds and corresponding depths. However, if this information is to be used in optimization schemes for the selection of maximized MRR over some set of process variables, the ability to automatically select the maximum allowable axial depth and corresponding spindle speed becomes critical. For example, recent research at the National Institute of Standards and Technology has focused on methods to analytically predict the tool point frequency response function (FRF) based on a minimum set of measurements and an analytic model of the cutting tool. Toward this end, the receptance coupling substructure analysis (RCSA) method has been developed to predict the tool point FRF and allows the selection of tool overhang for maximum MRR [13–15,22], in a technique referred to as 'tool tuning' [16–18]. The ability to scan the complete set of stability lobe diagrams, which correspond individually to the simulated FRF for each overhang, and select the optimum combination of spindle speed and axial depth is especially important in this situation. Therefore, a method is described here to convert a given series of analytically derived stability lobes into a continuous representation of the boundary between stable and unstable cuts using the mathematical software MATLAB and graphical computer-aided design package AutoCAD LT[1].

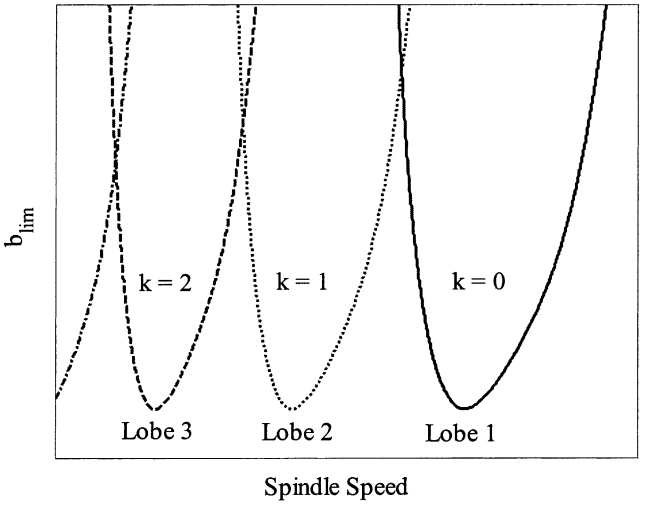

Fig. 2. Multiple stability lobes ($k = 0$ to 2 identified).

2. The algorithm

The procedure for consolidation of the stability information carried by multiple lobes into a single pair of vectors (spindle speed, n, and allowable axial depth, b_{lim}), as well as the necessary trimming of these lobes, is composed of three fundamental steps:

1. Calculate the individual stability lobes using an appropriate analytic method (the algorithm described in [8] was selected for this research) and use the coordinates described by the multiple $\{n, b_{lim}\}$ parameterized pairs to write an ASCII file containing the necessary AutoCAD commands, referred to as a 'script' file, to plot each successive lobe. This script file will also contain the required commands (that would otherwise be typed at the command prompt in AutoCAD) for trimming adjacent lobes and removing 'loops' in the individual lobes due to closely spaced modes. The latter will be detailed in subsequent paragraphs.
2. Once the lobes have been plotted and trimmed in AutoCAD, export the coordinate information using the Drawing Interchange File format (file extension .dxf).

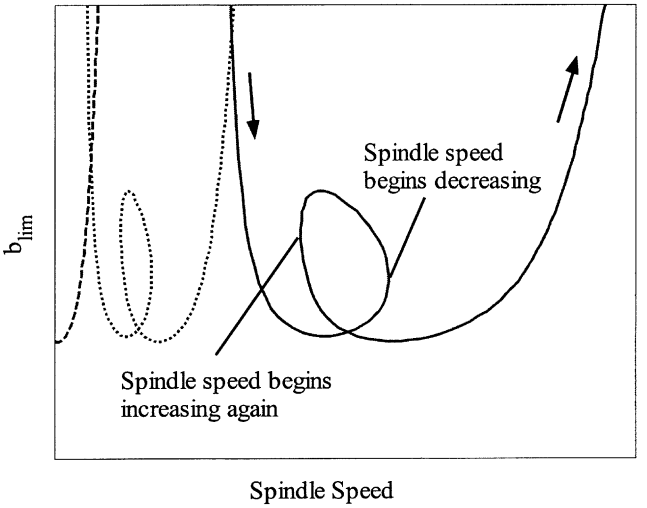

Fig. 3. Identifying loops in lobes.

[1] Commercial equipment is identified in order to adequately specify certain procedures. In no case does such identification imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the equipment identified is necessarily the best available for the purpose.

3. Read and record the ASCII .dxf file coordinates into the desired pair of n and b_{lim} vectors and analyze the results.

A description of each necessary step, as well as example MATLAB and AutoCAD code, will now be provided for a number of common situations including systems best modeled by two closely spaced modes, and the direct use of experimental data, rather than modal fits, to develop stability lobe diagrams.

2.1. Step 1

The procedure begins with the calculation of the stability lobes for a given dynamic system (tool/holder/spindle/machine/workpiece) and machining parameters (including axial and radial depth, cutter geometry, and specific cutting energy coefficients). This analysis gives a user-selected number, K, of $\{n, b_{lim}\}$ pairs spanning the desired spindle speed range. The respective $\{n, b_{lim}\}$ coordinates are then used to produce K continuous line segments in AutoCAD using the 'pline', or polyline, command.

Prior to plotting the individual polylines, however, three actions are taken to modify the data. First, loops in the individual $\{n, b_{lim}\}$ pairs are removed. These loops occur due to closely spaced modes of similar magnitude in the tool point FRF and must be identified and trimmed. An example of these loops is shown in Fig. 3. The method employed here to remove the superfluous section of the lobes is to identify portions of the lobes where the spindle speeds twice changes direction, i.e., locate the section of the lobe between the first decrease in spindle speed and subsequent increase in spindle speed as the lobe is scanned from left to right in Fig. 3. This portion of each lobe is then deleted and the original lobe divided into two overlapping sections. MATLAB code used to perform this operation is given here, where *n__1* is the spindle speed vector for the $k = 0$ lobe and *blim__1* is the corresponding allowable axial depth vector:

```
% Break loops in lobes
counter = 1;
flag = 0;
while (counter < length(n_1))
   while ((counter < length(n_1)) & (n_1(counter + 1) > n_1(counter)))
      counter = counter + 1;
   end
   start_cut = counter;

   counter = start_cut + 1;
   while ((counter < length(n_1)) & (n_1(counter + 1) < n_1(counter)))
      counter = counter + 1;
   end
   end_cut = counter;

   if (end_cut > start_cut + 3)     % Found loop, stop searching
      flag = 1;
      break;      % Otherwise, continue until the end of vector is reached
   end
end

% Divide original lobes into two separate sections
if ((start_cut < length(n_1)) & (flag == 1))
   % Lobe 1 (k = 0)
   n_1a = n_1(1:start_cut);
   blim_1a = blim_1(1:start_cut);
   n_1b = n_1(end_cut:length(n_1));
   blim_1b = blim_1(end_cut:length(n_1));

   % Lobe 2 (k = 1)
   n_2a = n_2(1:start_cut);
   blim_2a = blim_2(1:start_cut);
   n_2b = n_2(end_cut:length(n_1));
   blim_2b = blim_2(end_cut:length(n_1));

   % Continue for all remaining lobes, k = 2 to K
end
```

The reader should note two important items from the preceding code. First, it was required that a minimum of three points be identified on the loop prior to its recognition as a loop. This is not a necessary condition for smooth modal fits, but must be incorporated if inherently noisy experimental data is to be analyzed. The selection of three points is based on experience, but may be adjusted depending on the quality of the data. Second, once the indices for the beginning and end points of the portion to be deleted (*start__cut* and *end__cut*, respectively) are identified, they may be used to section all lobes due to the nature of the algorithm used for the analytic stability lobe calculation [8].

The second adjustment applied to the data is the truncation of the end of the first, or right most, lobe (lowest k value) and beginning of the last lobe ($k = K$). This is not a critical step, but was added because the first and last values in the allowable depth of cut vectors are, in general, much higher than the intersections of adjacent lobes. For simplicity, the first and last lobes were truncated at the minimum value, or critical stability limit. The corresponding code is given below (in this example lobes 2–6, $k = 1$ to 5, were desired):

```
% Trim end of first lobe (right) and beginning of last lobe (left)
if flag == 0   % No loops
   [bmin, min_index] = min(blim_2);
   n_2 = n_2(1:min_index);
   blim_2 = blim_2(1:min_index);

   [bmin, min_index] = min(blim_6);
   n_6 = n_6(min_index:length(n_6));
   blim_6 = blim_6(min_index:length(blim_6));
else     % Loops exist
   [bmin, min_index] = min(blim_2b);
   n_2b = n_2b(1:min_index);
   blim_2b = blim_2b(1:min_index);

   [bmin, min_index] = min(blim_6a);
   n_6a = n_6a(min_index:length(n_6a));
   blim_6a = blim_6a(min_index:length(blim_6a));
end
```

The third adjustment to the multiple $\{n, b_{lim}\}$ pairs is the application of a scaling factor to normalize the span to height ratio of the entire stability lobe diagram to unity. This action is taken because the default aspect ratio for AutoCAD drawings is one (i.e., $\Delta x = \Delta y$ for plotting images to the screen) and, in the author's experience, very high or low aspect ratio drawings posed difficulties for the AutoCAD 'trim' command used to automatically select and trim adjacent lobes and loops. The scaling factor, *scale*, was calculated using the following code (again arbitrarily using lobes 2–6):

```
% Determine scaling factor for AutoCAD with dx = dy plotting
if (flag == 0)
   scale = round((max(n_2) - min(n_6))/(max(blim_6) - min(blim_6)))
else
   scale = round((max(n_2b) - min(n_6a))/(max(blim_6) - min(blim_6)))
end
```

Once the scaling factor was determined, the *blim* data was first multiplied by this value and then used in the development of the AutoCAD script file.

After the previous three modifications had been applied, the new $\{n, b_{lim}\}$ vector pairs were used to generate multiple polylines in AutoCAD. These polylines

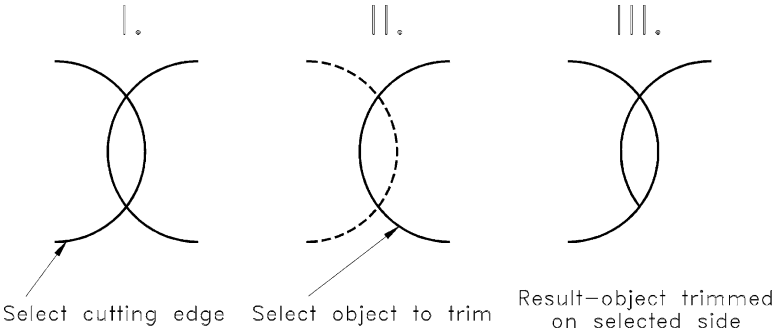

Fig. 4. AutoCAD 'trim' command example.

were then trimmed using the AutoCAD 'trim' command at intersections between lobes and truncated loop sections, if any. The AutoCAD trim command prompts the user to first select a 'cutting edge' or boundary. An object to be trimmed is then requested. Provided the selected object physically crosses the cutting edge, any portion of the object that lies on the selected side of the cutting edge is deleted (see Fig. 4). This operation may be performed automatically from the script file by entering coordinates of points on the cutting edge and object to be trimmed (known from the $\{n, b_{\lim}\}$ pairs). It is during this step that a unity aspect ratio for the stability lobe diagram is helpful.

The trimming algorithm begins with lobe K and continues to the right as shown in Fig. 5. First, the final point on the scaled K lobe is used to select the cutting edge (shown as the dashed line in step II of Fig. 5). Second, the first point on the $(K-1)$ lobe, located immediately to the right, is used to identify the object to the trimmed. This trims the portion of the $(K-1)$ lobe that lies to the left of the K lobe cutting edge. Third, the end of the $(K-1)$ lobe is used to identify the cutting edge in another trim command; note that the beginning of the $(K-1)$ lobe no longer exists. Fourth, the final point on the K lobe is selected and the portion of this lobe to the right of the trimmed $(K-1)$ lobe is deleted. The same algorithm is used whether trimming adjacent lobes or loops within individual lobes. Example code for constructing and trimming the necessary polylines for loop-containing lobes (i.e., flag = 1) in a script file, ident-

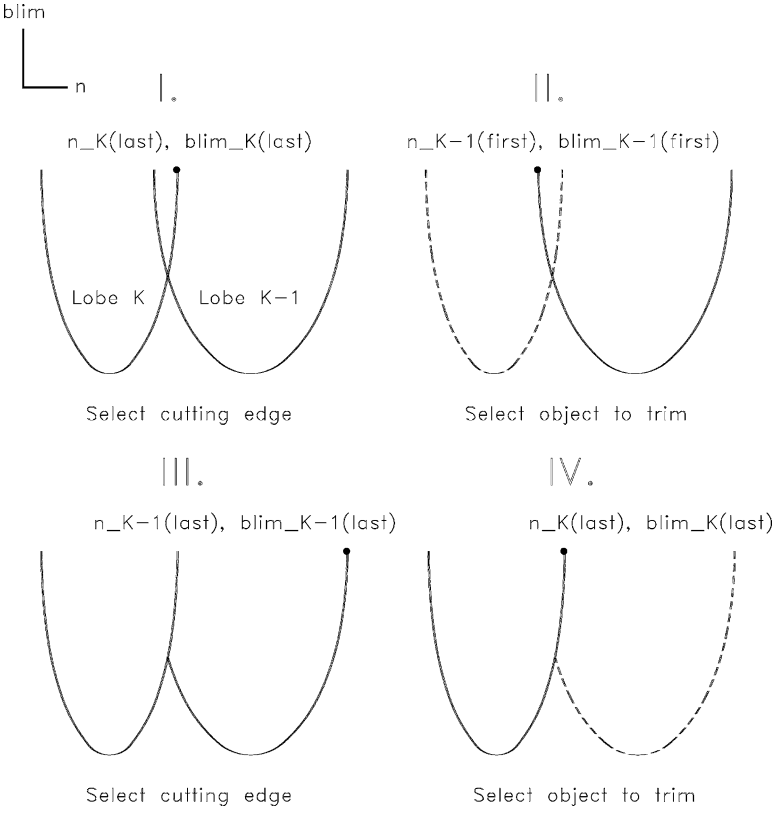

Fig. 5. Trimming algorithm.

ified as 'lobes.scr', is shown below (again lobes two through six are arbitrarily considered):

```
% Write stability lobes to script file
outfile = fopen('lobes.scr', 'w');

% k = 1
% Construct polyline for left section of lobe (after sectioning loop)
fprintf(outfile,'pline %.2f,%.2f ', n_2a(1), blim_2a(1)*scale);
for (counter = 2:length(n_2a))
   fprintf(outfile, '%.2f,%.2f ', n_2a(counter), blim_2a(counter)*scale);
end
fprintf(outfile,' ');

% Construct polyline for right section of lobe
fprintf(outfile,'pline %.2f,%.2f ', n_2b(1), blim_2b(1)*scale);
for (counter = 2:length(n_2b))
   fprintf(outfile, '%.2f,%.2f ', n_2b(counter), blim_2b(counter)*scale);
end
fprintf(outfile,' ');

% Repeat previous constructions for all lobes k = 2 to 5

% Trim loop in lobe 6 (sections 6a and 6b)
fprintf(outfile,'trim %.2f,%.2f  %.2f,%.2f  ', n_6a(length(n_6a)),
   blim_6a(length(blim_6a))*scale, n_6b(1), blim_6b(1)*scale);
fprintf(outfile,'trim %.2f,%.2f  %.2f,%.2f  ', n_6b(length(n_6b)),
   blim_6b(length(blim_6b))*scale, n_6a(length(n_6a)),
      blim_6a(length(blim_6a))*scale);

% Trim overlap of adjacent lobes 6 and 5 (sections 6b and 5a, respectively)
fprintf(outfile,'trim %.2f,%.2f  %.2f,%.2f  ', n_6b(length(n_6b)),
   blim_6b(length(blim_6b))*scale, n_5a(1), blim_5a(1)*scale);
fprintf(outfile,'trim %.2f,%.2f  %.2f,%.2f  ', n_5a(length(n_5a)),
   blim_5a(length(blim_5a))*scale, n_6b(length(n_6b)),
      blim_6b(length(blim_6b))*scale);

% Continue trimming lobes 5, 4, and 3 in same manner

% Finish by trimming loop in first lobe, 2 (sections 2a and 2b)
fprintf(outfile,'trim %.2f,%.2f  %.2f,%.2f  ', n_2a(length(n_2a)),
   blim_2a(length(blim_2a))*scale, n_2b(1), blim_2b(1)*scale);
fprintf(outfile,'trim %.2f,%.2f  %.2f,%.2f  ', n_2b(length(n_2b)),
   blim_2b(length(blim_2b))*scale, n_2a(length(n_2a)),
      blim_2a(length(blim_2a))*scale);

% Replot drawing in AutoCAD
fprintf(outfile,'regen ');

% Close script file
fclose(outfile);
```

2.2. Step 2

This step begins by executing the 'script' command in AutoCAD. This command causes the statements listed in the ASCII .scr file developed in Step 1 to be completed. At the end of the script file execution, the trimmed lobes may be viewed on the screen[2]. The coordinate information for the polylines is then recorded in a .dxf file using the 'dxfout' command. When using this function, the user is first prompted for a filename, then the number of decimal places for the exported polyline coordinates. The selected number of decimal places should match the precision of the floating point values written to the script file in Step 1 (e.g. two decimal places were used in the previous MATLAB code).

2.3. Step 3

The final step in the procedure is reconstructing the trimmed stability lobes from the polyline coordinate information contained in the .dxf file. This is accomplished by scanning the file using the 'fscanf' function and recording the spindle speed and corresponding allowable axial depth for each point on the trimmed polylines. In order to do this, however, the format of the .dxf file must be known beforehand. The format for the polyline descriptions contained in the .dxf files exported by AutoCAD LT, Release 2—c1 (1996) is given below (note that several hundred lines of header information precede this text in the actual files):

```
POLYLINE
(17 lines of filler information)
VERTEX
(5 lines of filler information)
n[1]  - defines first spindle speed coordinate
(1 line of filler information)
b[1]  - defines corresponding first allowable axial depth coordinate
(3 lines of filler information)
VERTEX
     …repeat previous section for each point on polyline
VERTEX
(5 lines of filler information)
n[last]    - last spindle speed coordinate in this polyline
(1 line of filler information)
b[last]    - last corresponding allowable axial depth coordinate
(3 lines of filler information)
SEQEND
(5 lines of filler information)
```

This block of information is repeated for each polyline that exists in the AutoCAD drawing. The end of the final polyline section is followed by the string 'ENDSEC'. Sample Matlab code for opening and reading AutoCAD LT-generated .dxf files follows

```
% Read stability lobes coordinates from .dxf file
infile = fopen('lobes.dxf', 'r');
scale = xxxxxxx;        % AutoCAD scaling factor
counter = 1;

% Read data from trimmed lobes
% Note: Must skip header information down to first appearance of 'POLYLINE'
while (length(fscanf(infile, '%s', 1)) == 8) % 8 characters in 'POLYLINE'
   filler = fscanf(infile, '%s', 17);
   while (fscanf(infile, '%s', 1) == 'VERTEX')
      filler = fscanf(infile, '%s', 5);
      spindle_speed = fscanf(infile,'%f', 1);
      n(counter) = spindle_speed;
      filler = fscanf(infile, '%s', 1);
      b = fscanf(infile,'%f', 1);
      blim(counter) = b;
      filler = fscanf(infile, '%s', 3);
      counter = counter + 1;
   end
   filler = fscanf(infile, '%s', 5);
end
fclose(infile);

% Sort vectors in ascending order of spindle speeds
[n, index] = sort(n);
blim = blim(index);

% Correct units
blim = blim/scale;
```

One necessary step in the previous code that may not be immediately obvious is the use of the MATLAB 'sort' command to arrange the coordinates in order of ascending spindle speed for subsequent plotting of the data. This is necessary because the .dxf file does not necessarily list the coordinate information for the individual polylines as they appear on the screen from left to right. It has been observed that other orders may occur.

3. Examples

Two examples will now be provided to illustrate the implementation of the three steps described in the pre-

[2] One important consideration for the script file trim commands to execute properly is that all points on the individual polylines must lie within the current viewing window in AutoCAD. Otherwise, the trim command cannot locate the appropriate points on the lobes and fails.

Table 1
Example 1 modal parameters

	Natural frequency (Hz)	Stiffness (N/m)	Damping ratio
Mode 1	1000	7×10^6	0.01
Mode 2	1050	7×10^6	0.01

vious paragraphs. First, a modal fit to a system best modeled by two closely spaced modes of equal amplitude is used to develop stability lobes that are then trimmed and arranged into the desired vector pair. Second, experimental tool point FRF measurements are used to generate stability lobes, which are also modified according to the algorithm described in the previous section.

3.1. Two mode modal fit

The selected modal parameters for both the *X* and *Y*-direction tool point FRFs are given in Table 1. It is seen that the two modes have identical modal stiffness and damping values, but are separated by 50 Hz. The real and imaginary parts of the corresponding FRF are shown in Fig. 6. The parameterized analytic stability lobes for slot milling in aluminum with a two-flute cutter (that exhibits the dynamic response shown in Fig. 6) are given in Fig. 7. Lobe 2, with the characteristic loop, is plotted with a dashed line. Also shown are lobes 3–6, which were adequate to cover the desired spindle speed range. The procedures described in Step 1 of the previous section are then implemented to give the modified lobes shown in Fig. 8, whose coordinate information is defined in the corresponding script file.

Step 2 is then completed and the coordinates for the trimmed lobes written to the appropriate .dxf file. The .dxf file is then scanned according to the instructions

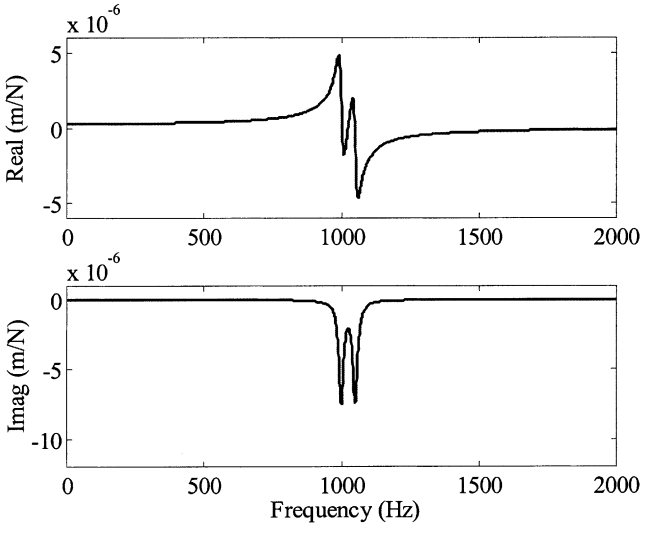

Fig. 6. FRF for Example 1.

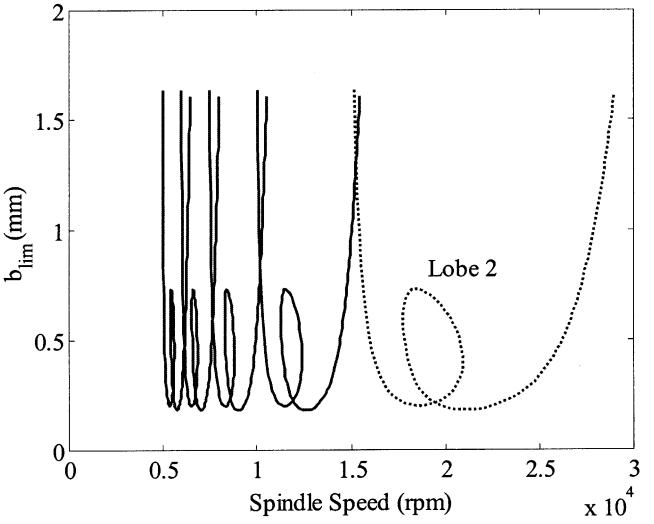

Fig. 7. Example 1 unmodified stability lobes.

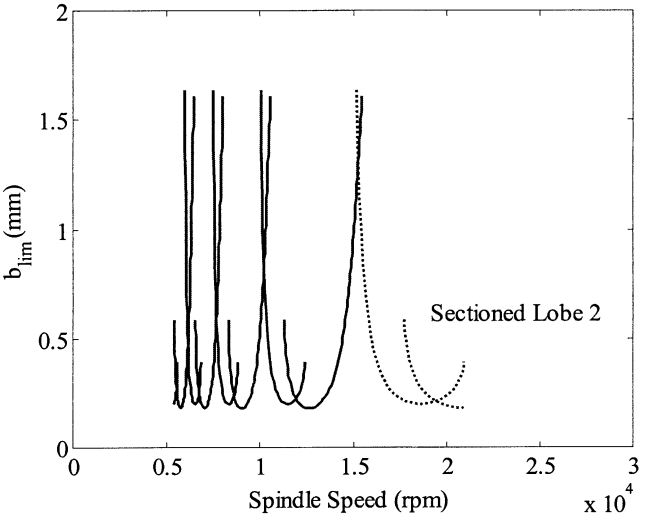

Fig. 8. Example 1 stability lobes after Step 1.

given in Step 3 and the single pair of $\{n, b_{\lim}\}$ vectors recorded. The results are shown in Fig. 9.

3.2. Example 2: experimental FRF data

In some instances it may be desirable to generate the stability lobes directly from experimental data and avoid the necessity of a modal fit to the measured FRF. The lobe trimming procedure detailed in the previous section does not change in this case. As an example, consider the measured Y-direction frequency response for a two-flute, 11.8 mm diameter by 118.5 mm overhang carbide endmill shown in Fig. 10 (the *X*-direction response was similar). The stability lobes ($k = 0$ to 5) for a 50% radial immersion, down milling cut in aluminum, calculated from the *X* and *Y*-direction FRFs, are shown in Fig. 11. It can be seen in the $k = 0$ lobe (plotted with a dashed

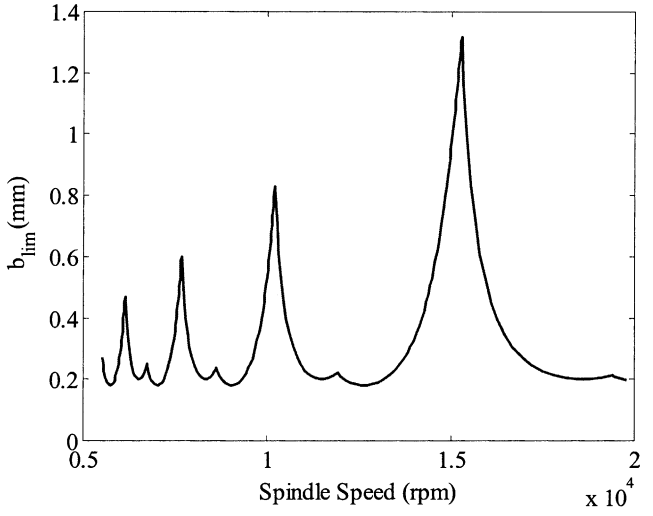

Fig. 9. Example 1 final results.

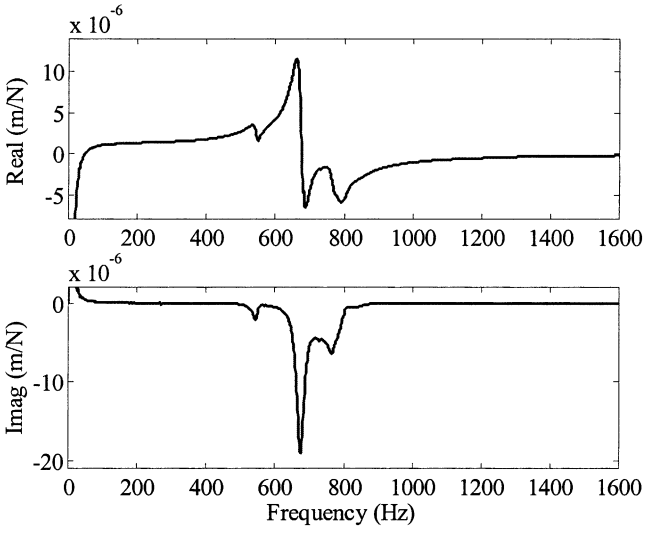

Fig. 10. Measured *Y*-direction frequency response for Example 2.

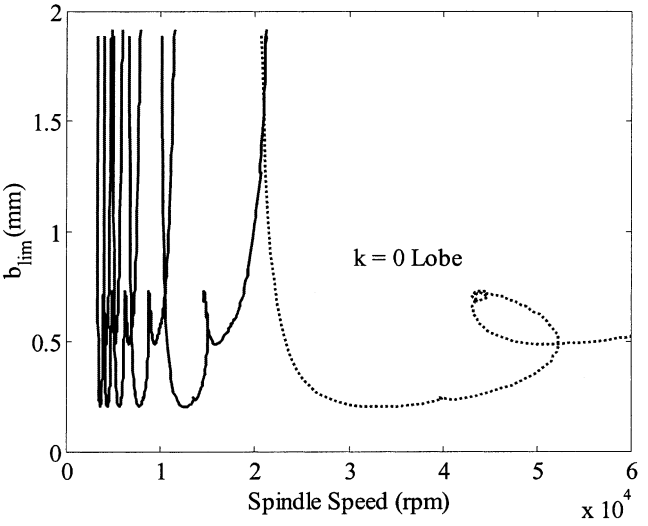

Fig. 11. Unmodified stability lobes for Example 2.

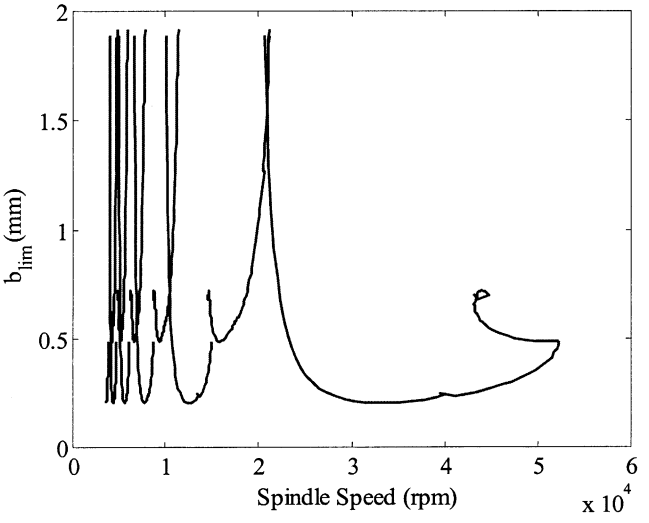

Fig. 12. Sectioned lobes for Example 2.

line) that two loops exist due to the three modes located within the measurement bandwidth. After deleting the redundant looped portion of the lobes according to Step 2, the result shown in Fig. 12 is obtained. The effect of noise in the FRF data is shown by the discontinuity in each of the lobes $k = 1$ to 5 after sectioning the loops. The script file is then executed in AutoCAD and the trimmed lobe coordinates exported to a .dxf file. The discontinuities cause no problem with the trimming algorithm described previously. Since the left and right (i.e. *a* and *b*) portions of the $k = 1$ to 5 lobes do not physically intersect, no trimming occurs. However, the script file is not interrupted and no errors are reported. The final pair of $\{n, b_{\lim}\}$ obtained from the .dxf file may then be plotted as shown in Fig. 13 (the maximum speed in the diagram has been limited to the top available spindle speed of 20 000 rpm for the high-speed machining center used in this study). It is clear that the discontinuity in the $k = 1$ lobe causes some undesirable local oscillations in predicted stability. However, this does not affect the overall utility of the diagram.

4. Conclusions

This paper details an automatic method for the trimming of analytic stability lobes and the consolidation of the stability information carried in the multiple lobes that typically make up a stability diagram into a single pair of vectors, collectively describing the continuous relationship between spindle speed and the allowable chip width for a given set of machining conditions. Sample MATLAB code is provided which contains the steps to:

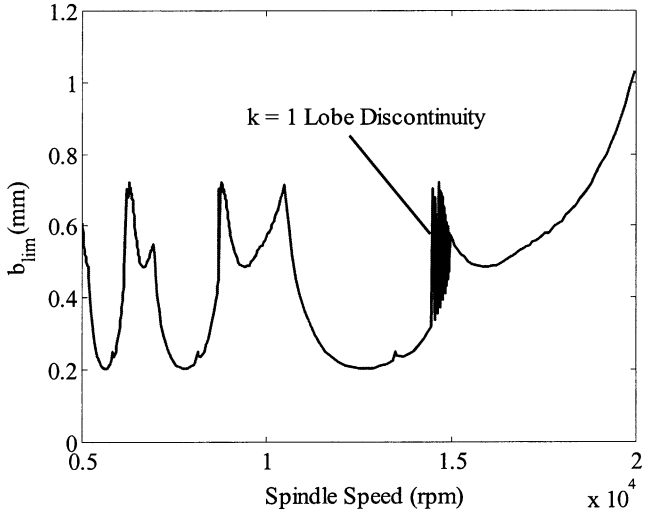

Fig. 13. Example 2 trimmed stability lobes.

1. perform the necessary trimming operations on adjacent lobes and remove spurious 'loops' from each lobe;
2. develop 'script' files which contain the required AutoCAD commands to automatically plot and trim the modified lobes; and
3. read the trimmed lobe coordinates from the drawing interchange file (.dxf) exported from AutoCAD, assemble these coordinates into the appropriate vectors, and plot the results.

Examples of the development of stability lobe diagrams from tool point dynamic data that exhibits multiple modes, as well as directly from experimental frequency response measurements, are provided.

Acknowledgements

The author would like to thank Dr Robert Ivester, National Institute of Standards and Technology, Gaithersburg, MD, and Dr Matthew Davies, University of North Carolina-Charlotte, Charlotte, NC, for helpful suggestions during this research.

References

[1] S. Tobias, Machine Tool Vibration, Blackie and Sons Ltd, London, 1965.
[2] S. Tobias, W. Fishwick, Theory of Regenerative Machine Tool Chatter, The Engineer, February, Vol. 205 (1958).
[3] F. Koenisberger, J. Tlusty, Machine Tool Structures, vol. I: Stability Against Chatter, Pergamon Press, New York, 1967.
[4] J. Tlusty, M. Polacek, The stability of the machine tool against self-excited vibration in machining, in: ASME Production Engineering Research Conference, Pittsburgh, PA (1963).
[5] H. Merritt, Theory of self-excited machine tool chatter, Transactions of ASME Journal of Engineering for Industry 87 (1965) 447–454.
[6] I. Minis, T. Yanushevsky, R. Tembo, R. Hocken, Analysis of linear and nonlinear chatter in milling, Annals of the CIRP 39 (1990) 459–462.
[7] R. Sridhar, R. Hohn, G. Long, A stability algorithm for the general milling process, Transactions of ASME Journal of Engineering for Industry 90 (1968) 330–334.
[8] Y. Altintas, E. Budak, Analytical prediction of stability lobes in milling, Annals of the CIRP 44 (1) (1995) 357–362.
[9] E. Budak, Y. Altintas, Analytical prediction of chatter stability conditions for multi-degree of freedom systems in milling. Part I: modeling, Part II: applications, Transactions of ASME Journal of Dynamic Systems, Measurement and Control 120 (1998) 22–36.
[10] Y. Altintas, S. Engin, E. Budak, Analytical stability prediction and design of variable pitch cutters, Transactions of ASME Journal of Manufacturing Science and Engineering 121 (1999) 173–178.
[11] Y. Altintas, E. Shamoto, P. Lee, E. Budak, Analytical prediction of stability lobes in ball end milling, Transactions of ASME Journal of Manufacturing Science and Engineering 121 (4) (1999) 586–592.
[12] S. Jensen, Y. Shin, Stability analysis in face milling operations. Part 1: Theory, Part 2: experimental validation, Transactions of ASME Journal of Manufacturing Science and Engineering 121 (4) (1999) 600–614.
[13] T. Schmitz, R. Donaldson, Predicting high-speed machining dynamics by substructure analysis, Annals of the CIRP 49 (1) (2000) 303–308.
[14] T. Schmitz, M. Davies, M. Kennedy, High-speed machining frequency response prediction for process optimization, in: Proceedings of the 2nd International Seminar on Improving Machine Tool Performance, La Baule, France, 3–5 July 2000.
[15] T. Schmitz, M. Davies, K. Medicus, J. Snyder, Improving high-speed machining material removal rates by rapid dynamic analysis, Annals of the CIRP 50 (1) (2001) 263–268.
[16] J. Tlusty, S. Smith, W. Winfough, Techniques for the use of long slender end mills in high-speed machining, Annals of the CIRP 45 (1) (1996) 393–396.
[17] S. Smith, W. Winfough, J. Halley, The effect of tool length on stable metal removal rate in high-speed milling, Annals of the CIRP 47 (1) (1998) 307–310.
[18] M. Davies, B. Dutterer, J. Pratt, A. Schaut, On the dynamics of high-speed milling with long, slender endmills, Annals of the CIRP 47 (1) (1998) 55–60.
[19] G. Stépán, T. Kalmár-Nagy, Nonlinear regenerative machine tool vibrations, DETC97/VIB-4021, in: Proceedings of DETC '97 1997 ASME Design Engineering Technical Conferences, Sacramento, CA, 14–17 September 1997.
[20] G. Stépán, T. Insperger, Stability of retarded systems with parametric excitation, Zeitschrift Fur Angewandte Mathematik Und Mechanik 81 (Suppl. 2) (2001) S195–S196.
[21] G. Stépán, Modelling Nonlinear regenerative effects in metal cutting, Philosophical Transactions of the Royal Society of London Series A—Mathematical, Physical, and Engineering Sciences 359 (1781) (2001) 739–757.
[22] T. Schmitz, M. Davies, Tool point frequency response prediction for high-speed machining by RCSA, Journal of Manufacturing Science and Engineering 123 (2001) 700–707.