Electronic Journal of Differential Equations
Vol. **1995**(1995), No. 05, pp. 1-13. Published May 2, 1995.
ISSN 1072-6691. URL: http://ejde.math.swt.edu or http://ejde.math.unt.edu
ftp (login: ftp) 147.26.103.110 or 129.120.3.113

Eigenvalue Computations for Regular Matrix Sturm-Liouville Problems *

H.I. Dwyer & A. Zettl

Abstract

An algorithm is presented for computing eigenvalues of regular self-adjoint Sturm-Liouville (SL) problems with matrix coefficients and separated boundary conditions.

Introduction

In his book *Discrete and Continuous Boundary Problems* [1], F.V. Atkinson characterizes eigenvalues of matrix Sturm-Liouville (SL) problems in terms of eigenvalues of certain unitary matrices. Based on this characterization the team of Atkinson, Krall, Leaf, and Zettl (AKLZ) developed an algorithm, and constructed a prototype FORTRAN code for the numerical computation of eigenvalues of SL problems. A description of the algorithm, with some test results, was published in an Argonne Laboratory Report [2]. While clearly demonstrating the feasibility of this algorithm, the tests showed that there were some difficulties remaining. Dwyer, in his dissertation as a student of Zettl, refined and significantly extended the algorithm, and corrected the difficulties in the original code.

In this paper, the completed algorithm for computing the eigenvalues for a regular Sturm-Liouville problem with separated boundary conditions is presented.

Problem Definitions

A SL problem consists of the second order linear ordinary differential equation

$$-(py')' + qy = \lambda wy \text{ on } (a,b) \tag{1}$$

* *1991 Mathematics Subject Classifications:* 34B24, 34L15, 34L05.
Key words and phrases: Sturm-Liouville problem, matrix coefficients, eigenvalues, numerical computation of eigenvalues

Submitted: August 25, 1994.

together with boundary conditions. For the case when both endpoints a, b are regular, these have the form

$$C \begin{pmatrix} y(a) \\ (py')(a) \end{pmatrix} + D \begin{pmatrix} y(b) \\ (py')(b) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{2}$$

Theorem 1 *Let p, q, w be complex $m \times m$ matrix functions defined on the interval $[a,b]$, $-\infty < a < b < \infty$, satisfying the following conditions:*

1. *The matrix $p(t)$ is invertible for almost all $t \in (a,b)$.*
2. *Each component of p^{-1}, q, w is in $L^1(a,b)$.*
3. *The matrices $p(t)$, $q(t)$, $w(t)$ are hermitian for almost all $t \in (a,b)$.*
4. *The matrices $p(t)$ and $w(t)$ are positive definite for almost all $t \in (a,b)$.*

Assume that the complex constant $2m \times 2m$ matrices C and D satisfy:

5. *The $2m \times 4m$ matrix $(C|D)$ has full rank.*
6. *$CJC^* = DJD^*$, where $J = \left(\begin{array}{c|c} 0 & I_m \\ \hline -I_m & 0 \end{array}\right)$ and I_m denotes the $m \times m$ identity matrix.*

Then the boundary value problem, (1),(2) is self-adjoint; its eigenvalues are all real, there are countably many of them $\{\lambda_n : n \in N_0 = \{0,1,2,\cdots\}\}$, and they can be ordered to satisfy

$$-\infty < \lambda_0 \le \lambda_1 \le \lambda_2 \le \cdots \ , \quad \textit{with} \ \ \lambda_n \to \infty \ \ \textit{as} \ \ n \to \infty.$$

Proof. See Möller and Zettl [7]. □

If condition (4) is weakened so that $p(t)$ is required to be invertible, but is not required to be positive definite, then a similar result holds, except that the spectrum will not, in general, be bounded below.

We are interested in the special case when the boundary conditions are *separated*. This means that the matrices C and D can each be partitioned into four $m \times m$ square matrices, as:

$$C = \left(\begin{array}{c|c} A_1 & A_2 \\ \hline 0 & 0 \end{array}\right) \ , \ D = \left(\begin{array}{c|c} 0 & 0 \\ \hline B_1 & B_2 \end{array}\right).$$

In this case, the boundary conditions (2) reduce to

$$A_1 y(a) + A_2 (py'(a)) = 0 \tag{3}$$
$$B_1 y(b) + B_2 (py'(b)) = 0 \tag{4}$$

where A_1, A_2, B_1, and B_2 are complex constant $m \times m$ matrices which satisfy

1. $rank(A_1|A_2) = rank(B_1|B_2) = m$
2. $A_1 A_2^* = A_2 A_1^*$
3. $B_1 B_2^* = B_2 B_1^*$.

Below we consider the question: How can the eigenvalues λ_n be computed numerically? The codes SLEDGE [9], and SLEIGN [3],[4] , as well as the codes in the NAG library [10], address this question only for the special case when $m = 1$ (scalar case).

Shooting Methods for Scalar Problems

Before beginning on the matrix problem, we consider the scalar problem. The boundary conditions specify the ratio of the values of the solution y and the quasi-derivative py' at each of the endpoints. We define

$$z(x) = \frac{y(x)}{(py')(x)}. \tag{5}$$

Since the solution is nontrivial, y and py' cannot be simultaneously zero. However, it is quite likely that py' will be zero at some points along the interval, leading to unbounded values for the ratio z. To avoid the difficulties of unbounded values, we might use a mapping which transforms the real line into some bounded set; applying the mapping to the ratio z will produce a variable whose initial value, at endpoint a, is also determined by the boundary condition, and which avoids the difficulties caused by the division by zero.

If we use the inverse-tangent mapping, the result is the Prüfer transformation (see [5]). The "Prüfer angle" is defined by

$$\Theta(x) = \arctan(z) = \arctan\left(\frac{y(x)}{(py')(x)}\right).$$

This variable satisfies a certain first order ordinary differential equation. Using this equation, and the initial value determined by the boundary condition at the endpoint a, it is possible to compute a numerical approximation to the resulting value for Θ at the other endpoint, b. Note that the initial value of Θ depends only on the boundary condition, but at other points in the interval the value will depend on the parameter λ. The boundary condition at b determines "target" values; if the value of Θ matches the target, we have found an eigenvalue. The success of the search algorithm depends on our ability to do three things:

1. Recognize that the value λ is an eigenvalue.
2. Determine whether a trial value of λ is "near" an eigenvalue, and whether it is too high or too low.

3. Determine *which* eigenvalue the trial value is close to.

With appropriate care about which branch of the inverse tangent is used, it can be shown that if $p(t)$ is positive for all t in the interval, then the value $\Theta(b;\lambda)$ is monotone increasing in λ, which allows a simple bisection search to be used to approximate the eigenvalue. More advanced algorithms (see [8]) use more efficient search methods, such as Newton's method.

Another possibility is to use a mapping from complex analysis to map the real line onto some bounded curve in the complex plane. The mapping

$$\Theta = \frac{1+iz}{1-iz}$$

takes the real axis onto the unit circle, with $z = \pm\infty$ mapped to $\Theta = -1$. This leads somewhat naturally to

$$\begin{aligned}
\Theta &= \frac{1+iz}{1-iz} \\
&= \frac{1+i\frac{y}{py'}}{1-i\frac{y}{py'}} \\
&= \frac{py'+iy}{py'-iy} \\
&= (py'+iy)(py'-iy)^{-1}.
\end{aligned}$$

Of course, even in the scalar case, and particularly in the matrix case, there are some details which must be addressed; the discussion above is meant only as a motivation for what follows.

A Shooting Method for Matrix Problems

Although it is possible to define an inverse-tangent function for matrix variables, the result is not particularly useful to us here. The difficulty is that the corresponding sine and cosine functions do not have the desirable derivative properties of their scalar counterparts, introducing difficulties in the formulation of a first order differential equation.

Instead, we will develop a method based on the second approach mentioned above. Let $Y(x)$ be a square matrix function such that

$$-(pY')' + qY = \lambda wY$$

satisfying the initial conditions at endpoint a:

$$Y(a) = -A_2^* \ , \ \ (pY')(a) = A_1^*.$$

The assumptions on the coefficients guarantee that such a solution exists, and is defined on the entire interval $[a,b]$.

We can write an equivalent system of first order equations as follows: let $U = Y$, and $V = pY'$, and we have

$$\begin{aligned} U' &= p^{-1}V \\ V' &= rU \text{ , where} \\ r(x,\lambda) &= q(x) - \lambda w(x) \end{aligned}$$

subject to:

$$\begin{aligned} U(a) &= -A_2^* \\ V(a) &= A_1^*. \end{aligned}$$

We are now ready to define the matrix Atkinson [1] calls theta.

Definition 1 *For each $x \in [a,b]$, and for each real λ, define*

$$\Theta(x,\lambda) = (V(x) + iU(x))(V(x) - iU(x))^{-1} \tag{6}$$

for U and V as defined above.

The dependence on λ arises because U and V are solutions to a system whose coefficients depend on λ. Having defined the matrix, we must verify that it will exist for a SL problem with separated, self-adjoint boundary conditions. We will need the following lemmas.

Lemma 2 *Suppose that A_1 and A_2 are complex, square $m \times m$ matrices such that*

1. *$rank(A_1|A_2) = m$*
2. *$A_1A_2^* = A_2A_1^*$*

then the matrix $(A_1 - iA_2)$ is invertible.

Proof. We begin by noting that $(A_1 - iA_2)$ is invertible if, and only if, $(A_1 - iA_2)(A_1 - iA_2)^*$ is invertible.

$$\begin{aligned} (A_1 - iA_2)(A_1 - iA_2)^* &= (A_1 - iA_2)(A_1^* + iA_2^*) \\ &= A_1A_1^* + A_2A_2^* - i(A_2A_1^* - A_1A_2^*) \\ &= A_1A_1^* + A_2A_2^*. \end{aligned}$$

Assume that the sum $A_1A_1^* + A_2A_2^*$ is singular. Then there is some nonzero vector v such that

$$\begin{aligned} (A_1A_1^* + A_2A_2^*)v &= 0 \\ v^*(A_1A_1^* + A_2A_2^*)v &= 0 \\ v^*A_1A_1^*v + v^*A_2A_2^*v &= 0 \end{aligned}$$

which is possible only if $v^*A_1 = 0$ and $v^*A_2 = 0$. However, this would imply that

$$v^*(A_1|A_2) = 0$$

which is a contradiction since $(A_1|A_2)$ is of full rank, by assumption, and $v^* \neq 0$. We conclude that no such v exists, so the sum is nonsingular and $(A_1 - iA_2)$ is invertible. □

Lemma 3 *([1]) For U, V as defined above, if U^*V is hermitian at any $x_0 \in [a,b]$, then U^*V is hermitian for all $x \in [a,b]$.*

Proof. See [1]. □

Lemma 4 *([1]) For U, V as defined above, suppose that U^*V is hermitian for all $x \in [a,b]$, and suppose that $(V - iU)$ is invertible at some $x_0 \in [a,b]$. Then $(V - iU)$ is invertible for all $x \in [a,b]$.*

Proof. See [1]. □

Theorem 5 *For a regular SL problem with separated, self-adjoint boundary conditions, the matrix $\Theta(x,\lambda)$ exists for all $x \in [a,b]$, for any real λ.*

Proof. At the point $x_0 = a$, we have $U(x_0)^*V(x_0) = -A_2A_1^*$. By assumption, this matrix is hermitian. Applying Lemma 3, we see that U^*V is hermitian for all $x \in [a,b]$. At the point $x_0 = a$, we have $V(x_0) - iU(x_0) = (A_1 - iA_2)^*$, which is invertible by Lemma 2. Finally, by Lemma 4, we see that $(V - iU)$ is invertible for all $x \in [a,b]$, so we conclude that the matrix Θ may be computed for every $x \in [a,b]$. □

It is easy to verify that the matrix $\Theta(x,\lambda)$ is unitary for all $x \in [a,b]$, for any real value of λ.

The boundary condition at the endpoint a provides an initial value for theta; in order to compute theta at other points along the interval, we develop a first order differential equation which has theta as a solution.

Theorem 6 *([1]) For $\Theta(x,\lambda)$ defined above, we have for any real λ*

$$\frac{\partial}{\partial x}\Theta(x,\lambda) = i\Theta(x,\lambda)\Omega(x,\lambda)$$

where $\Omega(x,\lambda)$ is defined by

$$\Omega(x,\lambda) = \frac{1}{2}((\Theta + I)^*p^{-1}(x)(\Theta + I) - (\Theta - I)^*r(x,\lambda)(\Theta - I)).$$

Proof. See [1]. □

The first requirement for a method is that we must be able to recognize that a value λ is an eigenvalue. It is convenient to define a new matrix, constructed by "scaling" the matrix $\Theta(x,\lambda)$ by a constant.

Definition 2 *For Θ as defined above, let*

$$B(x,\lambda) = (B_1 - iB_2)^{-1}(B_1 + iB_2)\Theta(x,\lambda)$$

where B_1 and B_2 are the constant matrices used to define the boundary condition at the endpoint b.

Note that the factor $(B_1 - iB_2)^{-1}$ exists, by Lemma 2. It is easy to verify that the matrix B is also unitary.

Lemma 7 *([2]) For U, V as defined above, we have:*

1. *A real value λ is an eigenvalue of the SL problem if, and only if, the matrix $B_1U(b) + B_2V(b)$ is singular.*
2. *The multiplicity of λ as an eigenvalue of the SL problem is equal to the dimension of the null space of $B_1U(b) + B_2V(b)$.*

Proof. See [2]. □

Theorem 8 *For $B(x,\lambda)$ as defined above, we have*

1. *A real value λ is an eigenvalue for the SL problem if, and only if, the number 1 is an eigenvalue of the matrix $B(b,\lambda)$.*
2. *The multiplicity of λ as an eigenvalue of the SL problem is equal to the multiplicity of the value 1 as an eigenvalue of the matrix $B(b,\lambda)$.*

Proof. (1) Suppose that 1 is an eigenvalue of $B(b,\lambda)$. Then there is a nonzero constant vector c such that $B(b,\lambda)c = 1c$. Hence

$$(B_1 - iB_2)^{-1}(B_1 + iB_2)(V(b) + iU(b))(V(b) - iU(b))^{-1}c = c.$$

Define $z = (V(b) - iU(b))^{-1}c$, and note that $z \neq 0$ and $c = (V(b) - iU(b))z$. Hence

$$\begin{aligned}(B_1 - iB_2)^{-1}(B_1 + iB_2)(V(b) + iU(b))z &= (V(b) - iU(b))z \\ (B_1 + iB_2)(V(b) + iU(b))z &= (B_1 - iB_2)(V(b) - iU(b))z \\ 2i(B_1U(b) + B_2V(b))z &= 0.\end{aligned}$$

Since we know $z \neq 0$, we conclude that $B_1U(b) + B_2V(b)$ is singular, and the result follows from Lemma 7. Conversely, if we know that λ is an eigenvalue of the SL porblem, we reverse the steps to produce an eigenvector for $B(b,\lambda)$ corresponding to an eigenvalue of 1.

Proof. (2) The multiplicity of 1 is the number of linearly independent choices for the vector c, and hence the number of linearly independent choices for the vector z, which is the dimension of the null space of $B_1U(b) + B_2V(b)$. The result is now immediate from Lemma 7. □

The second requirement for a method is that we must be able to determine if a trial value for λ is "close" to an actual eigenvalue, and whether it is too high or too low. To do this, we will need the following lemma, which is proved by Atkinson:

Lemma 9 *([1]) Suppose that t is a real variable, $\Theta(t)$ is unitary, $\Omega(t)$ is hermitian, continuous, and positive definite, and suppose that Θ satisfies*

$$\frac{d}{dt}\Theta = i\Theta\Omega$$

then as t increases the eigenvalues of Θ move anti-clockwise around the complex unit circle.

Proof. See [1]. □

It can be shown (see [1]) that, for any fixed choice of x, the matrix $\Theta(x, \lambda)$ satisfies a (partial) differential equation in the variable λ of the form required by Lemma 9. Since the matrix B is simply Θ multiplied by a constant, the same conclusions hold for B; for $x = b$ we have the eigenvalues of $B(b, \lambda)$ moving anti-clockwise around the complex unit circle as the parameter λ is increased. For any trial value of λ, we examine the position of the eigenvalues of $B(b, \lambda)$ and we can readily determine if the trial value is "close", and whether it is too high or too low. A simple bisection search procedure will quickly determine the required eigenvalue to any precision we might require.

The third requirement for a method, that we can determine *which* eigenvalue we are close to, is more difficult to satisfy. We will discuss this issue in the next section.

Implementation Details

The team of Atkinson, Krall, Leaf, and Zettl (AKLZ) developed a prototype FORTRAN code [2], based on the algorithm described above. This early code produced excellent results for some trial problems, but had difficulties in certain cases. A later implementation by Dwyer duplicated the earlier results, and the difficulties as well. These difficulties were caused by some details in the way that the algorithm was implemented, and have since been corrected; the algorithm itself is perfectly sound. A corrected FORTRAN code, developed by Dwyer and Zettl, is available as an appendix to the doctoral thesis of Dwyer [6].

The initial value problem (IVP) for Θ may be solved using any of the standard techniques. It is necessary to keep track of the eigenvalues of the matrix

$B(x,\lambda)$ as the variable x increases from a to b. The numerical method for the IVP solution uses a step-size which is rather small, to increase the accuracy of the computed result. However, since eigenvalue computations are relatively slow, we do not wish to examine the eigenvalues at every step; the code is written to perform the eigenvalue computations only at selected steps. This modification saves considerable execution time. The algorithm has been implemented using both the NAG and IMSL linear algebra libraries, with similar results.

The matrix $B(x,\lambda)$ is sampled at equally spaced points along the interval $[a,b]$. At each point, the eigenvalues of B are computed using a library routine. We compare the lists of eigenvalues for adjacent samples of B to determine whether the eigenvalues have passed through the value 1 on the complex unit circle. A difficulty arises in this comparison if the two lists for adjacent samples are not returned by the library routine in the same order. At various points along the interval, the matrix B will change enough that the library routine will "shuffle" the eigenvalue list; the list will contain eigenvalues which are similar to the previous list, but in a different order. This re-ordering cannot be avoided by reducing the distance between samples. It is necessary to correctly match the two lists, before the eigenvalue movements can be tracked. Since the lists are usually *not* shuffled, the matching routine should be designed to quickly recognize this situation. An algorithm for matching the lists is described in detail in [6].

The movements of the eigenvalues of $B(x,\lambda)$ are tracked as x increases from a to b. We keep track of the total number of times that an eigenvalue passes anti-clockwise through the value 1; we will call this total the *index* for the particular choice of λ. As we increase the λ value, we will get a "jump" in the index value as λ passes through an eigenvalue of the SL problem. The size of the jump indicates the multiplicity of the eigenvalue. The first (lowest) eigenvalue for the SL problem corresponds to the first jump in the index. However, the first jump need not be a jump from index=0 to index=1. The eigenvalues of $B(x,\lambda)$ do not, in general, move anti-clockwise as x increases; each may behave differently, and may move clockwise during some portion of the interval $[a,b]$. Thus, it is possible to have a negative index value in some problems. Other problems may have an index which is never zero, since it is possible that the first eigenvalue may correspond to the jump from index=5 to index=6.

If a lower bound on the eigenvalues is known, then we may solve the labeling difficulty by computing the index corresponding to the lower bound and using this value as an offset. We may then compute any desired eigenvalue, λ_n, by searching for the index jump from index=offset+n to index=offset+n+1. We continue a bisection search until the λ-interval $[\lambda_{lo},\lambda_{hi}]$ is sufficiently narrow to meet the user specified tolerance, where λ_{lo} is a value whose index is less than or equal to offset+n, and λ_{hi} is a value whose index is greater than offset+n. In cases where the spectrum is not bounded below, we use the convention that λ_0 is the first nonnegative eigenvalue. This is achieved by computing the index corresponding to $\lambda = 0$, and using that value as offset.

After a bracketing interval has been found using bisection, the multiplicity of the eigenvalue is simply the size of the jump in the index across the bracket. If the user has specified a very tight or a very loose tolerance on the eigenvalue, the multiplicity which is computed may be incorrect. What is actually being computed is the total number of eigenvalues which lie within the bracket. If the tolerance is larger than the spacing between adjacent eigenvalues, they may both be included within the same bracket and appear as one multiple eigenvalue. The reported multiplicity will thus be too large. Conversely, if the tolerance is too tight, the reported multiplicity may be too low; due to numerical errors, an eigenvalue of multiplicity may appear to be a cluster of simple eigenvalues, very close together. If the tolerance is tighter than the spacing of this cluster, or if the cluster is very near to one of the endpoints of the bracket, some of the simple eigenvalues will be excluded from the bracket, resulting in a multiplicity count which is too low.

Experiments indicate that the best method is to work with a moderate tolerance for initial computations, determine the multiplicity using that tolerance, and then continue the computation to a tighter tolerance. If the multiplicities do not match, we are alerted that some additional care is required. The next example was selected to illustrate some of these difficulties.

Example

We consider the following three-dimensional problem:

Example 1 *On the interval* $[0, \pi]$*, with coefficients*

$$P(x) = \begin{pmatrix} 11 & 6 & 3 \\ 6 & 12 & 2 \\ 3 & 2 & 1 \end{pmatrix}, \; Q(x) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \; W(x) = \begin{pmatrix} 38 & 24 & 12 \\ 24 & 18 & 8 \\ 12 & 8 & 4 \end{pmatrix}$$

subject to the condition $Y(0) = Y(\pi) = 0$.

It is easy to verify that both P and W are positive definite. The problem may be solved without difficulty, so that we know exactly what the eigenvalues are; on this interval they are either integers or integers plus one fourth. The example was constructed so that there are eigenvalues with each of the possible multiplicities. The first ten eigenvalues are given in Table 1. The accuracy which can be achieved will, of course, depend on the particular numerical methods used. A relatively straight-forward implementation of the algorithm in FORTRAN, using double precision arithmetic, produced the results shown in Table 2. The bisection search produces an interval estimate; the midpoint of the interval is used as the estimate for the eigenvalue. Each search began with a random real value. Each interval was refined until the relative error is less than 0.1% .

n	λ_n	n	λ_n
0	0.25	5	4.00
1	1.00	6	4.00
2	1.00	7	6.25
3	2.25	8	9.00
4	4.00	9	9.00

Table 1: Exact Values For Eigenvalues

n	Interval	Midpoint	Mult.
0	(0.24991 , 0.250968)	0.250479	1
1	(0.999991 , 1.000968)	1.000479	2
2	”	”	
3	(2.249014 , 2.250968)	2.249991	1
4	(3.997061 , 4.000968)	3.999014	3
5	”	”	
6	”	”	
7	(6.247061 , 6.250968)	6.249014	1
8	(8.993155 , 9.000968)	8.997061	2
9	”	”	

Table 2: Results, Beginning With Random Real Guess Values

If the computations are repeated using an integer as the initial guess, each eigenvalue will be one of the endpoints of the bisection interval. The results of these computations are shown in Table 3. Even a small numerical error will split the multiple eigenvalues across the endpoint, resulting in errors in the reported multiplicities. Note that each reported eigenvalue *is* within tolerance of the correct value; it is only the reported multiplicity which is in error. Similar errors appear when the tolerance is specified to be very small.

As a final test, we computed the first and second eigenvalues, beginning with a random real initial guess, and allowed the bisection process to continue until an incorrect bisection step occurred. Beginning with an interval of length 1, the code made 32 correct bisections when computing the first eigenvalue, and 21 correct bisections before the second eigenvalue “split” and lost its multiplicity. These results are shown in Table 4. These results indicate that the values of the eigenvalues may be computed quite accurately using this algorithm. However, when computations are continued this far the reported multiplicity is almost always in error; multiple eigenvalues are reported as a “cluster” of closely spaced simple eigenvalues.

n	Interval	Midpoint	Mult.
0	(0.250000 , 0.250977)	0.250488	1
1	(0.999023 , 1.000000)	0.999512	1
2	(1.000000 , 1.000977)	1.000488	1
3	(2.250000 , 2.251953)	2.250977	1
4	(3.996094 , 4.000000)	3.998047	1
5	(4.000000 , 4.003906)	4.001953	2
6	”	”	
7	(6.250000 , 6.253906)	6.251953	1
8	(8.992188 , 9.000000)	8.996094	1
9	(9.000000 , 9.007813)	9.003906	1

Table 3: Results, Beginning With Integer Guess Values

n	Interval	Midpoint
0	(0.249999999889 , 0.250000000121)	0.250000000005
1	(0.999999728102 , 1.000000204939)	0.999999966521

Table 4: Results of Continued Bisection

References

[1] F.V. Atkinson, *Discrete and Continuous Boundary Problems*, Academic Press, New York, 1964.

[2] F.V. Atkinson, A.M. Krall, G.K. Leaf, A. Zettl, *On the Numerical Computation of Eigenvalues of Matrix Sturm-Liouville Problems with Matrix Coefficients*, Argonne National Laboratory Reports, Darien, 1987.

[3] P.B. Bailey, *SLEIGN: An Eigenfunction-eigenvalue Code for Sturm-Liouville Problems*, SAND7-2044, Sandia Laboratories, Albuquerque, 1978.

[4] P.B. Bailey, M.K. Gordon, L.F. Shampine, *Automatic Solution of the Sturm-Liouville Problem*, ACM Trans. Math. Software, 4, 1978, 193-208.

[5] E.A. Coddington, N. Levinson, *Theory of Ordinary Differential Equations*, Krieger Publishing, Malabar, 1984.

[6] H.I. Dwyer, *Eigenvalues of Matrix Sturm-Liouville Problems with Separated or Coupled Boundary Conditions*, Doctoral Thesis, Northern Illinois University, 1993.

[7] M. Möller, A. Zettl, *Semi-boundedness of Ordinary Differential Operators*, Journal of Differential Equations, (to appear).

[8] P.B.Bailey, B.S.Garbow, H.G.Kaper and A.Zettl, *Eigenvalue and Eigenfunction Computations for Sturm-Liouville Problems*, ACM TOMS 17(1991), 491-499.

[9] S.Pruess and C.Fulton, *Mathematical Software for Sturm-Liouville Problems*, ACM TOMS (1994), (to appear).

[10] J. Pryce, *D02KEF*, NAG Library Reference Guide.

H.I. Dwyer
Mathematics Department
University of Wisconsin
Platteville, Wi. 53818
E-mail address: dwyer@uwplatt.edu

A. Zettl
Mathematical Sciences Dept.
Northern Illinois University
DeKalb, IL. 60115
E-mail address: zettl@math.niu.edu