Solutions of Direct and Inverse Sturm–Liouville Problems

Bodhiya Baduge Upeksha Piyaruwani Perera

Univ.-Diss.
zur Erlangung des akademischen Grades

"doctor rerum naturalium"
(Dr. rer. nat.)

in der Wissenschaftsdisziplin "Numerische Mathematik"

eingereicht an der
Mathematisch-Naturwissenschaftlichen Fakultät
Institut für Mathematik
der Universität Potsdam

Ort und Tag der Disputation: Potsdam, den 29. November 2021

Hauptbetreuer*in:

apl. Prof. Dr. rer. nat. habil. Christine Böckmann
Institut für Mathematik, Universität Potsdam

Gutachter*innen:

Prof. Dr. Melina Freitag
Institut für Mathematik, Universität Potsdam

Prof. Dr. Đinh Nho Hào
Hanoi Institute of Mathematics, Vietnam Academy of Science and Technology

Published online on the
Publication Server of the University of Potsdam:
https://doi.org/10.25932/publishup-53006
https://nbn-resolving.org/urn:nbn:de:kobv:517-opus4-530064

Abstract

Lie group method in combination with Magnus expansion is utilized to develop a universal method applicable to solving a Sturm–Liouville Problem (SLP) of any order with arbitrary boundary conditions. It is shown that the method has ability to solve direct regular and some singular SLPs of even orders (tested up to order eight), with a mix of boundary conditions (including non-separable and finite singular endpoints), accurately and efficiently.

The present technique is successfully applied to overcome the difficulties in finding suitable sets of eigenvalues so that the inverse SLP problem can be effectively solved.

Next, a concrete implementation to the inverse Sturm–Liouville problem algorithm proposed by Barcilon (1974) is provided. Furthermore, computational feasibility and applicability of this algorithm to solve inverse Sturm–Liouville problems of order $n = 2, 4$ is verified successfully. It is observed that the method is successful even in the presence of significant noise, provided that the assumptions of the algorithm are satisfied.

In conclusion, this work provides methods that can be adapted successfully for solving a direct (regular/singular) or inverse SLP of an arbitrary order with arbitrary boundary conditions.

Zusammenfassung

Die Lie-Gruppen-Methode in Kombination mit der Magnus-Expansion wird verwendet, um eine universelle Methode zu entwickeln, die zur Lösung eines Sturm-Liouville-Problems (SLP) beliebiger Ordnung mit beliebigen Randbedingungen anwendbar ist. Es wird gezeigt, dass die Methode in der Lage ist, direkte reguläre und einige singuläre SLPs gerader Ordnung (getestet bis zur 8. Ordnung) mit einer Mischung von Randbedingungen (einschließlich nicht trennbarer und endlicher singulärer Endpunkte) genau und effizient zu lösen.

Die vorliegende Technik wird erfolgreich angewendet, um die Schwierigkeiten beim Finden geeigneter Sätze von Eigenwerten zu überwinden, so dass das inverse SLP-Problem effektiv gelöst werden kann.

Als nächstes wird eine konkrete Implementierung des von Barcilon (1974) vorgeschlagenen inversen Sturm-Liouville-Problemalgorithmus bereitgestellt. Weiterhin wird die rechnerische Durchführbarkeit und Anwendbarkeit dieses Algorithmus zur Lösung inverser Sturm-Liouville-Probleme der Ordnung $n = 2, 4$ erfolgreich verifiziert. Es wird beobachtet, dass das Verfahren selbst bei Vorhandensein von signifikantem Rauschen erfolgreich ist, vorausgesetzt, dass die Annahmen des Algorithmus erfüllt sind.

Zusammenfassend stellt diese Arbeit Methoden zur Verfügung, die erfolgreich zur Lösung eines direkten (regulär/singulären) oder inversen SLP beliebiger Ordnung mit beliebigen Randbedingungen angepasst werden können.

Acknowledgments

First and foremost, I would like to express my sincere gratitude to my advisor apl. Prof. Dr. rer. nat. habil. Christine Böckmann for the continuous support of my Ph.D study and related research, for her patience, motivation, and immense knowledge. She stood by me through the challenging and difficult times I had to go through and always had a solution for each and every obstacle I had to face and showed unlimited sympathy and understanding.

I would like to thank the faculty, visiting scholars and the staff for their support. I extend my gratitude towards my home university, University of Kelaniya, Sri Lanka, for providing the financial support and granting study leave to pursue my degree. And my heartfelt gratitude goes towards each and every person that made me and my family's stay in Germany affordable, memorable and worthwhile, specially the staff at Welcome Center of Potsdam University for their unfathomable support.

Last but not the least, I would like to thank my family: my husband - Manjula for helping me find my adorable supervisor and always pressing me and reminding me that I am doing a PhD; my darling son - Imeth for being brave enough to let me go and do my work alone in Germany; my mother - Yasawathie for taking care of my son while I was away (and even when I am there) and being a better mom than I ever can be for my precious one; my father - Felix for although hardly expressed in words having silently supporting me and having faith in me; and my twin sister Kaushalya for being a second mom to my son while I was away. I love you all, for supporting me throughout - without failing and sacrificing your time and dreams, adopting my dreams as your own, so that even though it's so late I could eventually make my dreams come true.

Contents

1 Introduction

In this thesis, Lie group method in combination with Magnus expansion is utilized to develop a universal method applicable to solving a Sturm–Liouville Problem (SLP) of any order with arbitrary boundary conditions. Furthermore, a concrete implementation to the inverse Sturm–Liouville algorithm proposed by Barcilon [16] is provided.

To begin with, the current chapter states some basic definitions, theorems and proofs.

The rest of the thesis is organized as follows: Chapter 2 based on [18] elaborates the proof of the uniqueness theorem for inverse eigenvalue problems. Chapters 3 and 4 which are based on two own publications [92] and [93], discuss direct and inverse Sturm–Liouville problems, respectively, and the methods of solution. Chapter 5 concludes the thesis by discussing the pros and cons of the methods and providing useful insights to future developments.

1.1 Definitions and Theorems

▶ **Definition 1.1. Linear Differential Equations of Order n.**
Let $a_0, a_1, \ldots, a_n$ are $n+1$ continuous (complex) functions defined on a real x interval I, and let $\mathcal{L}_n$ denote the differential operator

$$\mathcal{L}_n = a_0 \frac{d^n}{dx^n} + a_1 \frac{d^{n-1}}{dx^{n-1}} + \ldots + a_n \,. \tag{1.1}$$

That is, if u is any function possessing n derivatives on I,

$$\mathcal{L}_n u = a_0 u^{(n)} + a_1 u^{(n-1)} + \ldots + a_n u \,.$$

If $a_0(x) \neq 0$, for any $x \in I$, then

$$\mathcal{L}_n u = 0 \implies u^{(n)} + p_1 u^{(n-1)} + \ldots + p_n u = 0 \,. \tag{1.2}$$

This is a linear homogeneous differential equation of order n.

Linear in the sense that p_i's are functions of x only, and no product terms involving dependent variable u, and homogeneous as there are no terms independent of u. And the order is defined to be the highest derivative in the differential equation. ◀

▶ **Definition 1.2. Inner Product**.
If $f, g \in \mathfrak{L}^2(a, b)$ (the set of all complex-valued functions f on $a \leq x \leq b$ which are Lebesgue-measurable),

$$\langle f, g \rangle = \int_a^b f g^* dx,$$

is called the inner product of f with g. Here g^* denotes the complex conjugate of g. ◀

▶ **Definition 1.3. Adjoint Equations**.
Consider the differential operator $\mathcal{L}_n$ defined in Equation (1.1):

$$\mathcal{L}_n = a_0 \frac{d^n}{dx^n} + a_1 \frac{d^{n-1}}{dx^{n-1}} + \ldots + a_n$$

The adjoint of $\mathcal{L}_n$ is given by

$$\mathcal{L}_n^* = (-1)^n \frac{d^n}{dx^n}(a_0^* \cdot) + (-1)^{n-1} \frac{d^{n-1}}{dx^{n-1}}(a_1^* \cdot) + \ldots + a_n^* \cdot \tag{1.3}$$

The equation

$$\mathcal{L}_n^* u = 0 \qquad (x \in I)$$

is called the adjoint equation to $\mathcal{L}_n u = 0$ on I, is defined to be the problem of finding a function ψ (a solution) on I such that $a_k^* \psi$ $(k = 0, 1, \ldots, n)$ has $n - k$ derivatives on I and satisfying

$$(-1)^n (a_0^* \psi)^{(n)} + (-1)^{n-1} (a_1^* \psi)^{(n-1)} + \ldots + a_n^* \psi = 0$$

on I. ◀

Proof.

$$\mathcal{L}[y] = a_0(x) y^{(n)} + a_1(x) y^{(n-1)} + \ldots + a_n(x) y$$

$$\implies \mathcal{L} \equiv a_0(x)\frac{d^n}{dx^n} + a_1(x)\frac{d^{n-1}}{dx^{n-1}} + \ldots + a_{n-1}(x)\frac{d}{dx} + a_n(x)$$

$$\begin{aligned}\langle v, \mathcal{L}y\rangle &= \int v^*\Big(a_0(x)y^{(n)} + a_1(x)y^{(n-1)} + \ldots + a_n(x)y\Big)dx \\ &= \int v^* a_0(x)y^{(n)}dx + \int v^* a_1(x)y^{(n-1)}dx + \ldots + \int v^* a_{n-1}(x)y'dx \\ &\quad + \int v^* a_n(x)ydx\end{aligned}$$

We have:

$$\int v^* a_{n-1}(x)y'dx = v^* a_{n-1}(x)y|_a^b - \int y(v^* a_{n-1}(x))'dx = BCs - \int y(va_{n-1}(x)^*)'^*dx$$

$$\int v^* a_{n-2}(x)y'dx = BCs + \int y(v^* a_{n-2}(x))''dx = BCs + \int y(va_{n-2}(x)^*)''^*dx$$

$$\vdots$$

$$\int v^* a_1(x)y^{(n-1)}dx = BCs + (-1)^{n-1}\int y((va_1(x)^*)^{(n-1)})^*dx$$

$$\int v^* a_0(x)y^{(n)}dx = BCs + (-1)^n\int y((va_0(x)^*)^{(n)})^*dx$$

Adding up we have:

$$\begin{aligned}\langle v, \mathcal{L}y\rangle =& BCs + (-1)^n\int y((va_0(x)^*)^{(n)})^*dx + (-1)^{n-1}\int y((va_1(x)^*)^{(n-1)})^*dx + \ldots \\ &- \int y(va_{n-1}(x)^*)'^*dx + \int (va_n(x)^*)^* ydx\end{aligned}$$

$$\begin{aligned}\implies \mathcal{L}^*[y] &= \frac{d^n}{dx^n}(a_0(x)^* y) - \frac{d^{n-1}}{dx^{n-1}}(a_1(x)^* y) + \ldots + (-1)^{n-1}\frac{d}{dx}(a_{n-1}(x)^* y) + (-1)^n a_n(x)^* y \\ &= \sum_{k=0}^{n}(-1)^k \mathcal{D}^k[a_{n-k}(x)^* y]\end{aligned}$$

So adjoint operator of $\mathcal{L}[y] = \sum_{k=0}^{n} a_{n-k}(x)\mathcal{D}^k[y]$ is $\mathcal{L}^*[y] = \sum_{k=0}^{n}(-1)^k\mathcal{D}^k[a_{n-k}(x)^* y]$. ■

▶ **Definition 1.4. Eigenvalue Problems.**

Let $\mathcal{L}_n$ be the nth-order operator given by

$$\mathcal{L}_n = a_0 \frac{d^n}{dx^n} + a_1 \frac{d^{n-1}}{dx^{n-1}} + \ldots + a_n$$

where the a_i are complex-valued functions on the closed interval $a \leq x \leq b$. Let

$$U_j u = \sum_{k=1}^{n} (M_{jk} u^{(k-1)}(a) + N_{jk} u^{(k-1)}(b)), \quad j = 1, \ldots, n$$

where M_{jk}, N_{jk} are constants. Denote the relationships $U_j u = 0$, $j = 1, \ldots, n$, by $Uu = 0$. The problem,

$$\Pi : \quad \mathcal{L}_n u = \lambda u, \quad Uu = 0 \tag{1.4}$$

is called an eigenvalue problem.
The element $u \neq 0$ is called the eigenvector and the number λ is called an eigenvalue.
The set of all eigenvalues satisfying the equation (1.4) is called the spectrum and denoted by $\rho(\mathcal{L}_n)$. ◀

▶ **Definition 1.5. Green's Formula**.
If u, v are any two functions on I possessing n derivatives, then for any $a, b \in I$,

$$\langle v, \mathcal{L}_n u \rangle - \langle u, \mathcal{L}_n^* v \rangle = [uv](a) - [uv](b) \tag{1.5}$$

where

$$[uv](x) = \sum_{m=1}^{n} \sum_{j+k=m-1} (-1)^j u^{(k)}(x) (a_{n-m} v^*)^{(j)}(x) \tag{1.6}$$

or

$$[uv](x) = \sum_{j,k=1}^{n} B_{jk}(x) u^{(k-1)}(x) v^{*(j-1)}(x) \tag{1.7}$$

with

$$\mathbf{B}(x) = \begin{pmatrix} B_{11} & B_{12} & \cdot & \cdot & a_0(x) \\ \cdot & \cdot & -a_0(x) & & \\ \cdot & \cdot & & & \\ \cdot & \cdot & & 0 & \\ (-1)^{n-1}a_0(x) & & & & \end{pmatrix} \tag{1.8}$$

Note that $\det \mathbf{B}(x) = (a_0(x))^n \neq 0$. ◀

▶ **Definition 1.6. Self-adjoint Eigenvalue Problems.**
An eigenvalue problem defined as in (1.4) is said to be self-adjoint if

$$\langle \mathcal{L}_n u, v \rangle = \langle u, \mathcal{L}_n v \rangle \tag{1.9}$$

for all $u, v \in C^n[a, b]$ which satisfy the boundary conditions:

$$Uu = Uv = 0 \tag{1.10}$$

◀

▶ **Theorem 1.7.** ([34], pp. 291, Theorem 3.2)
The homogeneous boundary conditions (1.10) can be written as:

$$\sum_{i=1}^{n} m_{ki} u^{(i-1)}(a) = 0, \qquad \sum_{i=1}^{n} n_{ki} u^{(i-1)}(b) = 0, \qquad k = 1, 2 \tag{1.11}$$

or

$$\mathbf{m}_i^T \mathbf{u}(a) = \mathbf{n}_i^T \mathbf{u}(b) = 0 \tag{1.12}$$

where $\mathbf{u}^T = [u, u', u'', u''', \ldots, u^{(n-1)}]$ and $\mathbf{m}_i^T = [m_{i1}, m_{i2}, \ldots, m_{in}]$. The boundary conditions $Ux = 0$ is adjoint to itself if and only if

$$\mathbf{M}\mathbf{B}^{-1}(a)\mathbf{M}^* = \mathbf{N}\mathbf{B}^{-1}(b)\mathbf{N}^* \tag{1.13}$$

Here, $\mathbf{M} = [\mathbf{m}_1^T, \ldots, \mathbf{m}_n^T]$ and $\mathbf{M}^*$ is the complex conjugate of $\mathbf{M}$. ◀

▶ **Theorem 1.8.** ([34], pp. 189, Theorem 2.1).
Let Π given by (1.4) be a self-adjoint eigenvalue problem. Then the eigenvalues are real and constitute an at most enumerable set with no finite cluster point. ◀

Proof. Let $\lambda = \lambda_0$ be an eigenvalue with χ and eigenfunction of Π given by (1.4). Then because $\mathcal{L}\chi = \lambda_0\chi$, the relation (1.9) gives $(\lambda_0 - \overline{\lambda}_0)\langle\chi, \chi\rangle = 0$. Because $\langle\chi, \chi\rangle > 0$, it follows that $\lambda_0 = \overline{\lambda}_0$ and thus an eigenvalue must be real. ■

▶ **Definition 1.9. Inverse Eigenvalue Problems**.
The inverse eigenvalue consisting of the differential equation

$$\mathcal{L}_n u = a_0 \frac{d^n u}{dx^n} + a_1 \frac{d^{n-1} u}{dx^{n-1}} + \ldots + a_n u = \lambda u$$

together with suitable boundary conditions, is determining the unknown functions $a_0, a_1, \cdots, a_n$, given spectral information $\{\lambda_k\}$. ◀

▶ **Definition 1.10. Kronecker delta**.

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{1.14}$$

◀

▶ **Definition 1.11. Entire function** [21].
An entire function, also called an integral function, is a complex-valued function that is holomorphic at all finite points over the whole complex plane. Every entire function $f(z)$ can be represented as a power series $f(z) = \sum_{n=0}^{\infty} a_n z^n$ that converges everywhere in the complex plane, hence uniformly on compact sets. Entire functions of finite order have Hadamard's canonical representation:

$$f(z) = z^m e^{P(z)} \prod_{n=1}^{\infty} \left(1 - \frac{z}{z_n}\right) \exp\left(\frac{z}{z_n} + \cdots + \frac{1}{p}\left(\frac{z}{z_n}\right)^p\right) \tag{1.15}$$

where z_k are those roots of f that are not zero ($z_k \neq 0$), P a polynomial (whose degree we shall call q). The order (at infinity) of an entire function $f(z)$ is defined using the limit superior as:

$$\rho = \limsup_{r \to \infty} \frac{\ln(\ln \|f\|_{\infty, B_r})}{\ln r} \tag{1.16}$$

where B_r is the disk of radius r and $\|f\|_{\infty, B_r}$ denotes the supremum norm of $f(z)$ on B_r. ◀

Interest reader is referred to [86] and [87] for a comprehensive discussion on linear differential operators.

2 Uniqueness of Inverse Eigenvalue Problems

The main reference for this chapter is [18].

2.1 The Main Theorem

▶ **Theorem 2.1. Uniqueness of Inverse Eigenvalue Problem of order** $2n$. Given the self-adjoint eigenvalue problem for a differential equation of order $2n$:

$$u^{(2n)} - (p_1 u^{(n-1)})^{(n-1)} + \ldots + (-1)^n p_n u = \lambda u, \tag{2.1}$$

n+1 spectra are needed to determine the functions $p_1(x), \ldots, p_n(x)$ uniquely. ◀

▶ **Proposition 2.2.** $\mathcal{L}[y] = [a_0(x)y^{(n)}]^{(n)} + \ldots + (a_{n-1}(x)y')' + a_n(x)y$ is a self-adjoint operator. ◀

Proof.

$$\begin{aligned}
\mathcal{L}[y] =& [a_0(x)y^{(n)}]^{(n)} + \ldots + (a_{n-1}(x)y')' + a_n(x)y \\
=& \sum_{k=0}^{n} {}^{n}C_k a_0^{(n-k)} y^{(n+k)} + \sum_{k=0}^{n-1} {}^{n-1}C_k a_1^{(n-1-k)} y^{(n-1+k)} + \sum_{k=0}^{n-2} {}^{n-2}C_k a_2^{(n-2-k)} y^{(n-2+k)} + \ldots \\
&+ \sum_{k=0}^{3} {}^{3}C_k a_{n-3}^{(3-k)} y^{(3+k)} + \sum_{k=0}^{2} {}^{2}C_k a_{n-2}^{(2-k)} y^{(2+k)} + \sum_{k=0}^{1} {}^{1}C_k a_{n-1}^{(1-k)} y^{(1+k)} + a_n y \\
=& y^{(2n)} a_0 + y^{(2n-1)}[n a_0' + a_1] + y^{(2n-2)}\left[\frac{n(n-1)}{2} a_0'' + (n-1)a_1' + a_2\right] + \ldots \\
&+ y''[a_{n-1} + 2a_{n-2}''] + y' a_{n-1}' + a_n y
\end{aligned}$$

$$\begin{aligned}
\mathcal{L}^*[y] =& (-1)^{2n} D^{2n}[a_0 y] + (-1)^{2n-1} D^{2n-1}[(n a_0' + a_1)y] + \ldots \\
&+ (-1)^2 D^2[(2a_{n-2}'' + a_{n-1})y] + (-1)^1 D[a_{n-1}' y] + a_n y \\
=& \sum_{k=0}^{2n} {}^{2n}C_k a_0^{(k)} y^{(2n-k)} - \sum_{k=0}^{2n-1} {}^{2n-1}C_k (n a_0' + a_1)^{(k)} y^{(2n-1-k)} + \ldots
\end{aligned}$$

$$
\begin{aligned}
&+\sum_{k=0}^{2} {}^{2}C_k (a_{n-1} + 2a''_{n-2})^{(k)} y^{(2-k)} - \sum_{k=0}^{1} {}^{1}C_k (a'_{n-1})^{(k)} y^{(1-k)} + a_n y \\
=&y[a_n - a''_{n-1} + (a_{n-1} + 2a''_{n-2})'' - \ldots - (na'_0 + a_1)^{(2n-1)} + a_0^{(2n)}] + \ldots - [na'_0 + a_1] y^{(2n-1)} \\
&+ a_0 y^{(2n)} \\
=&\mathcal{L}[y]
\end{aligned}
$$

So $\mathcal{L}[y] = [a_0(x)y^{(n)}]^{(n)} + \ldots + (a_{n-1}(x)y')' + a_n(x)y$ is a self-adjoint operator. ■

2.2 Fourth order Sturm Liouville Problem (FSLP)

Consider the fourth order, self-adjoint differential operator

$$\mathcal{L}u \equiv u^{(4)} - (pu')' + qu = \lambda u, \quad x \in (0,1) \tag{2.2}$$

where $p(x)$ and $q(x)$ are real functions of x, $p(x)$ being differentiable. Consider the eigenvalue problem

$$\mathcal{L}u = \lambda u \tag{2.3}$$

together with the homogeneous boundary conditions:

$$\sum_{i=1}^{4} m_{ki} u^{(i-1)}(0) = 0, \qquad \sum_{i=1}^{4} n_{ki} u^{(i-1)}(1) = 0, \qquad k = 1, 2$$

or

$$\mathbf{m}_i^T \mathbf{u}(0) = \mathbf{n}_i^T \mathbf{u}(1) = 0 \tag{2.4}$$

where $\mathbf{u}^T = [u, u', u'', u''']$ and $\mathbf{m}_i^T = [m_{i1}, m_{i2}, m_{i3}, m_{i4}]$. From Theorem 1.7 for self-adjointness claim

$$\mathbf{m}_1^T \mathbf{B}^{-1}(0) \mathbf{m}_2 = 0, \quad \mathbf{n}_1^T \mathbf{B}^{-1}(1) \mathbf{n}_2 = 0. \tag{2.5}$$

where,

$$\mathbf{B}(x) = \begin{pmatrix} 0 & -p(x) & 0 & 1 \\ p(x) & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{pmatrix} \tag{2.6}$$

Proof.

$$\begin{aligned} \mathcal{L}u &\equiv u^{(4)} - (pu')' + qu \\ \langle \mathcal{L}u, v\rangle &= \int_0^1 [u^{(4)} - (pu')' + qu]v^* \, dx \\ &= [\underbrace{(+)}_{B_{14}} u'''v^* + \underbrace{(-)}_{B_{23}} u''(v^*)' + \underbrace{(+)}_{B_{32}} u'(v^*)'' + \underbrace{(-)}_{B_{41}} u(v^*)''' + \underbrace{(-p)}_{B_{12}} u'v^* + \underbrace{(p)}_{B_{21}} u(v^*)']_0^1 \\ &\quad + \int_0^1 u[(v^*)^{(4)} - (p(v^*)')' + qv^*] \, dx \end{aligned}$$

And all the other $B_{ij} = 0$, hence the Equation (2.6). ∎

Also, since we have $\mathbf{B}(x)$ is regular (it is an upper-triangular matrix), it has the inverse

$$\mathbf{B}^{-1}(x) = \begin{pmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & -p(x) \\ 1 & 0 & p(x) & 0 \end{pmatrix} \tag{2.7}$$

Let $\mathbf{m}_3$ be a third boundary operator which is linearly independent from $\mathbf{m}_1$ and $\mathbf{m}_2$.
Let $S(\mathbf{m}_i, \mathbf{m}_j), i \neq j$ be the spectrum associated with the boundary operators $\mathbf{m}_i$, $\mathbf{m}_j$ at $x = 0$ for the operator $\mathcal{L}$ and the boundary conditions $\mathbf{n}_k^T \mathbf{u}(1) = 0, k = 1, 2$.

2.2.1 Wronskian of FSLP

Let $\mathbf{m}_4$, $\mathbf{n}_3$ and $\mathbf{n}_4$ are any vectors such that $\mathbf{m}_i$ and $\mathbf{n}_i$, $(i = 1, 2, 3, 4)$ form two bases. Introduce four fundamental solutions of (2.2), $\phi(x, \lambda)$, $\psi(x, \lambda)$, $\eta(x, \lambda)$,

and $\zeta(x,\lambda)$ which are defined as:

$$\mathbf{m}_i^T \boldsymbol{\phi}(0,\lambda) = \delta_{i3} \tag{2.8a}$$
$$\mathbf{m}_i^T \boldsymbol{\psi}(0,\lambda) = \delta_{i4} \tag{2.8b}$$
$$\mathbf{n}_i^T \boldsymbol{\eta}(1,\lambda) = \delta_{i3} \tag{2.8c}$$
$$\mathbf{n}_i^T \boldsymbol{\zeta}(1,\lambda) = \delta_{i4} \tag{2.8d}$$

where δ_{ij} is the Kronecker delta.

▶ **Proposition 2.3.** [101], Lemma 5.4.1

$$|f(x,\lambda)| \le 2e^{|\lambda^{1/4}|x}, \qquad f \in \{\phi,\psi,\eta,\zeta\}$$

◀

From the above proposition, the functions ϕ, ψ, η, and ζ are entire functions of λ of order $\frac{1}{4}$.

For large values of $|\lambda|$, the asymptotic behavior of these functions is obtained by setting $p(x)$ and $q(x)$ equal to zero in (2.2). In other words,

$$\left.\begin{aligned} \phi(x,\lambda) &\sim \phi^{(0)}(x,\lambda) \\ \psi(x,\lambda) &\sim \psi^{(0)}(x,\lambda) \\ \eta(x,\lambda) &\sim \eta^{(0)}(x,\lambda) \\ \zeta(x,\lambda) &\sim \zeta^{(0)}(x,\lambda) \end{aligned}\right\} \quad \text{as } |\lambda| \to \infty \tag{2.9}$$

where $\phi^{(0)}$, $\psi^{(0)}$, $\eta^{(0)}$ and $\zeta^{(0)}$ are the solutions of

$$u^{(4)} = \lambda u$$

subject to the boundary conditions (2.8).

The fundamental matrix of (2.2) is

$$\mathbf{W}(\xi,\lambda) = [\boldsymbol{\eta}(\xi,\lambda), \boldsymbol{\zeta}(\xi,\lambda), \boldsymbol{\phi}(\xi,\lambda), \boldsymbol{\psi}(\xi,\lambda)] \tag{2.10}$$

The Wronskian of the fundamental solutions ϕ, ψ, η, ζ is $W(\lambda)$.

▶ **Theorem 2.4.** $W(\lambda)$ is independent of the variable ξ. ◀

Proof.

$$W(\xi,\lambda) = \begin{vmatrix} \eta & \zeta & \phi & \psi \\ \eta' & \zeta' & \phi' & \psi' \\ \eta'' & \zeta'' & \phi'' & \psi'' \\ \eta''' & \zeta''' & \phi''' & \psi''' \end{vmatrix}$$

$$\implies \frac{dW}{d\xi} = \begin{vmatrix} \eta & \zeta & \phi & \psi \\ \eta' & \zeta' & \phi' & \psi' \\ \eta'' & \zeta'' & \phi'' & \psi'' \\ \eta'''' & \zeta'''' & \phi'''' & \psi'''' \end{vmatrix}$$

From equation (2.2), we see that $u^{(4)} = (\lambda - q)u + p'u' + pu''$, hence last row is a linear combination of other three rows. So the derivative is zero, which implies that W is independent of ξ. ■

▶ **Lemma 2.5.** Given the boundary conditions $\mathbf{m}_i$, $\mathbf{n}_i$, $(i = 1, 2)$ and the spectrum $S(\mathbf{m}_1, \mathbf{m}_2)$, $W(\lambda)$ is completely determined. ◀

Proof. As $W(\lambda)$ is independent of the variable ξ, we can set ξ equal to 0 or 1 in (2.10) to evaluate $W(\lambda)$.
Assume without loss of generality

$$\det(\mathbf{M}) = \det(\mathbf{N}) = 1.$$

Then,

$$\det(\mathbf{W}) = \det(\mathbf{MW}) \tag{2.11a}$$
$$\det(\mathbf{W}) = \det(\mathbf{NW}). \tag{2.11b}$$

$$\mathbf{MW} = \begin{pmatrix} \mathbf{m}_1^T \\ \mathbf{m}_2^T \\ \mathbf{m}_3^T \\ \mathbf{m}_4^T \end{pmatrix} \begin{pmatrix} \boldsymbol{\eta} & \boldsymbol{\zeta} & \boldsymbol{\phi} & \boldsymbol{\psi} \end{pmatrix}$$

$$\det(\mathbf{MW}) = \begin{vmatrix} \mathbf{m}_1^T\boldsymbol{\eta} & \mathbf{m}_1^T\boldsymbol{\zeta} & \mathbf{m}_1^T\boldsymbol{\phi} & \mathbf{m}_1^T\boldsymbol{\psi} \\ \mathbf{m}_2^T\boldsymbol{\eta} & \mathbf{m}_2^T\boldsymbol{\zeta} & \mathbf{m}_2^T\boldsymbol{\phi} & \mathbf{m}_2^T\boldsymbol{\psi} \\ \mathbf{m}_3^T\boldsymbol{\eta} & \mathbf{m}_3^T\boldsymbol{\zeta} & \mathbf{m}_3^T\boldsymbol{\phi} & \mathbf{m}_3^T\boldsymbol{\psi} \\ \mathbf{m}_4^T\boldsymbol{\eta} & \mathbf{m}_4^T\boldsymbol{\zeta} & \mathbf{m}_4^T\boldsymbol{\phi} & \mathbf{m}_4^T\boldsymbol{\psi} \end{vmatrix}$$

Using (2.8)

$$\xi = 0 \implies \det(\mathbf{MW}) = \begin{vmatrix} \mathbf{m}_1^T\boldsymbol{\eta}(0,\lambda) & \mathbf{m}_1^T\boldsymbol{\zeta}(0,\lambda) & 0 & 0 \\ \mathbf{m}_2^T\boldsymbol{\eta}(0,\lambda) & \mathbf{m}_2^T\boldsymbol{\zeta}(0,\lambda) & 0 & 0 \\ \mathbf{m}_3^T\boldsymbol{\eta}(0,\lambda) & \mathbf{m}_3^T\boldsymbol{\zeta}(0,\lambda) & 1 & 0 \\ \mathbf{m}_4^T\boldsymbol{\eta}(0,\lambda) & \mathbf{m}_4^T\boldsymbol{\zeta}(0,\lambda) & 0 & 1 \end{vmatrix}$$

$$= \begin{vmatrix} \mathbf{m}_1^T\boldsymbol{\eta}(0,\lambda) & \mathbf{m}_1^T\boldsymbol{\zeta}(0,\lambda) \\ \mathbf{m}_2^T\boldsymbol{\eta}(0,\lambda) & \mathbf{m}_2^T\boldsymbol{\zeta}(0,\lambda) \end{vmatrix}$$

$$W(\lambda) = \{\mathbf{m}_1^T\boldsymbol{\eta}(0,\lambda)\}\{\mathbf{m}_2^T\boldsymbol{\zeta}(0,\lambda)\} - \{\mathbf{m}_2^T\boldsymbol{\eta}(0,\lambda)\}\{\mathbf{m}_1^T\boldsymbol{\zeta}(0,\lambda)\} \quad (2.12)$$

Similarly,

$$\mathbf{NW} = \begin{pmatrix} \mathbf{n}_1^T \\ \mathbf{n}_2^T \\ \mathbf{n}_3^T \\ \mathbf{n}_4^T \end{pmatrix} \begin{pmatrix} \boldsymbol{\eta} & \boldsymbol{\zeta} & \boldsymbol{\phi} & \boldsymbol{\psi} \end{pmatrix}$$

$$\det(\mathbf{NW}) = \begin{vmatrix} \mathbf{n}_1^T\boldsymbol{\eta} & \mathbf{n}_1^T\boldsymbol{\zeta} & \mathbf{n}_1^T\boldsymbol{\phi} & \mathbf{n}_1^T\boldsymbol{\psi} \\ \mathbf{n}_2^T\boldsymbol{\eta} & \mathbf{n}_2^T\boldsymbol{\zeta} & \mathbf{n}_2^T\boldsymbol{\phi} & \mathbf{n}_2^T\boldsymbol{\psi} \\ \mathbf{n}_3^T\boldsymbol{\eta} & \mathbf{n}_3^T\boldsymbol{\zeta} & \mathbf{n}_3^T\boldsymbol{\phi} & \mathbf{n}_3^T\boldsymbol{\psi} \\ \mathbf{n}_4^T\boldsymbol{\eta} & \mathbf{n}_4^T\boldsymbol{\zeta} & \mathbf{n}_4^T\boldsymbol{\phi} & \mathbf{n}_4^T\boldsymbol{\psi} \end{vmatrix}$$

Using (2.8)

$$\xi = 1 \implies \det(\mathbf{NW}) = \begin{vmatrix} 0 & 0 & \mathbf{n}_1^T\boldsymbol{\phi}(1,\lambda) & \mathbf{n}_1^T\boldsymbol{\psi}(1,\lambda) \\ 0 & 0 & \mathbf{n}_2^T\boldsymbol{\phi}(1,\lambda) & \mathbf{n}_2^T\boldsymbol{\psi}(1,\lambda) \\ 1 & 0 & \mathbf{n}_3^T\boldsymbol{\phi}(1,\lambda) & \mathbf{n}_3^T\boldsymbol{\psi}(1,\lambda) \\ 0 & 1 & \mathbf{n}_4^T\boldsymbol{\phi}(1,\lambda) & \mathbf{n}_4^T\boldsymbol{\psi}(1,\lambda) \end{vmatrix}$$

$$= \begin{vmatrix} \mathbf{n}_1^T\boldsymbol{\phi}(1,\lambda) & \mathbf{n}_1^T\boldsymbol{\psi}(1,\lambda) \\ \mathbf{n}_2^T\boldsymbol{\phi}(1,\lambda) & \mathbf{n}_2^T\boldsymbol{\psi}(1,\lambda) \end{vmatrix}$$

$$W(\lambda) = \{\mathbf{n}_1^T\boldsymbol{\phi}(1,\lambda)\}\{\mathbf{n}_2^T\boldsymbol{\psi}(1,\lambda)\} - \{\mathbf{n}_2^T\boldsymbol{\phi}(1,\lambda)\}\{\mathbf{n}_1^T\boldsymbol{\psi}(1,\lambda)\} \quad (2.13)$$

Since (2.2) is self-adjoint, according to Theorem 1.8, the zeros $\{\lambda_n\}_1^\infty$ of $W(\lambda)$ are real and simple, and they coincide with the eigenvalues of (2.2), (2.4), i.e.

$$S(\mathbf{m}_1, \mathbf{m}_2) = \{\lambda_n\}_1^\infty$$

Also $W(\lambda)$ is an entire function of order $\frac{1}{2}$, except for a constant multiplicative factor, it

is completely determined by its distribution of zeros, namely

$$W(\lambda) = k \prod_{i=1}^{\infty} \left(1 - \frac{\lambda}{\lambda_i}\right)$$

The constant k can be obtained by examining the asymptotic behavior of $W(\lambda)$ for $|\lambda| \to \infty$.

Also from (2.12) (or (2.13)) and (2.9),

$$\begin{aligned}\lim_{|\lambda|\to\infty} W(\lambda) &= \lim_{|\lambda|\to\infty} \left\{\mathbf{m}_1^T \boldsymbol{\eta}(0,\lambda)\right\}\left\{\mathbf{m}_2^T \boldsymbol{\zeta}(0,\lambda)\right\} - \left\{\mathbf{m}_2^T \boldsymbol{\eta}(0,\lambda)\right\}\left\{\mathbf{m}_1^T \boldsymbol{\zeta}(0,\lambda)\right\} \\ &= \left\{\mathbf{m}_1^T \boldsymbol{\eta}^{(0)}(0,\lambda)\right\}\left\{\mathbf{m}_2^T \boldsymbol{\zeta}^{(0)}(0,\lambda)\right\} - \left\{\mathbf{m}_2^T \boldsymbol{\eta}^{(0)}(0,\lambda)\right\}\left\{\mathbf{m}_1^T \boldsymbol{\zeta}^{(0)}(0,\lambda)\right\}\end{aligned}$$

Since $\boldsymbol{\eta}^{(0)}(0,\lambda)$ and $\boldsymbol{\zeta}^{(0)}(0,\lambda)$, are independent of $p(x)$ and $q(x)$, so does k. Thus, given the boundary conditions $\mathbf{m}_i$, $\mathbf{n}_i$, $(i = 1, 2)$ and the spectrum $S(\mathbf{m}_1, \mathbf{m}_2)$, $W(\lambda)$ is completely determined. ■

2.2.2 Green's Function of FSLP

Consider the Green's function $g(x, \eta; \lambda)$ which is defined as:

$$\begin{aligned}\mathcal{L}g - \lambda g &= \delta(x - \xi), \\ \mathbf{m}_i^T \mathbf{g}(0, \xi; \lambda) &= \mathbf{n}_i^T \mathbf{g}(1, \xi; \lambda) = 0,\end{aligned}$$

where δ is the Dirac delta-function.

$$g(x, \xi; \lambda) = \frac{1}{W(\lambda)} \begin{cases} a(\xi,\lambda)\eta(x,\lambda) + b(\xi,\lambda)\zeta(x,\lambda), & x > \xi \\ -c(\xi,\lambda)\phi(x,\lambda) - d(\xi,\lambda)\psi(x,\lambda), & x < \xi. \end{cases} \tag{2.14}$$

The functions a, b, c and d are found by solving the matrix equation

$$\mathbf{W}(\zeta, \lambda) \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ W(\lambda) \end{bmatrix} \tag{2.15}$$

► **Lemma 2.6.**

$$\mathbf{a}(0,\lambda) = \left\{\mathbf{m}_1^T \boldsymbol{\zeta}(0,\lambda)\right\}\mathbf{B}^{-1}(0)\mathbf{m}_2 - \left\{\mathbf{m}_2^T \boldsymbol{\zeta}(0,\lambda)\right\}\mathbf{B}^{-1}(0)\mathbf{m}_1. \tag{2.16}$$

◀

Proof. From (2.15), using Cramer's rule,

$$a(x,\lambda) = \frac{\Delta_1}{\Delta},$$

where

$$\Delta = |\mathbf{W}(\lambda)| = \begin{vmatrix} \eta & \zeta & \phi & \psi \\ \eta' & \zeta' & \phi' & \psi' \\ \eta'' & \zeta'' & \phi'' & \psi'' \\ \eta''' & \zeta''' & \phi''' & \psi''' \end{vmatrix} = W(\lambda)$$

and

$$\Delta_1 = \begin{vmatrix} 0 & \zeta & \phi & \psi \\ 0 & \zeta' & \phi' & \psi' \\ 0 & \zeta'' & \phi'' & \psi'' \\ W(\lambda) & \zeta''' & \phi''' & \psi''' \end{vmatrix} = -W(\lambda) \begin{vmatrix} \zeta & \phi & \psi \\ \zeta' & \phi' & \psi' \\ \zeta'' & \phi'' & \psi'' \end{vmatrix}$$

We can obtain an explicit formula for $a(x,\lambda)$, as:

$$a(x,\lambda) = -\begin{vmatrix} \zeta(x,\lambda) & \phi(x,\lambda) & \psi(x,\lambda) \\ \zeta'(x,\lambda) & \phi'(x,\lambda) & \psi'(x,\lambda) \\ \zeta''(x,\lambda) & \phi''(x,\lambda) & \psi''(x,\lambda) \end{vmatrix}. \tag{2.17}$$

Differentiating $a(x,\lambda)$ once,

$$a'(x,\lambda) = -\begin{vmatrix} \zeta & \phi & \psi \\ \zeta' & \phi' & \psi' \\ \zeta''' & \phi''' & \psi''' \end{vmatrix}. \tag{2.18}$$

Differentiating $a'(x,\lambda)$ again,

$$a''(x,\lambda) = -\begin{vmatrix} \zeta & \phi & \psi \\ \zeta'' & \phi'' & \psi'' \\ \zeta''' & \phi''' & \psi''' \end{vmatrix} - \begin{vmatrix} \zeta & \phi & \psi \\ \zeta' & \phi' & \psi' \\ \zeta'''' & \phi'''' & \psi'''' \end{vmatrix} = -\begin{vmatrix} \zeta & \phi & \psi \\ \zeta'' & \phi'' & \psi'' \\ \zeta''' & \phi''' & \psi''' \end{vmatrix} - \begin{vmatrix} \zeta & \phi & \psi \\ \zeta' & \phi' & \psi' \\ p\zeta'' & p\phi'' & p\psi'' \end{vmatrix}. \tag{2.19}$$

Differentiating $a''(x,\lambda)$ again,

$$a'''(x,\lambda) = -\begin{vmatrix} \zeta' & \phi' & \psi' \\ \zeta'' & \phi'' & \psi'' \\ \zeta''' & \phi''' & \psi''' \end{vmatrix} - \begin{vmatrix} \zeta & \phi & \psi \\ \zeta'' & \phi'' & \psi'' \\ \zeta'''' & \phi'''' & \psi'''' \end{vmatrix} - \underbrace{\begin{vmatrix} \zeta & \phi & \psi \\ \zeta'' & \phi'' & \psi'' \\ p\zeta'' & p\phi'' & p\psi'' \end{vmatrix}}_{0}$$

$$- \begin{vmatrix} \zeta & \phi & \psi \\ \zeta' & \phi' & \psi' \\ p\zeta''' + p'\zeta'' & p\phi''' + p'\phi'' & p\psi''' + p'\psi'' \end{vmatrix}$$

(we have used in last row of last determinant $u^{(5)} = p'u'' + pu''' - qu'$)

$$= -\begin{vmatrix} \zeta' & \phi' & \psi' \\ \zeta'' & \phi'' & \psi'' \\ \zeta''' & \phi''' & \psi''' \end{vmatrix} - \begin{vmatrix} \zeta & \phi & \psi \\ \zeta'' & \phi'' & \psi'' \\ p'\zeta' & p'\phi' & p'\psi' \end{vmatrix} - \begin{vmatrix} \zeta & \phi & \psi \\ \zeta' & \phi' & \psi' \\ p'\zeta'' & p'\phi'' & p'\psi'' \end{vmatrix} - \begin{vmatrix} \zeta & \phi & \psi \\ \zeta' & \phi' & \psi' \\ p\zeta''' & p\phi''' & p\psi''' \end{vmatrix}$$

$$= -\begin{vmatrix} \zeta' & \phi' & \psi' \\ \zeta'' & \phi'' & \psi'' \\ \zeta''' & \phi''' & \psi''' \end{vmatrix} \underbrace{+p'\begin{vmatrix} \zeta & \phi & \psi \\ \zeta' & \phi' & \psi' \\ \zeta'' & \phi'' & \psi'' \end{vmatrix} - p'\begin{vmatrix} \zeta & \phi & \psi \\ \zeta' & \phi' & \psi' \\ \zeta'' & \phi'' & \psi'' \end{vmatrix}}_{0} - p\begin{vmatrix} \zeta & \phi & \psi \\ \zeta' & \phi' & \psi' \\ \zeta''' & \phi''' & \psi''' \end{vmatrix}$$

$$= -\begin{vmatrix} \zeta' & \phi' & \psi' \\ \zeta'' & \phi'' & \psi'' \\ \zeta''' & \phi''' & \psi''' \end{vmatrix} - p\begin{vmatrix} \zeta & \phi & \psi \\ \zeta' & \phi' & \psi' \\ \zeta''' & \phi''' & \psi''' \end{vmatrix}. \tag{2.20}$$

Let

$$\alpha_i(x,\lambda) = \det([\boldsymbol{\delta}_1, \boldsymbol{\zeta}(x,\lambda), \boldsymbol{\phi}(x,\lambda), \boldsymbol{\psi}(x,\lambda)]) \tag{2.21}$$

and $\boldsymbol{\delta}_i = (\delta_{ij})$. That is,

$$\alpha_1(x,\lambda) = \det([\boldsymbol{\delta}_1, \boldsymbol{\zeta}(x,\lambda), \boldsymbol{\phi}(x,\lambda), \boldsymbol{\psi}(x,\lambda)]) = \begin{vmatrix} 1 & \zeta & \phi & \psi \\ 0 & \zeta' & \phi' & \psi' \\ 0 & \zeta'' & \phi'' & \psi'' \\ 0 & \zeta''' & \phi''' & \psi''' \end{vmatrix} = \begin{vmatrix} \zeta' & \phi' & \psi' \\ \zeta'' & \phi'' & \psi'' \\ \zeta''' & \phi''' & \psi''' \end{vmatrix}$$

$$\alpha_2(x,\lambda) = \det([\boldsymbol{\delta}_2, \boldsymbol{\zeta}(x,\lambda), \boldsymbol{\phi}(x,\lambda), \boldsymbol{\psi}(x,\lambda)]) = \begin{vmatrix} 0 & \zeta & \phi & \psi \\ 1 & \zeta' & \phi' & \psi' \\ 0 & \zeta'' & \phi'' & \psi'' \\ 0 & \zeta''' & \phi''' & \psi''' \end{vmatrix} = -\begin{vmatrix} \zeta & \phi & \psi \\ \zeta'' & \phi'' & \psi'' \\ \zeta''' & \phi''' & \psi''' \end{vmatrix}$$

$$\alpha_3(x,\lambda) = \det([\boldsymbol{\delta}_3, \boldsymbol{\zeta}(x,\lambda), \boldsymbol{\phi}(x,\lambda), \boldsymbol{\psi}(x,\lambda)]) = \begin{vmatrix} 0 & \zeta & \phi & \psi \\ 0 & \zeta' & \phi' & \psi' \\ 1 & \zeta'' & \phi'' & \psi'' \\ 0 & \zeta''' & \phi''' & \psi''' \end{vmatrix} = \begin{vmatrix} \zeta & \phi & \psi \\ \zeta' & \phi' & \psi' \\ \zeta''' & \phi''' & \psi''' \end{vmatrix}$$

$$\alpha_4(x,\lambda) = \det([\boldsymbol{\delta}_4, \boldsymbol{\zeta}(x,\lambda), \boldsymbol{\phi}(x,\lambda), \boldsymbol{\psi}(x,\lambda)]) = \begin{vmatrix} 0 & \zeta & \phi & \psi \\ 0 & \zeta' & \phi' & \psi' \\ 0 & \zeta'' & \phi'' & \psi'' \\ 1 & \zeta''' & \phi''' & \psi''' \end{vmatrix} = -\begin{vmatrix} \zeta & \phi & \psi \\ \zeta' & \phi' & \psi' \\ \zeta'' & \phi'' & \psi'' \end{vmatrix}$$

Hence equations (2.17)-(2.20) reduce to

$$a = \alpha_4, \quad a' = -\alpha_3, \quad a'' = \alpha_2 + p\alpha_4, \quad a''' = -\alpha_1 - p\alpha_3$$

so that,

$$\mathbf{a} = \begin{pmatrix} a \\ a' \\ a'' \\ a''' \end{pmatrix} = \begin{pmatrix} \alpha_4 \\ -\alpha_3 \\ \alpha_2 + p\alpha_4 \\ -\alpha_1 - p\alpha_3 \end{pmatrix} = -\begin{pmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & -p(x) \\ 1 & 0 & p(x) & 0 \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{pmatrix} \tag{2.22}$$

From (2.7), we recognize that

$$\mathbf{a}(x,\lambda) = -\mathbf{B}^{-1}(x)\boldsymbol{\alpha}(x,\lambda) \tag{2.23}$$

Since $\det(\mathbf{M})$ is equal to unity, we can rewrite (2.21) as

$$\alpha_i(x,\lambda) = \det(\mathbf{M} \cdot [\boldsymbol{\delta}_i, \boldsymbol{\zeta}, \boldsymbol{\phi}, \boldsymbol{\psi}]) \tag{2.24}$$

Consider,

$$\mathbf{M} \cdot [\boldsymbol{\delta}_i, \boldsymbol{\zeta}(0,\lambda), \boldsymbol{\phi}(0,\lambda), \boldsymbol{\psi}(0,\lambda)] = \begin{pmatrix} \mathbf{m}_1^T \\ \mathbf{m}_2^T \\ \mathbf{m}_3^T \\ \mathbf{m}_4^T \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta}_i & \boldsymbol{\zeta}_0 & \boldsymbol{\phi}_0 & \boldsymbol{\psi}_0 \end{pmatrix}$$

$$= \begin{pmatrix} \mathbf{m}_1^T\boldsymbol{\delta}_i & \mathbf{m}_1^T\boldsymbol{\zeta}_0 & \mathbf{m}_1^T\boldsymbol{\phi}_0 & \mathbf{m}_1^T\boldsymbol{\psi}_0 \\ \mathbf{m}_2^T\boldsymbol{\delta}_i & \mathbf{m}_2^T\boldsymbol{\zeta}_0 & \mathbf{m}_2^T\boldsymbol{\phi}_0 & \mathbf{m}_2^T\boldsymbol{\psi}_0 \\ \mathbf{m}_3^T\boldsymbol{\delta}_i & \mathbf{m}_3^T\boldsymbol{\zeta}_0 & \mathbf{m}_3^T\boldsymbol{\phi}_0 & \mathbf{m}_3^T\boldsymbol{\psi}_0 \\ \mathbf{m}_4^T\boldsymbol{\delta}_i & \mathbf{m}_4^T\boldsymbol{\zeta}_0 & \mathbf{m}_4^T\boldsymbol{\phi}_0 & \mathbf{m}_4^T\boldsymbol{\psi}_0 \end{pmatrix}$$

where $\boldsymbol{\zeta}_0$ denotes $\boldsymbol{\zeta}(0,\lambda)$, and so on. From (2.8), we have

$$\mathbf{M}\cdot[\boldsymbol{\delta}_i,\boldsymbol{\zeta}(0,\lambda),\boldsymbol{\phi}(0,\lambda),\boldsymbol{\psi}(0,\lambda)]=\begin{pmatrix}\mathbf{m}_1^T\boldsymbol{\delta}_i & \mathbf{m}_1^T\boldsymbol{\zeta}_0 & 0 & 0\\ \mathbf{m}_2^T\boldsymbol{\delta}_i & \mathbf{m}_2^T\boldsymbol{\zeta}_0 & 0 & 0\\ \mathbf{m}_3^T\boldsymbol{\delta}_i & \mathbf{m}_3^T\boldsymbol{\zeta}_0 & 1 & 0\\ \mathbf{m}_4^T\boldsymbol{\delta}_i & \mathbf{m}_4^T\boldsymbol{\zeta}_0 & 0 & 1\end{pmatrix}$$

$$\det(\mathbf{M}\cdot[\boldsymbol{\delta}_i,\boldsymbol{\zeta}_0,\boldsymbol{\phi}_0,\boldsymbol{\psi}_0])=\begin{vmatrix}\mathbf{m}_1^T\boldsymbol{\delta}_i & \mathbf{m}_1^T\boldsymbol{\zeta}_0\\ \mathbf{m}_2^T\boldsymbol{\delta}_i & \mathbf{m}_2^T\boldsymbol{\zeta}_0\end{vmatrix}=\{\mathbf{m}_1^T\boldsymbol{\delta}_i\}\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\}-\{\mathbf{m}_2^T\boldsymbol{\delta}_i\}\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\}$$

Also, $\mathbf{m}_1^T\boldsymbol{\delta}_i=m_{1i}$ and $\mathbf{m}_2^T\boldsymbol{\delta}_i=m_{2i}$. For $x=0$, equation (2.24) can be written as,

$$\alpha_i(0,\lambda)=\det(\mathbf{M}\cdot[\boldsymbol{\delta}_i,\boldsymbol{\zeta}(0,\lambda),\boldsymbol{\phi}(0,\lambda),\boldsymbol{\psi}(0,\lambda)])$$

So that,

$$\left.\begin{aligned}\alpha_1(0,\lambda) &= m_{11}\{\mathbf{m}_2^T\boldsymbol{\zeta}(0,\lambda)\}-m_{21}\{\mathbf{m}_1^T\boldsymbol{\zeta}(0,\lambda)\}\\ \alpha_2(0,\lambda) &= m_{12}\{\mathbf{m}_2^T\boldsymbol{\zeta}(0,\lambda)\}-m_{22}\{\mathbf{m}_1^T\boldsymbol{\zeta}(0,\lambda)\}\\ \alpha_3(0,\lambda) &= m_{13}\{\mathbf{m}_2^T\boldsymbol{\zeta}(0,\lambda)\}-m_{23}\{\mathbf{m}_1^T\boldsymbol{\zeta}(0,\lambda)\}\\ \alpha_4(0,\lambda) &= m_{14}\{\mathbf{m}_2^T\boldsymbol{\zeta}(0,\lambda)\}-m_{24}\{\mathbf{m}_1^T\boldsymbol{\zeta}(0,\lambda)\}.\end{aligned}\right\}$$

or:

$$\alpha_i(0,\lambda)=m_{1i}\{\mathbf{m}_2^T\boldsymbol{\zeta}(0,\lambda)\}-m_{2i}\{\mathbf{m}_1^T\boldsymbol{\zeta}(0,\lambda)\}. \tag{2.25}$$

By substituting (2.25) in (2.23) we obtain

$$\begin{aligned}\mathbf{a}(0,\lambda) &= -\mathbf{B}^{-1}(0)\begin{pmatrix}m_{11}\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\}-m_{21}\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\}\\ m_{12}\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\}-m_{22}\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\}\\ m_{13}\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\}-m_{23}\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\}\\ m_{14}\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\}-m_{24}\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\}\end{pmatrix}\\ &= \{\mathbf{m}_1^T\boldsymbol{\zeta}(0,\lambda)\}\mathbf{B}^{-1}(0)\mathbf{m}_2-\{\mathbf{m}_2^T\boldsymbol{\zeta}(0,\lambda)\}\mathbf{B}^{-1}(0)\mathbf{m}_1.\end{aligned}$$

■

Similarly,

► **Lemma 2.7.**

$$\mathbf{b}(0,\lambda)=-\{\mathbf{m}_1^T\boldsymbol{\eta}(0,\lambda)\}\mathbf{B}^{-1}(0)\mathbf{m}_2+\{\mathbf{m}_2^T\boldsymbol{\eta}(0,\lambda)\}\mathbf{B}^{-1}(0)\mathbf{m}_1. \tag{2.26}$$

◄

Proof. From (2.15), using Cramer's rule,

$$b(x,\lambda) = \frac{\Delta_2}{\Delta},$$

where

$$\Delta_2 = \begin{vmatrix} \eta & 0 & \phi & \psi \\ \eta' & 0 & \phi' & \psi' \\ \eta'' & 0 & \phi'' & \psi'' \\ \eta''' & W(\lambda) & \phi''' & \psi''' \end{vmatrix} = W(\lambda) \begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ \eta'' & \phi'' & \psi'' \end{vmatrix}$$

We can obtain an explicit formula for $b(x,\lambda)$, as:

$$b(x,\lambda) = \begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ \eta'' & \phi'' & \psi'' \end{vmatrix}. \tag{2.27}$$

Differentiating $b(x,\lambda)$ once,

$$b'(x,\lambda) = \begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ \eta''' & \phi''' & \psi''' \end{vmatrix}. \tag{2.28}$$

Differentiating $b'(x,\lambda)$ again,

$$b''(x,\lambda) = \begin{vmatrix} \eta & \phi & \psi \\ \eta'' & \phi'' & \psi'' \\ \eta''' & \phi''' & \psi''' \end{vmatrix} + \begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ \eta'''' & \phi'''' & \psi'''' \end{vmatrix} = \begin{vmatrix} \eta & \phi & \psi \\ \eta'' & \phi'' & \psi'' \\ \eta''' & \phi''' & \psi''' \end{vmatrix} + \begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ p\eta'' & p\phi'' & p\psi'' \end{vmatrix}. \tag{2.29}$$

Differentiating $b''(x,\lambda)$ again,

$$b'''(x,\lambda) = \begin{vmatrix} \eta' & \phi' & \psi' \\ \eta'' & \phi'' & \psi'' \\ \eta''' & \phi''' & \psi''' \end{vmatrix} - \begin{vmatrix} \eta & \phi & \psi \\ \eta'' & \phi'' & \psi'' \\ \eta'''' & \phi'''' & \psi'''' \end{vmatrix} + \underbrace{\begin{vmatrix} \eta & \phi & \psi \\ \eta'' & \phi'' & \psi'' \\ p\eta'' & p\phi'' & p\psi'' \end{vmatrix}}_{0}$$

$$+ \begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ p\eta''' + p'\eta'' & p\phi''' + p'\phi'' & p\psi''' + p'\psi'' \end{vmatrix}$$

(we have used in last row of last determinant $u^{(5)} = p'u'' + pu''' - qu'$)

$$
\begin{aligned}
&= \begin{vmatrix} \eta' & \phi' & \psi' \\ \eta'' & \phi'' & \psi'' \\ \eta''' & \phi''' & \psi''' \end{vmatrix} + \begin{vmatrix} \eta & \phi & \psi \\ \eta'' & \phi'' & \psi'' \\ p'\eta' & p'\phi' & p'\psi' \end{vmatrix} + \begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ p'\eta'' & p'\phi'' & p'\psi'' \end{vmatrix} + \begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ p\eta''' & p\phi''' & p\psi''' \end{vmatrix} \\
&= \begin{vmatrix} \eta' & \phi' & \psi' \\ \eta'' & \phi'' & \psi'' \\ \eta''' & \phi''' & \psi''' \end{vmatrix} \underbrace{-p' \begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ \eta'' & \phi'' & \psi'' \end{vmatrix} + p' \begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ \eta'' & \phi'' & \psi'' \end{vmatrix}}_{0} + p \begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ \eta''' & \phi''' & \psi''' \end{vmatrix} \\
&= \begin{vmatrix} \eta' & \phi' & \psi' \\ \eta'' & \phi'' & \psi'' \\ \eta''' & \phi''' & \psi''' \end{vmatrix} + p \begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ \eta''' & \phi''' & \psi''' \end{vmatrix}. \qquad (2.30)
\end{aligned}
$$

Let

$$\beta_i(x,\lambda) = \det([\boldsymbol{\delta}_1, \boldsymbol{\eta}(x,\lambda), \boldsymbol{\phi}(x,\lambda), \boldsymbol{\psi}(x,\lambda)]) \qquad (2.31)$$

That is,

$$\beta_1(x,\lambda) = \det([\boldsymbol{\delta}_1, \boldsymbol{\eta}(x,\lambda), \boldsymbol{\phi}(x,\lambda), \boldsymbol{\psi}(x,\lambda)]) = \begin{vmatrix} 1 & \eta & \phi & \psi \\ 0 & \eta' & \phi' & \psi' \\ 0 & \eta'' & \phi'' & \psi'' \\ 0 & \eta''' & \phi''' & \psi''' \end{vmatrix} = \begin{vmatrix} \eta' & \phi' & \psi' \\ \eta'' & \phi'' & \psi'' \\ \eta''' & \phi''' & \psi''' \end{vmatrix}$$

$$\beta_2(x,\lambda) = \det([\boldsymbol{\delta}_2, \boldsymbol{\eta}(x,\lambda), \boldsymbol{\phi}(x,\lambda), \boldsymbol{\psi}(x,\lambda)]) = \begin{vmatrix} 0 & \eta & \phi & \psi \\ 1 & \eta' & \phi' & \psi' \\ 0 & \eta'' & \phi'' & \psi'' \\ 0 & \eta''' & \phi''' & \psi''' \end{vmatrix} = -\begin{vmatrix} \eta & \phi & \psi \\ \eta'' & \phi'' & \psi'' \\ \eta''' & \phi''' & \psi''' \end{vmatrix}$$

$$\beta_3(x,\lambda) = \det([\boldsymbol{\delta}_3, \boldsymbol{\eta}(x,\lambda), \boldsymbol{\phi}(x,\lambda), \boldsymbol{\psi}(x,\lambda)]) = \begin{vmatrix} 0 & \eta & \phi & \psi \\ 0 & \eta' & \phi' & \psi' \\ 1 & \eta'' & \phi'' & \psi'' \\ 0 & \eta''' & \phi''' & \psi''' \end{vmatrix} = \begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ \eta''' & \phi''' & \psi''' \end{vmatrix}$$

$$\beta_4(x,\lambda) = \det([\boldsymbol{\delta}_4, \boldsymbol{\eta}(x,\lambda), \boldsymbol{\phi}(x,\lambda), \boldsymbol{\psi}(x,\lambda)]) = \begin{vmatrix} 0 & \eta & \phi & \psi \\ 0 & \eta' & \phi' & \psi' \\ 0 & \eta'' & \phi'' & \psi'' \\ 1 & \eta''' & \phi''' & \psi''' \end{vmatrix} = -\begin{vmatrix} \eta & \phi & \psi \\ \eta' & \phi' & \psi' \\ \eta'' & \phi'' & \psi'' \end{vmatrix}$$

Hence equations (2.27)-(2.30) reduce to

$$b = -\beta_4, \quad b' = \beta_3, \quad b'' = -\beta_2 - p\beta_4, \quad b''' = \beta_1 + p\beta_3$$

so that,

$$\mathbf{b} = \begin{pmatrix} b \\ b' \\ b'' \\ b''' \end{pmatrix} = \begin{pmatrix} -\beta_4 \\ \beta_3 \\ -\beta_2 - p\beta_4 \\ \beta_1 + p\beta_3 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & -p(x) \\ 1 & 0 & p(x) & 0 \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{pmatrix} \tag{2.32}$$

From (2.7), we recognize that

$$\mathbf{b}(x,\lambda) = \mathbf{B}^{-1}(x)\boldsymbol{\beta}(x,\lambda) \tag{2.33}$$

Since $\det(\mathbf{M})$ is equal to unity, we can rewrite (2.31) as

$$\beta_i(x,\lambda) = \det(\mathbf{M} \cdot [\boldsymbol{\delta}_i, \boldsymbol{\eta}, \boldsymbol{\phi}, \boldsymbol{\psi}]) \tag{2.34}$$

Consider,

$$\mathbf{M} \cdot [\boldsymbol{\delta}_i, \boldsymbol{\eta}(0,\lambda), \boldsymbol{\phi}(0,\lambda), \boldsymbol{\psi}(0,\lambda)] = \begin{pmatrix} \mathbf{m}_1^T \\ \mathbf{m}_2^T \\ \mathbf{m}_3^T \\ \mathbf{m}_4^T \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta}_i & \boldsymbol{\eta}_0 & \boldsymbol{\phi}_0 & \boldsymbol{\psi}_0 \end{pmatrix}$$

$$= \begin{pmatrix} \mathbf{m}_1^T\boldsymbol{\delta}_i & \mathbf{m}_1^T\boldsymbol{\eta}_0 & \mathbf{m}_1^T\boldsymbol{\phi}_0 & \mathbf{m}_1^T\boldsymbol{\psi}_0 \\ \mathbf{m}_2^T\boldsymbol{\delta}_i & \mathbf{m}_2^T\boldsymbol{\eta}_0 & \mathbf{m}_2^T\boldsymbol{\phi}_0 & \mathbf{m}_2^T\boldsymbol{\psi}_0 \\ \mathbf{m}_3^T\boldsymbol{\delta}_i & \mathbf{m}_3^T\boldsymbol{\eta}_0 & \mathbf{m}_3^T\boldsymbol{\phi}_0 & \mathbf{m}_3^T\boldsymbol{\psi}_0 \\ \mathbf{m}_4^T\boldsymbol{\delta}_i & \mathbf{m}_4^T\boldsymbol{\eta}_0 & \mathbf{m}_4^T\boldsymbol{\phi}_0 & \mathbf{m}_4^T\boldsymbol{\psi}_0 \end{pmatrix}$$

where $\boldsymbol{\eta}_0$ denotes $\boldsymbol{\eta}(0,\lambda)$, and so on. From (2.8), we have

$$\mathbf{M} \cdot [\boldsymbol{\delta}_i, \boldsymbol{\eta}(0,\lambda), \boldsymbol{\phi}(0,\lambda), \boldsymbol{\psi}(0,\lambda)] = \begin{pmatrix} \mathbf{m}_1^T\boldsymbol{\delta}_i & \mathbf{m}_1^T\boldsymbol{\eta}_0 & 0 & 0 \\ \mathbf{m}_2^T\boldsymbol{\delta}_i & \mathbf{m}_2^T\boldsymbol{\eta}_0 & 0 & 0 \\ \mathbf{m}_3^T\boldsymbol{\delta}_i & \mathbf{m}_3^T\boldsymbol{\eta}_0 & 1 & 0 \\ \mathbf{m}_4^T\boldsymbol{\delta}_i & \mathbf{m}_4^T\boldsymbol{\eta}_0 & 0 & 1 \end{pmatrix}$$

$$\det(\mathbf{M} \cdot [\boldsymbol{\delta}_i, \boldsymbol{\eta}_0, \boldsymbol{\phi}_0, \boldsymbol{\psi}_0]) = \begin{vmatrix} \mathbf{m}_1^T\boldsymbol{\delta}_i & \mathbf{m}_1^T\boldsymbol{\eta}_0 \\ \mathbf{m}_2^T\boldsymbol{\delta}_i & \mathbf{m}_2^T\boldsymbol{\eta}_0 \end{vmatrix} = \{\mathbf{m}_1^T\boldsymbol{\delta}_i\}\{\mathbf{m}_2^T\boldsymbol{\eta}_0\} - \{\mathbf{m}_2^T\boldsymbol{\delta}_i\}\{\mathbf{m}_1^T\boldsymbol{\eta}_0\}$$

Also, $\mathbf{m}_1^T\boldsymbol{\delta}_i = m_{1i}$ and $\mathbf{m}_2^T\boldsymbol{\delta}_i = m_{2i}$. For $x = 0$, equation (2.34) can be written as,

$$\beta_i(0,\lambda) = \det(\mathbf{M} \cdot [\boldsymbol{\delta}_i, \boldsymbol{\eta}(0,\lambda), \boldsymbol{\phi}(0,\lambda), \boldsymbol{\psi}(0,\lambda)])$$

So that,

$$\left.\begin{array}{ll} \beta_1(0,\lambda) & = m_{11}\{\mathbf{m}_2^T\boldsymbol{\eta}(0,\lambda)\} - m_{21}\{\mathbf{m}_1^T\boldsymbol{\eta}(0,\lambda)\} \\ \beta_2(0,\lambda) & = m_{12}\{\mathbf{m}_2^T\boldsymbol{\eta}(0,\lambda)\} - m_{22}\{\mathbf{m}_1^T\boldsymbol{\eta}(0,\lambda)\} \\ \beta_3(0,\lambda) & = m_{13}\{\mathbf{m}_2^T\boldsymbol{\eta}(0,\lambda)\} - m_{23}\{\mathbf{m}_1^T\boldsymbol{\eta}(0,\lambda)\} \\ \beta_4(0,\lambda) & = m_{14}\{\mathbf{m}_2^T\boldsymbol{\eta}(0,\lambda)\} - m_{24}\{\mathbf{m}_1^T\boldsymbol{\eta}(0,\lambda)\}. \end{array}\right\}$$

or:

$$\beta_i(0,\lambda) = m_{1i}\{\mathbf{m}_2^T\boldsymbol{\eta}(0,\lambda)\} - m_{2i}\{\mathbf{m}_1^T\boldsymbol{\eta}(0,\lambda)\}. \tag{2.35}$$

By substituting (2.35) in (2.32) we obtain

$$\mathbf{b}(0,\lambda) = \mathbf{B}^{-1}(0)\begin{pmatrix} m_{11}\{\mathbf{m}_2^T\boldsymbol{\eta}_0\} - m_{21}\{\mathbf{m}_1^T\boldsymbol{\eta}_0\} \\ m_{12}\{\mathbf{m}_2^T\boldsymbol{\eta}_0\} - m_{22}\{\mathbf{m}_1^T\boldsymbol{\eta}_0\} \\ m_{13}\{\mathbf{m}_2^T\boldsymbol{\eta}_0\} - m_{23}\{\mathbf{m}_1^T\boldsymbol{\eta}_0\} \\ m_{14}\{\mathbf{m}_2^T\boldsymbol{\eta}_0\} - m_{24}\{\mathbf{m}_1^T\boldsymbol{\eta}_0\} \end{pmatrix}$$

$$\mathbf{b}(0,\lambda) = -\{\mathbf{m}_1^T\boldsymbol{\eta}(0,\lambda)\}\mathbf{B}^{-1}(0)\mathbf{m}_2 + \{\mathbf{m}_2^T\boldsymbol{\eta}(0,\lambda)\}\mathbf{B}^{-1}(0)\mathbf{m}_1.$$

■

From (2.5) on $\mathbf{m}_i$ and $\mathbf{n}_i$, $(i = 1, 2)$, the problem is self-adjoint. So $g(x, \xi; \lambda)$ is a symmetric function of its arguments x and ξ ([65], Lemma. 4.2, pp. 446). Interchanging x and ξ in (2.14),

$$g(\xi, x; \lambda) = \frac{1}{W(\lambda)}\begin{cases} a(x,\lambda)\eta(\xi,\lambda) + b(x,\lambda)\zeta(\xi,\lambda), & x < \xi \\ -c(x,\lambda)\phi(\xi,\lambda) - d(x,\lambda)\psi(\xi,\lambda), & x > \xi \end{cases} \tag{2.36}$$

we see that $a(x, \lambda)$ and $b(x, \lambda)$, must satisfy the same boundary conditions as $g(x, \xi; \lambda)$ at $x = 0$. It is because of (2.5) and

$$\mathbf{m}_i^T\mathbf{B}^{-1}(0)\mathbf{m}_i = 0.$$

Therefore,

$$\left.\begin{array}{l} a(x,\lambda) = a_1(\lambda)\phi(x,\lambda) + a_2(\lambda)\psi(x,\lambda), \\ b(x,\lambda) = b_1(\lambda)\phi(x,\lambda) + b_2(\lambda)\psi(x,\lambda). \end{array}\right\} \tag{2.37}$$

Differentiating this thrice w.r.t. x, we get a set of equations

$$a'(x,\lambda) = a_1(\lambda)\phi'(x,\lambda) + a_2(\lambda)\psi'(x,\lambda),$$

$$
\begin{aligned}
b'(x,\lambda) &= b_1(\lambda)\phi'(x,\lambda) + b_2(\lambda)\psi'(x,\lambda),\\
a''(x,\lambda) &= a_1(\lambda)\phi''(x,\lambda) + a_2(\lambda)\psi''(x,\lambda),\\
b''(x,\lambda) &= b_1(\lambda)\phi''(x,\lambda) + b_2(\lambda)\psi''(x,\lambda),\\
a'''(x,\lambda) &= a_1(\lambda)\phi'''(x,\lambda) + a_2(\lambda)\psi'''(x,\lambda),\\
b'''(x,\lambda) &= b_1(\lambda)\phi'''(x,\lambda) + b_2(\lambda)\psi'''(x,\lambda)
\end{aligned}
$$

or

$$\mathbf{a}(x,\lambda) = a_1(\lambda)\boldsymbol{\phi}(x,\lambda) + a_2(\lambda)\boldsymbol{\psi}(x,\lambda), \tag{2.38}$$

$$\mathbf{b}(x,\lambda) = b_1(\lambda)\boldsymbol{\phi}(x,\lambda) + b_2(\lambda)\boldsymbol{\psi}(x,\lambda) \tag{2.39}$$

Setting $x = 0$,

$$\mathbf{a}(0,\lambda) = a_1(\lambda)\boldsymbol{\phi}(0,\lambda) + a_2(\lambda)\boldsymbol{\psi}(0,\lambda), \tag{2.40}$$

$$\mathbf{b}(0,\lambda) = b_1(\lambda)\boldsymbol{\phi}(0,\lambda) + b_2(\lambda)\boldsymbol{\psi}(0,\lambda). \tag{2.41}$$

Using (2.16) and (2.26), we can evaluate a_1, a_2, b_1 and b_2 in (2.40) and (2.41).

Multiplying (2.40) by $\mathbf{m}_3^T$, we get

$$\mathbf{m}_3^T\mathbf{a}(0,\lambda) = a_1(\lambda)\underbrace{\left\{\mathbf{m}_3^T\boldsymbol{\phi}(0,\lambda)\right\}}_{1} + a_2(\lambda)\underbrace{\left\{\mathbf{m}_3^T\boldsymbol{\psi}(0,\lambda)\right\}}_{0}$$

using boundary conditions (2.8a) and (2.8b). Then we have,

$$a_1(\lambda) = \mathbf{m}_3^T\mathbf{a}(0,\lambda) \tag{2.42}$$

Similarly, multiplying (2.40) by $\mathbf{m}_4^T$, we get

$$\mathbf{m}_4^T\mathbf{a}(0,\lambda) = a_1(\lambda)\underbrace{\left\{\mathbf{m}_4^T\boldsymbol{\phi}(0,\lambda)\right\}}_{0} + a_2(\lambda)\underbrace{\left\{\mathbf{m}_4^T\boldsymbol{\psi}(0,\lambda)\right\}}_{1}$$

using boundary conditions (2.8a) and (2.8b). Then we have,

$$a_2(\lambda) = \mathbf{m}_4^T\mathbf{a}(0,\lambda) \tag{2.43}$$

Substituting (2.42) and (2.43) for a_1 and a_2 in (2.38)

$$\mathbf{a}(x,\lambda) = \left\{\mathbf{m}_3^T\mathbf{a}(0,\lambda)\right\}\boldsymbol{\phi}(x,\lambda) + \left\{\mathbf{m}_4^T\mathbf{a}(0,\lambda)\right\}\boldsymbol{\psi}(x,\lambda).$$

Using (2.16) we get

$$\begin{aligned}\mathbf{a}(x,\lambda) =&\left[\mathbf{m}_3^T\left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\}\mathbf{B}^{-1}(0)\mathbf{m}_2 - \mathbf{m}_3^T\left\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\right\}\mathbf{B}^{-1}(0)\mathbf{m}_1\right]\boldsymbol{\phi}(x,\lambda)\\ &+\left[\mathbf{m}_4^T\left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\}\mathbf{B}^{-1}(0)\mathbf{m}_2 - \mathbf{m}_4^T\left\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\right\}\mathbf{B}^{-1}(0)\mathbf{m}_1\right]\boldsymbol{\psi}(x,\lambda)\end{aligned}$$

or

$$\mathbf{a}(x,\lambda) = \left[\beta_{32}\left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\} - \beta_{31}\left\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\right\}\right]\boldsymbol{\phi}(x,\lambda) + \left[\beta_{42}\left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\} - \beta_{41}\left\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\right\}\right]\boldsymbol{\psi}(x,\lambda)$$

where

$$\beta_{ij} = \mathbf{m}_i^T\mathbf{B}^{-1}(0)\mathbf{m}_j.$$

Now we have

$$a(x,\lambda) = \left[\beta_{32}\left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\} - \beta_{31}\left\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\right\}\right]\phi(x,\lambda) + \left[\beta_{42}\left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\} - \beta_{41}\left\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\right\}\right]\psi(x,\lambda) \tag{2.44}$$

Similarly, multiplying (2.41) by $\mathbf{m}_3^T$, we get

$$\mathbf{m}_3^T\mathbf{b}(0,\lambda) = b_1(\lambda)\underbrace{\left\{\mathbf{m}_3^T\boldsymbol{\phi}(0,\lambda)\right\}}_{1} + b_2(\lambda)\underbrace{\left\{\mathbf{m}_3^T\boldsymbol{\psi}(0,\lambda)\right\}}_{0}$$

using boundary conditions (2.8a) and (2.8b). Then we have,

$$b_1(\lambda) = \mathbf{m}_3^T\mathbf{b}(0,\lambda) \tag{2.45}$$

Similarly, multiplying (2.41) by $\mathbf{m}_4^T$, we get

$$\mathbf{m}_4^T\mathbf{b}(0,\lambda) = b_1(\lambda)\underbrace{\left\{\mathbf{m}_4^T\boldsymbol{\phi}(0,\lambda)\right\}}_{0} + b_2(\lambda)\underbrace{\left\{\mathbf{m}_4^T\boldsymbol{\psi}(0,\lambda)\right\}}_{1}$$

using boundary conditions (2.8a) and (2.8b). Then we have,

$$b_2(\lambda) = \mathbf{m}_4^T\mathbf{b}(0,\lambda) \tag{2.46}$$

Substituting (2.45) and (2.46) for b_1 and b_2 in (2.39)

$$\mathbf{b}(x,\lambda) = \{\mathbf{m}_3^T\mathbf{b}(0,\lambda)\}\boldsymbol{\phi}(x,\lambda) + \{\mathbf{m}_4^T\mathbf{b}(0,\lambda)\}\boldsymbol{\psi}(x,\lambda)$$

Using (2.26) we get

$$\begin{aligned}\mathbf{b}(x,\lambda) =&\big[\mathbf{m}_3^T\{-\{\mathbf{m}_1^T\boldsymbol{\eta}_0\}\mathbf{B}^{-1}(0)\mathbf{m}_2 + \{\mathbf{m}_2^T\boldsymbol{\eta}_0\}\mathbf{B}^{-1}(0)\mathbf{m}_1\}\big]\boldsymbol{\phi}(x,\lambda)\\ &+\big[\mathbf{m}_4^T\{-\{\mathbf{m}_1^T\boldsymbol{\eta}_0\}\mathbf{B}^{-1}(0)\mathbf{m}_2 + \{\mathbf{m}_2^T\boldsymbol{\eta}_0\}\mathbf{B}^{-1}(0)\mathbf{m}_1\}\big]\boldsymbol{\psi}(x,\lambda)\end{aligned}$$

or

$$\mathbf{b}(x,\lambda) = \big[-\beta_{32}\{\mathbf{m}_1^T\boldsymbol{\eta}_0\} + \beta_{31}\{\mathbf{m}_2^T\boldsymbol{\eta}_0\}\big]\boldsymbol{\phi}(x,\lambda) + \big[-\beta_{42}\{\mathbf{m}_1^T\boldsymbol{\eta}_0\} + \beta_{41}\{\mathbf{m}_2^T\boldsymbol{\eta}_0\}\big]\boldsymbol{\psi}(x,\lambda).$$

Now we have

$$b(x,\lambda) = \big[-\beta_{32}\{\mathbf{m}_1^T\boldsymbol{\eta}_0\} + \beta_{31}\{\mathbf{m}_2^T\boldsymbol{\eta}_0\}\big]\phi(x,\lambda) + \big[-\beta_{42}\{\mathbf{m}_1^T\boldsymbol{\eta}_0\} + \beta_{41}\{\mathbf{m}_2^T\boldsymbol{\eta}_0\}\big]\psi(x,\lambda). \tag{2.47}$$

Knowing $a(x,\lambda)$ and $b(x,\lambda)$, we can find an expression for $g(x,\xi;\lambda)$ in terms of the fundamental solution, the boundary operators $\mathbf{m}_i$ and the matrix $\mathbf{B}(0)$, substituting for $x < \xi$ in (2.36),

$$W(\lambda)g(\xi,x;\lambda) = a(x,\lambda)\eta(\xi,\lambda) + b(x,\lambda)\zeta(\xi,\lambda), \quad x < \xi$$

and using (2.44) and (2.47).

Because of the symmetry of $g(x,\xi;\lambda)$ with respect to its arguments, (2.36) can be written as,

$$g(\xi,x;\lambda) = \frac{1}{W(\lambda)}\begin{cases} -c(\xi,\lambda)\phi(x,\lambda) - d(\xi,\lambda)\psi(x,\lambda), & x < \xi \\ -c(x,\lambda)\phi(\xi,\lambda) - d(x,\lambda)\psi(\xi,\lambda), & x > \xi. \end{cases} \tag{2.48}$$

or

$$g(\xi,x;\lambda) = \frac{1}{W(\lambda)}\begin{cases} a(x,\lambda)\eta(\xi,\lambda) + b(x,\lambda)\zeta(\xi,\lambda), & x < \xi \\ a(\xi,\lambda)\eta(x,\lambda) + b(\xi,\lambda)\zeta(x,\lambda), & x > \xi. \end{cases}$$

so that

$$W(\lambda)g(\xi,x;\lambda) = a(\xi,\lambda)\eta(x,\lambda) + b(\xi,\lambda)\zeta(x,\lambda), \quad x > \xi \tag{2.49}$$

Now setting $x \longrightarrow \xi$, in (2.44),

$$a(\xi,\lambda) = \left[\beta_{32}\left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\} - \beta_{31}\left\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\right\}\right]\phi(\xi,\lambda) + \left[\beta_{42}\left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\} - \beta_{41}\left\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\right\}\right]\psi(\xi,\lambda)$$

and setting $x \longrightarrow \xi$, in (2.47),

$$b(\xi,\lambda) = \left[-\beta_{32}\left\{\mathbf{m}_1^T\boldsymbol{\eta}_0\right\} + \beta_{31}\left\{\mathbf{m}_2^T\boldsymbol{\eta}_0\right\}\right]\phi(\xi,\lambda) + \left[-\beta_{42}\left\{\mathbf{m}_1^T\boldsymbol{\eta}_0\right\} + \beta_{41}\left\{\mathbf{m}_2^T\boldsymbol{\eta}_0\right\}\right]\psi(\xi,\lambda).$$

Equation (2.49) reads as,

$$\begin{aligned}
&W(\lambda)g(\xi,x;\lambda)\\
=&\left[\beta_{32}\left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\} - \beta_{31}\left\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\right\}\right]\phi(\xi,\lambda)\eta(x,\lambda)\\
&+\left[\beta_{42}\left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\} - \beta_{41}\left\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\right\}\right]\psi(\xi,\lambda)\eta(x,\lambda)\\
&+\left[-\beta_{32}\left\{\mathbf{m}_1^T\boldsymbol{\eta}_0\right\} + \beta_{31}\left\{\mathbf{m}_2^T\boldsymbol{\eta}_0\right\}\right]\phi(\xi,\lambda)\zeta(x,\lambda)\\
&+\left[-\beta_{42}\left\{\mathbf{m}_1^T\boldsymbol{\eta}_0\right\} + \beta_{41}\left\{\mathbf{m}_2^T\boldsymbol{\eta}_0\right\}\right]\psi(\xi,\lambda)\zeta(x,\lambda), \quad x > \xi\\
=&\left\{\left[\beta_{32}\left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\} - \beta_{31}\left\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\right\}\right]\eta(x,\lambda) + \left[-\beta_{32}\left\{\mathbf{m}_1^T\boldsymbol{\eta}_0\right\} + \beta_{31}\left\{\mathbf{m}_2^T\boldsymbol{\eta}_0\right\}\right]\zeta(x,\lambda)\right\}\phi(\xi,\lambda)\\
&+\left\{\left[\beta_{42}\left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\} - \beta_{41}\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\}\right]\eta(x,\lambda) + \left[-\beta_{42}\left\{\mathbf{m}_1^T\boldsymbol{\eta}_0\right\} + \beta_{41}\left\{\mathbf{m}_2^T\boldsymbol{\eta}_0\right\}\right]\zeta(x,\lambda)\right\}\psi(\xi,\lambda)\\
=&\left\{\beta_{31}\left[\left\{\mathbf{m}_2^T\boldsymbol{\eta}_0\right\}\zeta(x,\lambda) - \left\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\right\}\eta(x,\lambda)\right] - \beta_{32}\left[\left\{\mathbf{m}_1^T\boldsymbol{\eta}_0\right\}\zeta(x,\lambda) - \left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\}\eta(x,\lambda)\right]\right\}\phi(\xi,\lambda)\\
&+\left\{\beta_{41}\left[\left\{\mathbf{m}_2^T\boldsymbol{\eta}_0\right\}\zeta(x,\lambda) - \left\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\right\}\eta(x,\lambda)\right] - \beta_{42}\left[\left\{\mathbf{m}_1^T\boldsymbol{\eta}_0\right\}\zeta(x,\lambda) - \left\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\right\}\eta(x,\lambda)\right]\right\}\psi(\xi,\lambda)
\end{aligned}$$

From (2.36)

$$g(\xi,x;\lambda)W(\lambda) = \begin{cases} a(x,\lambda)\eta(\xi,\lambda) + b(x,\lambda)\zeta(\xi,\lambda), & x < \xi \\ -c(x,\lambda)\phi(\xi,\lambda) - d(x,\lambda)\psi(\xi,\lambda), & x > \xi \end{cases} \tag{2.50}$$

we see that for $x > \xi$,

$$\begin{aligned}
c(x,\lambda) =&\beta_{31}\left[\left\{\mathbf{m}_2^T\boldsymbol{\zeta}(0,\lambda)\right\}\eta(x,\lambda) - \left\{\mathbf{m}_2^T\boldsymbol{\eta}(0,\lambda)\right\}\zeta(x,\lambda)\right]\\
&-\beta_{32}\left[\left\{\mathbf{m}_1^T\boldsymbol{\zeta}(0,\lambda)\right\}\eta(x,\lambda) - \left\{\mathbf{m}_1^T\boldsymbol{\eta}(0,\lambda)\right\}\zeta(x,\lambda)\right]\\
d(x,\lambda) =&\beta_{41}\left[\left\{\mathbf{m}_2^T\boldsymbol{\zeta}(0,\lambda)\right\}\eta(x,\lambda) - \left\{\mathbf{m}_2^T\boldsymbol{\eta}(0,\lambda)\right\}\zeta(x,\lambda)\right]\\
&-\beta_{42}\left[\left\{\mathbf{m}_1^T\boldsymbol{\zeta}(0,\lambda)\right\}\eta(x,\lambda) - \left\{\mathbf{m}_1^T\boldsymbol{\eta}(0,\lambda)\right\}\zeta(x,\lambda)\right]
\end{aligned}$$

From the above expressions for $c(x,\lambda)$ and $d(x,\lambda)$, it is clear that as $\lambda \longrightarrow \lambda_n$, an eigenvalue of (2.2)-(2.4), both $c(x,\lambda)$ and $d(x,\lambda)$ tend toward an eigenfunction

(since $\zeta(x,\lambda)$ and $\eta(x,\lambda)$ tend towards eigenfunctions), namely

$$c(x,\lambda_n) = C_n u_n(x), \tag{2.51}$$
$$d(x,\lambda_n) = D_n u_n(x). \tag{2.52}$$

▶ **Lemma 2.8.** $\{C_n\}_1^\infty$ and $\{D_n\}_1^\infty$ are completely determined by $S(\mathbf{m}_1,\mathbf{m}_2)$, $S(\mathbf{m}_i,\mathbf{m}_3)$, $(i=1,2)$, the boundary operators $\mathbf{m}_1$, $\mathbf{m}_2$, $\mathbf{n}_1$, $\mathbf{n}_2$ and $p(0)$. ◀

Proof. Let us normalize the eigenfunctions $\{u_n(x)\}_1^\infty$ by means of the condition

$$\mathbf{m}_3^T \mathbf{u}_n(0) = 1. \tag{2.53}$$

Then

$$\begin{aligned} c(x,\lambda) =& \beta_{31}\Big[\Big\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\eta}_0\Big\}\eta(x,\lambda) - \Big\{\mathbf{m}_2^T\boldsymbol{\eta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\zeta}_0\Big\}\zeta(x,\lambda)\Big] \\ &- \beta_{32}\Big[\Big\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\eta}_0\Big\}\eta(x,\lambda) - \Big\{\mathbf{m}_1^T\boldsymbol{\eta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\zeta}_0\Big\}\zeta(x,\lambda)\Big] \end{aligned} \tag{2.54}$$

$$\begin{aligned} d(x,\lambda) =& \beta_{41}\Big[\Big\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\eta}_0\Big\}\eta(x,\lambda) - \Big\{\mathbf{m}_2^T\boldsymbol{\eta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\zeta}_0\Big\}\zeta(x,\lambda)\Big] \\ &- \beta_{42}\Big[\Big\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\eta}_0\Big\}\eta(x,\lambda) - \Big\{\mathbf{m}_1^T\boldsymbol{\eta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\zeta}_0\Big\}\zeta(x,\lambda)\Big] \end{aligned} \tag{2.55}$$

then

$$c(x,\lambda_n) = \underbrace{\left\{\beta_{31}\underbrace{\left[\Big\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\eta}_0\Big\} - \Big\{\mathbf{m}_2^T\boldsymbol{\eta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\zeta}_0\Big\}\right]}_{W_2(\lambda_n)} - \beta_{32}\underbrace{\left[\Big\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\eta}_0\Big\} - \Big\{\mathbf{m}_1^T\boldsymbol{\eta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\zeta}_0\Big\}\right]}_{W_1(\lambda_n)}\right\}}_{C_n} u_n(x)$$

$$d(x,\lambda_n) = \underbrace{\left\{\beta_{41}\underbrace{\left[\Big\{\mathbf{m}_2^T\boldsymbol{\zeta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\eta}_0\Big\} - \Big\{\mathbf{m}_2^T\boldsymbol{\eta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\zeta}_0\Big\}\right]}_{W_2(\lambda_n)} - \beta_{42}\underbrace{\left[\Big\{\mathbf{m}_1^T\boldsymbol{\zeta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\eta}_0\Big\} - \Big\{\mathbf{m}_1^T\boldsymbol{\eta}_0\Big\}\Big\{\mathbf{m}_3^T\boldsymbol{\zeta}_0\Big\}\right]}_{W_1(\lambda_n)}\right\}}_{D_n} u_n(x)$$

so that

$$C_n = \beta_{31}W_2(\lambda_n) - \beta_{32}W_1(\lambda_n) \tag{2.56}$$

and

$$D_n = \beta_{41}W_2(\lambda_n) - \beta_{42}W_1(\lambda_n) \tag{2.57}$$

where

$$W_1(\lambda) = \left\{\mathbf{m}_1^T \boldsymbol{\zeta}(0,\lambda)\right\}\left\{\mathbf{m}_3^T \boldsymbol{\eta}(0,\lambda)\right\} - \left\{\mathbf{m}_1^T \boldsymbol{\eta}(0,\lambda)\right\}\{\mathbf{m}_3^T \boldsymbol{\zeta}(0,\lambda)\},$$
$$W_2(\lambda) = \left\{\mathbf{m}_2^T \boldsymbol{\zeta}(0,\lambda)\right\}\left\{\mathbf{m}_3^T \boldsymbol{\eta}(0,\lambda)\right\} - \left\{\mathbf{m}_2^T \boldsymbol{\eta}(0,\lambda)\right\}\left\{\mathbf{m}_3^T \boldsymbol{\zeta}(0,\lambda)\right\}.$$

From (2.12)

$$W(\lambda) = \left\{\mathbf{m}_1^T \boldsymbol{\eta}(0,\lambda)\right\}\left\{\mathbf{m}_2^T \boldsymbol{\zeta}(0,\lambda)\right\} - \left\{\mathbf{m}_2^T \boldsymbol{\eta}(0,\lambda)\right\}\left\{\mathbf{m}_1^T \boldsymbol{\zeta}(0,\lambda)\right\}$$

which is the Wronskian of the solutions associated with the differential operator $\mathcal{L} - \lambda$ and the boundary conditions

$$\mathbf{m}_1^T \mathbf{u}(0) = \mathbf{m}_2^T \mathbf{u}(0) = \mathbf{n}_1^T \mathbf{u}(1) = \mathbf{n}_2^T \mathbf{u}(1) = 0.$$

We can see that, $W_1(\lambda)$ is the Wronskian of the solutions associated with the differential operator $\mathcal{L} - \lambda$ and the boundary conditions

$$\mathbf{m}_1^T \mathbf{u}(0) = \mathbf{m}_3^T \mathbf{u}(0) = \mathbf{n}_1^T \mathbf{u}(1) = \mathbf{n}_2^T \mathbf{u}(1) = 0.$$

and $W_2(\lambda)$ is the Wronskian of the solutions associated with the differential operator $\mathcal{L} - \lambda$ and the boundary conditions

$$\mathbf{m}_2^T \mathbf{u}(0) = \mathbf{m}_3^T \mathbf{u}(0) = \mathbf{n}_1^T \mathbf{u}(1) = \mathbf{n}_2^T \mathbf{u}(1) = 0.$$

By Lemma 2.5, it follows that $W_1(\lambda)$ is completely determined by the spectra $S(\mathbf{m}_1, \mathbf{m}_3)$, and $W_2(\lambda)$ is completely determined by the spectra $S(\mathbf{m}_2, \mathbf{m}_3)$.
Hence, from (2.56) and (2.57), $\{C_n\}_1^\infty$ and $\{D_n\}_1^\infty$ are completely determined by $S(\mathbf{m}_1, \mathbf{m}_2)$, $S(\mathbf{m}_i, \mathbf{m}_3)$, $(i = 1, 2)$, the boundary operators $\mathbf{m}_1$, $\mathbf{m}_2$, $\mathbf{n}_1$, $\mathbf{n}_2$ and $p(0)$. It should be emphasized that the evaluation of $\{C_n\}_1^\infty$ and $\{D_n\}_1^\infty$ can be carried out without a knowledge of $p(x)$ and $q(x)$. ■

► **Lemma 2.9.**

$$-C_n\phi(\xi,\lambda_n) - D_n\psi(\xi,\lambda_n) = -C_n u_n(\xi). \tag{2.58}$$

◄

Proof. From the properties of the Green's function, we know that

$$\lim_{\lambda\to\lambda_n} W(\lambda)g(x,\xi;\lambda) = \kappa_n u_n(x)u_n(\xi), \tag{2.59}$$

From (2.50)

$$g(\xi,x;\lambda)W(\lambda) = \begin{cases} a(x,\lambda)\eta(\xi,\lambda) + b(x,\lambda)\zeta(\xi,\lambda), & x < \xi \\ -c(x,\lambda)\phi(\xi,\lambda) - d(x,\lambda)\psi(\xi,\lambda), & x > \xi \end{cases}$$

we see that

$$\lim_{\lambda\to\lambda_n} W(\lambda)g(x,\xi;\lambda) = \lim_{\lambda\to\lambda_n}\{-c(x,\lambda)\phi(\xi,\lambda) - d(x,\lambda)\psi(\xi,\lambda)\}$$

and from (2.51) and (2.52)

$$\begin{aligned}
\lim_{\lambda\to\lambda_n} W(\lambda)g(x,\xi;\lambda) &= -C_n u_n(x)\phi(\xi,\lambda_n) - D_n u_n(x)\psi(\xi,\lambda_n) \\
\kappa_n u_n(x)u_n(\xi) &= -C_n u_n(x)\phi(\xi,\lambda_n) - D_n u_n(x)\psi(\xi,\lambda_n), \quad \because (2.59) \qquad (2.60)\\
\implies \kappa_n u_n(\xi) &= -C_n\phi(\xi,\lambda_n) - D_n\psi(\xi,\lambda_n) \qquad (2.61)
\end{aligned}$$

The constant κ_n can be determined by operating on the above equation with $\mathbf{m}_3^T$ and setting $\xi = 0$; this yields

$$\kappa_n \underbrace{\mathbf{m}_3^T u_n(0)}_{1} = -C_n \underbrace{\mathbf{m}_3^T\phi(0,\lambda_n)}_{1} - D_n \underbrace{\mathbf{m}_3^T\psi(0,\lambda_n)}_{0} \quad \because (2.53), (2.8a), (2.8b)$$

$$\kappa_n = -C_n$$

Hence (2.61) reads as

$$-C_n u_n(\xi) = -C_n\phi(\xi,\lambda_n) - D_n\psi(\xi,\lambda_n)$$

■

2.3 Uniqueness of Inverse FSLP

▶ **Theorem 2.10. Uniqueness of Inverse FSLP**
Given

1. the differential operator $\mathcal{L}$ defined in Equation (2.2):

$$\mathcal{L}u \equiv u^{(4)} - (pu')' + qu = \lambda u, \quad x \in (0,1)$$
$$\mathcal{L}u = \lambda u$$

2. the three spectra $S(\mathbf{m}_1,\mathbf{m}_2)$, $S(\mathbf{m}_1,\mathbf{m}_3)$, $S(\mathbf{m}_2,\mathbf{m}_3)$ satisfying Equation (2.5):

$$\mathbf{m}_i^T\mathbf{u}(0) = \mathbf{n}_i^T\mathbf{u}(1) = 0, \qquad i = 1,2,3$$

3. the value of $p(0)$,

then $p(x)$ and $q(x)$ are uniquely determined. ◀

Proof. Consider the three pairs of eigenvalue problems:

$$\left.\begin{array}{ll} \mathcal{L}u_n = \lambda_n u_n, & \mathbf{m}_1^T \mathbf{u}_n(0) = \mathbf{m}_2^T \mathbf{u}_n(0) = 0, \\ \hat{\mathcal{L}}\hat{u}_n = \lambda_n \hat{u}_n, & \mathbf{m}_1^T \hat{\mathbf{u}}_n(0) = \mathbf{m}_2^T \hat{\mathbf{u}}_n(0) = 0, \end{array}\right\} \tag{2.62}$$

$$\left.\begin{array}{ll} \mathcal{L}v_n = \mu_n v_n, & \mathbf{m}_1^T \mathbf{v}_n(0) = \mathbf{m}_3^T \mathbf{v}_n(0) = 0, \\ \hat{\mathcal{L}}\hat{v}_n = \mu_n \hat{v}_n, & \mathbf{m}_1^T \hat{\mathbf{v}}_n(0) = \mathbf{m}_3^T \hat{\mathbf{v}}_n(0) = 0, \end{array}\right\} \tag{2.63}$$

$$\left.\begin{array}{ll} \mathcal{L}w_n = \nu_n w_n, & \mathbf{m}_2^T \mathbf{w}_n(0) = \mathbf{m}_3^T \mathbf{w}_n(0) = 0, \\ \hat{\mathcal{L}}\hat{w}_n = \nu_n \hat{w}_n, & \mathbf{m}_2^T \hat{\mathbf{w}}_n(0) = \mathbf{m}_3^T \hat{\mathbf{w}}_n(0) = 0, \end{array}\right\} \tag{2.64}$$

and $\mathbf{n}_1^T \cdot (1) = \mathbf{n}_2^T \cdot (1) = 0$.

The operators $\mathcal{L}$ and $\hat{\mathcal{L}}$ are defined as:

$$\begin{aligned} \mathcal{L}u &\equiv u^{(4)} - (pu')' + qu \\ \hat{\mathcal{L}}\hat{u} &\equiv \hat{u}^{(4)} - (\hat{p}\hat{u}')' + \hat{q}\hat{u}. \end{aligned}$$

Assume that each of the two eigenvalue problems constituting a pair have the same spectrum, i.e.

$$\left.\begin{array}{ll} S(\mathbf{m}_1, \mathbf{m}_2) & = \hat{S}(\mathbf{m}_1, \mathbf{m}_2), \\ S(\mathbf{m}_2, \mathbf{m}_3) & = \hat{S}(\mathbf{m}_2, \mathbf{m}_3), \\ S(\mathbf{m}_1, \mathbf{m}_3) & = \hat{S}(\mathbf{m}_1, \mathbf{m}_3) \end{array}\right\}$$

To prove the theorem, we need to show that,

$$\boxed{P(x) \equiv p(x) - \hat{p}(x) = 0,} \tag{2.65}$$

$$\boxed{Q(x) \equiv q(x) - \hat{q}(x) = 0.} \tag{2.66}$$

Also, assume that

$$\begin{aligned} & \hat{p}(0) = p(0) \\ \Longrightarrow\ & P(0) = 0 \end{aligned} \tag{2.67}$$

Let us introduce two contour integrals:

$$I_n(\xi) = \frac{1}{2\pi i} \oint_{\Gamma_n} d\lambda \int_0^1 f(x) g(x, \xi; \lambda) dx$$

and

$$J_n(\xi) = \frac{1}{2\pi i}\oint_{\Gamma_n} d\lambda \int_0^1 f(x)\gamma(x,\xi;\lambda)dx.$$

Here, $f(x)$ is an arbitrary, real, differentiable function, and Γ_n is a circle of radius r_n where

$$\lambda_n < r_n < \lambda_{n+1}, \qquad n = 1, 2, \ldots.$$

λ_n is the nth eigenvalue of Equation (2.62). The function $\gamma(x,\xi;\lambda)$ which is the green's function associated with the operator $\hat{\mathcal{L}}$ is defined as follows (similar to (2.48)):

$$W(\lambda)\gamma(x,\xi;\lambda) = \begin{cases} -\hat{c}(x,\lambda)\phi(\xi,\lambda) - \hat{d}(x,\lambda)\psi(\xi,\lambda), & x > \xi \\ -c(\xi,\lambda)\hat{\phi}(x,\lambda) - d(\xi,\lambda)\hat{\psi}(x,\lambda), & x < \xi \end{cases}$$

where (similar to (2.54) and (2.55)),

$$\begin{aligned}
\hat{c}(x,\lambda) =& \beta_{31}\left[\left\{\mathbf{m}_2^T\hat{\boldsymbol{\zeta}}(0,\lambda)\right\}\hat{\eta}(x,\lambda) - \left\{\mathbf{m}_2^T\hat{\boldsymbol{\eta}}(0,\lambda)\right\}\hat{\zeta}(x,\lambda)\right] \\
&- \beta_{32}\left[\left\{\mathbf{m}_1^T\hat{\boldsymbol{\zeta}}(0,\lambda)\right\}\hat{\eta}(x,\lambda) - \left\{\mathbf{m}_1^T\hat{\boldsymbol{\eta}}(0,\lambda)\right\}\hat{\zeta}(x,\lambda)\right] \\
\hat{d}(x,\lambda) =& \beta_{41}\left[\left\{\mathbf{m}_2^T\hat{\boldsymbol{\zeta}}(0,\lambda)\right\}\hat{\eta}(x,\lambda) - \left\{\mathbf{m}_2^T\hat{\boldsymbol{\eta}}(0,\lambda)\right\}\hat{\zeta}(x,\lambda)\right] \\
&- \beta_{42}\left[\left\{\mathbf{m}_1^T\hat{\boldsymbol{\zeta}}(0,\lambda)\right\}\hat{\eta}(x,\lambda) - \left\{\mathbf{m}_1^T\hat{\boldsymbol{\eta}}(0,\lambda)\right\}\hat{\zeta}(x,\lambda)\right]
\end{aligned}$$

Here, $\hat{\eta}, \hat{\zeta}, \hat{\phi}$ and $\hat{\psi}$ are the solutions of $(\hat{\mathcal{L}} - \lambda)u = 0$, which satisfy the boundary conditions (similar to (2.8)):

$$\begin{aligned}
\mathbf{m}_i^T\hat{\boldsymbol{\phi}}(0,\lambda) &= \delta_{i3}, \\
\mathbf{m}_i^T\hat{\boldsymbol{\psi}}(0,\lambda) &= \delta_{i4}, \\
\mathbf{m}_i^T\hat{\boldsymbol{\eta}}(1,\lambda) &= \delta_{i3}, \\
\mathbf{m}_i^T\hat{\boldsymbol{\zeta}}(1,\lambda) &= \delta_{i4}
\end{aligned}$$

From Theorem 1.8 the zeros of $W(\lambda)$ are simple (since the problem is self-adjoint). Using the calculus of residues,

$$\begin{aligned}
J_n(\xi) =& \sum_{k=1}^{n}\frac{1}{W'(\lambda_k)}\int_0^{\xi}\left\{-c(\xi,\lambda_k)\hat{\phi}(x,\lambda_k) - d(\xi,\lambda_k)\hat{\psi}(x,\lambda_k)\right\}f(x)dx \\
&+ \sum_{k=1}^{n}\frac{1}{W'(\lambda_k)}\int_{\xi}^{1}\left\{-\hat{c}(x,\lambda_k)\phi(\xi,\lambda_k) - \hat{d}(x,\lambda_k)\psi(\xi,\lambda_k)\right\}f(x)dx
\end{aligned}$$

But, from equation (2.51), $c(\xi,\lambda_k) = C_k u_k(\xi)$ and $\hat{c}(x,\lambda_k) = C_k \hat{u}_k(x)$, since the two constants of proportionality are identical as they depend solely on the spectra $\{\lambda_n\}_1^\infty$, $\{\mu_n\}_1^\infty$, $\{v_n\}_1^\infty$, m_{ij}, and $p(0)$ (by Lemma 2.8). Similarly, from equation (2.52), $d(\xi,\lambda_k) = D_k u_k(\xi)$ and $\hat{d}(x,\lambda_k) = D_k \hat{u}_k(x)$. So,

$$J_n(\xi) = \sum_{k=1}^{n} \frac{u_k(\xi)}{W'(\lambda_k)} \int_0^\xi \left\{-C_k\hat{\phi}(x,\lambda_k) - D_k\hat{\psi}(x,\lambda_k)\right\} f(x)dx + \sum_{k=1}^{n} \frac{1}{W'(\lambda_k)} \{-C_k\phi(\xi,\lambda_k) - D_k\psi(\xi,\lambda_k)\} \int_\xi^1 \hat{u}_k(x) f(x)dx, \tag{2.68}$$

or from Equation (2.58) (Lemma 2.9),

$$-C_k\hat{\phi}(x,\lambda_k) - D_k\hat{\psi}(x,\lambda_k) = -C_k\hat{u}_k(x)$$

and

$$-C_k\phi(\xi,\lambda_k) - D_k\psi(\xi,\lambda_k) = -C_k u_k(\xi)$$

so that (2.68) reduces to

$$J_n(\xi) = \sum_{k=1}^{n} \frac{u_k(\xi)}{W'(\lambda_k)} \int_0^\xi \{-C_k\hat{u}_k(x)\} f(x)dx + \sum_{k=1}^{n} \frac{1}{W'(\lambda_k)} \{-C_k u_k(\xi)\} \int_\xi^1 \hat{u}_k(x) f(x)dx$$

$$J_n(\xi) = -\sum_{k=1}^{n} \frac{C_k u_k(\xi)}{W'(\lambda_k)} \int_0^1 \hat{u}_k(x) f(x)dx \tag{2.69}$$

Similarly,

$$I_n(\xi) = \sum_{k=1}^{n} \frac{1}{W'(\lambda_k)} \int_0^\xi \{-c(\xi,\lambda_k)\phi(x,\lambda_k) - d(\xi,\lambda_k)\psi(x,\lambda_k)\} f(x)dx + \sum_{k=1}^{n} \frac{1}{W'(\lambda_k)} \int_\xi^1 \{-c(x,\lambda_k)\phi(\xi,\lambda_k) - d(x,\lambda_k)\psi(\xi,\lambda_k)\} f(x)dx$$

But, from equation (2.51), $c(\xi,\lambda_k) = C_k u_k(\xi)$ and $c(x,\lambda_k) = C_k u_k(x)$, since the two constants of proportionality are identical as they depend solely on the spectra $\{\lambda_n\}_1^\infty$, $\{\mu_n\}_1^\infty$, $\{v_n\}_1^\infty$, m_{ij}, and $p(0)$ (by Lemma 2.8). Similarly, from equation (2.52), $d(\xi,\lambda_k) =$

$D_k u_k(\xi)$ and $d(x,\lambda_k) = D_k u_k(x)$. So,

$$I_n = \sum_{k=1}^{n} \frac{u_k(\xi)}{W'(\lambda_k)} \int_0^{\xi} \{-C_k\phi(x,\lambda_k) - D_k\psi(x,\lambda_k)\} f(x)dx + \sum_{k=1}^{n} \frac{1}{W'(\lambda_k)} \{-C_k\phi(\xi,\lambda_k) - D_k\psi(\xi,\lambda_k)\} \int_{\xi}^{1} u_k(x) f(x)dx, \tag{2.70}$$

or from Equation (2.58) (Lemma 2.9),

$$-C_k\phi(x,\lambda_k) - D_k\psi(x,\lambda_k) = -C_k u_k(x)$$

and

$$-C_k\phi(\xi,\lambda_k) - D_k\psi(\xi,\lambda_k) = -C_k u_k(\xi)$$

so that (2.70) reduces to

$$I_n(\xi) = \sum_{k=1}^{n} \frac{u_k(\xi)}{W'(\lambda_k)} \int_0^{\xi} \{-C_k\hat{u}_k(x)\} f(x)dx + \sum_{k=1}^{n} \frac{1}{W'(\lambda_k)} \{-C_k u_k(\xi)\} \int_{\xi}^{1} \hat{u}_k(x) f(x)dx$$

$$I_n(\xi) = -\sum_{k=1}^{n} \frac{C_k u_k(\xi)}{W'(\lambda_k)} \int_0^{1} u_k(x) f(x)dx \tag{2.71}$$

As $n \to \infty$, it is possible to evaluate the limit of $I_n(\xi)$ and $J_n(\xi)$ by replacing ϕ, ψ, η, ζ and $\hat{\phi}, \hat{\psi}, \hat{\eta}, \hat{\zeta}$ by the first term of their respective asymptotic expansions for $|\lambda| \to \infty$. Hence,

$$\lim_{n\to\infty} I_n(\xi) = \lim_{n\to\infty} J_n(\xi). \tag{2.72}$$

That is

$$-\sum_{k=1}^{\infty} \frac{C_k u_k(\xi)}{W'(\lambda_k)} \int_0^{1} u_k(x) f(x)dx = -\sum_{k=1}^{\infty} \frac{C_k u_k(\xi)}{W'(\lambda_k)} \int_0^{1} \hat{u}_k(x) f(x)dx$$

$$\implies -\sum_{k=1}^{\infty} \frac{C_k u_k(\xi)}{W'(\lambda_k)} \int_0^{1} (u_k(x) - \hat{u}_k(x)) f(x)dx = 0.$$

By orthogonality of the eigenfunctions $\{u_k(x)\}_1^\infty$,

$$\int_0^1 f(x)(u_k(x) - \hat{u}_k(x))dx = 0$$

Since $f(x)$ arbitrary, (for example, $f(x) \neq 0$)

$$\implies u_k(x) = \hat{u}_k(x), \qquad k = 1, 2, \ldots$$

From the pair of eigenvalue problems (2.62), we see that,

$$\begin{aligned}
\lambda_n \hat{u}_n &= \lambda_n u_n \\
\implies \hat{\mathcal{L}}\hat{u}_n &= \mathcal{L}u_n \\
\implies \cancelto{0}{\hat{u}_n^{(4)}} - (\hat{p}\hat{u}_n')' + \hat{q}\hat{u}_n &= \cancelto{0}{u_n^{(4)}} - (pu_n')' + qu_n. \\
\implies -(\hat{p}\hat{u}_n')' + (pu_n')' &= qu_n - \hat{q}\hat{u}_n \\
\implies ((p - \hat{p})u_n')' &= (q - \hat{q})u_n, \qquad \because u_n(x) = \hat{u}_n(x)
\end{aligned}$$

From equations (2.65) and (2.66),

$$(Pu_n')' = Qu_n. \tag{2.73}$$

Let $f(x)$ be an arbitrary differentiable function such that

$$f(1) = f'(1) = 0 \tag{2.74}$$

Multiplying (2.73) by f and integrating over $(0, 1)$, we get:

$$\begin{aligned}
\int_0^1 (Pu_n')' f dx &= \int_0^1 Q f u_n dx \\
\left[Pu_n' f\right]_0^1 - \int_0^1 (Pu_n')f' dx &= \int_0^1 Q f u_n dx \\
\cancelto{0}{P(1)u_n'(1)\cancelto{0}{f(1)}} - \cancelto{0}{P(0)u_n'(0)}f(0) - \int_0^1 (Pu_n')f' dx &= \int_0^1 Q f u_n dx, \qquad \because (2.74), (2.67)
\end{aligned}$$

$$-\int_0^1 (Pf')u_n' dx = \int_0^1 Q f u_n dx.$$

Integrating once more,

$$-\left[(Pf')u_n'\right]_0^1 + \int_0^1 (Pf')'u_n dx = \int_0^1 Qfu_n dx$$

$$-P(1)\cancelto{0}{f'(1)}u_n'(1) + \cancelto{0}{P(0)}\cancelto{0}{f'(0)}u_n'(0) + \int_0^1 (Pf')'u_n dx = \int_0^1 Qfu_n dx, \qquad \because (2.74), (2.67)$$

$$\implies \int_0^1 ((Pf')' - Qf)u_n dx = 0,$$

and due to the completeness of $\{u_n(x)\}_1^\infty$

$$(Pf')' = Qf. \tag{2.75}$$

Integrating again,

$$\int_0^1 d(Pf')' = \int_0^1 Qf dx$$

$$[Pf']_0^1 = \int_0^1 Qf dx$$

$$P(1)\cancelto{0}{f'(1)} - \cancelto{0}{P(0)}f'(0) = \int_0^1 Qf dx$$

$$\implies \int_0^1 Qf dx = 0. \tag{2.76}$$

Since f is arbitrary,

$$Q(x) \equiv 0.$$

Integrating, equation (2.75),

$$\int_0^1 d(Pf') = \int_0^1 Qf dx = 0, \qquad \because (2.76)$$
$$\implies Pf' = 0$$

which means, since f is arbitrary,

$$P(x) \equiv 0.$$

This completes the proof. ■

2.4 Uniqueness of Inverse SLP of even order

The above proof can be generalized to $2n$th order differential operator as follows. The Green's function expression (2.14) must be replaced by

$$W(\lambda)g(x,\xi;\lambda) = -\sum_{k=1}^{n} c^{(k)}(x,\lambda)\phi_k(\xi,\lambda), \qquad x > \xi,$$

where $\phi_k(x,\lambda)$, $k = 1, 2, \ldots, n$ are the n linearly independent solutions which satisfy the boundary conditions at $x = 0$. As λ tends toward an eigenvalue of the basic eigenfunction problem, each of the functions $c^{(k)}(x,\lambda)$ tends to a multiple of the corresponding eigenfunction. In other words, (2.51)-(2.52) generalize to

$$c^{(k)}(x,\lambda_l) = C_l^{(k)} u_l(x).$$

The n sequences of constants $\{C^{(k)}\}_{l=1}^{\infty}$ are determined by means of n different spectra. The knowledge of these spectra together with the spectrum of the basic eigenvalue problem (which is needed to compute $W(\lambda)$) are sufficient to guarantee that if $\gamma(x,\xi;\lambda)$ is defined thus:

$$W(\lambda)\gamma(x,\xi;\lambda) = \begin{cases} -\sum_{k=1}^{n} \hat{c}^{(k)}(x,\lambda)\phi_k(\xi,\lambda), & x > \xi, \\ -\sum_{k=1}^{n} c^{(k)}(x,\lambda)\phi_k(\xi,\lambda), & x < \xi \end{cases}$$

where a caret has the same meaning as before, then I_l and J_l defined earlier have the same limit as $l \to \infty$. Consequently,

$$u_l(x) = \hat{u}_l(x).$$

The fact that

$$P_k(x) \equiv p_k(x) - \hat{p}_k(x), \qquad k = 1, 2, \ldots, n$$

are all zero can then be deduced by means of a reasoning similar to one previously used.

3 Solutions of Direct Sturm-Liouville Problems

This chapter is based on two of own publications [92] and [93].

3.1 Introduction

Direct and inverse eigenvalue problems (EVP) of linear differential operators play an important role in all vibration problems in engineering and physics [59]. The theory of direct Sturm-Liouville problems (SLP) started around 1830's in the independent works of Sturm and Liouville.

Consider the $2m$th order, nonsingular, self-adjoint eigenvalue problem:

$$\begin{aligned}&(-1)^m(p_m(x)y^{(m)})^{(m)} + (-1)^{m-1}(p_{m-1}(x)y^{(m-1)})^{(m-1)} \\ &+ \ldots + (p_2(x)y'')'' - (p_1(x)y')' + p_0(x)y = \lambda w(x)y, \qquad a < x < b \end{aligned} \tag{3.1}$$

with appropriate boundary conditions (such as Equation (3.2)). For the Equation (3.1), the direct Sturm–Liouville problem is concerned with determining the λ given the coefficient information p_k, $(0 \leq k \leq m)$.

There is a variety of numerical methods for the solutions of the simplest case of Equation (3.1) known as the direct Sturm-Liouville problem with $m = 1$, most notable ones being SLEIGN [15], five-diagonal fourth order method (FD-FOM) [32], Finite difference method (FDM) [91], Numerov's method (NM) [11], higher order finite difference methods (HOFDM) [108], Finite element method (FEM) [10], Modified Numerov's method (MNM) [110], SLEIGN2 [14], Pruess method [74], SLEGDGE [94], FEM with trigonometric hat functions using Simpson's rule (FEMS) and using trapezoidal rule (FEMT) [19], Lie-Group methods (LGM) [84], MATSLISE [61], Chebyshev collocation method [25], differential quadrature (DQ) method [111], functional-discrete (FD-) method [72], sampling theory [29], Legendre–Galerkin– Chebyshev collocation method (LGCC) [33], Boundary Value Method (BVM) [2, 4], Magnus integrators (MI) [60], Dirichlet spectra method [37, 38, 39], mapped barycentric Chebyshev differentiation matrix (MBCDM) [113], and Homotopy Perturbation Method (HPM) [12].

Slightly fewer techniques are available for the direct eigenvalue problem of case $m = 2$ known as fourth order Sturm Liouville problem (FSLP). For example, symmetric five-diagonal finite difference method (FDFDM) [31], seven-diagonal fourth order method (SDFOM) [30], Sturm–Liouville Eigenvalues using Theta matrices (SLEUTH) [49], finite difference method of order two (FDM2) [109], Fliess series [27, 28], Adomian decomposition method (ADM) [13], Chebyshev spectral collocation method (CSCM) [70], Variational Iteration Methods (VIM) [103], Extended Sampling Method (ESM) [26], Homotopy Perturbation Method (HPM) [12], homotopy analysis method (HAM) [1], Differential quadrature method (DQM) and Boubaker polynomials expansion scheme (BPES) [112], Chebychev method (CM) [44], Spectral parameter power series (SPPS) [56], Chebyshev differentiation matrices (CDM) [105], variational iteration method (VIM) [104], Matrix methods (MM) [97], Chebyshev Collocation Method (CCM) [35], Legendre-Galerkin method (LGM) [43], and Lie Group method [83] and functional-discrete (FD-) method [71] are the prominent techniques available.

Handful of methods are available for $m = 3$, or Sixth-order SLP (SSLP). They are shooting method [50], Chebyshev spectral collocation method (CSCM) [70], Adomian decomposition method (ADM) [62], variational iteration methods (VIM) [102], Chebyshev collocation method (CCM) [82], and Chebyshev polynomials (CP) [5].

However, as the order of the direct problem increases, it becomes much harder to solve and also the accuracy of the eigenvalues decreases with the index since only a small portion of numerical eigenvalues are reliable [114].

All of the above mentioned methods are applicable only to a particular order of the SLP and specific set(s) of boundary conditions. In contrast, we propose Lie group method in combination with Magnus expansion to construct a method applicable to solving any order of SLP with arbitrary boundary conditions, which was initially applied for solving SLP by [84] and [60] and FSLP by [83]. The proposed method is universal in the sense that it can be applied to solve a SLP of any (even) order with different types of boundary conditions (including some singular) subject to computational feasibility inherent to large matrix computations.

Section 3.2 will describe the method constructed using Lie group method in combination with Magnus expansion and Section 3.3 provides some numerical examples of direct Sturm–Liouville problems of different orders $m = 1, 2, 3, 4$ with a variety of (including one singular) boundary conditions. Furthermore, the

efficiency and accuracy of the proposed technique is illustrated, using Example 3.1.

3.2 Materials and Methods

3.2.1 Notations

Consider Equation (3.1) under the assumptions:

A1: all coefficient functions are real valued

A2: interval (a, b) is finite

A3: the coefficient functions p_k, $(0 \le k \le m-1)$, w and $1/p_m$ are in $\mathcal{L}^1(a, b)$

A4: infima of p_m and w are both positive.

(The last two assumptions A3, and A4 will be relaxed in Example 3.3). By Greenberg and Marletta [50], (provided the above assumptions are true) one gets:

R1: the eigenvalues are bounded below,

R2: the eigenvalues can be ordered: $\lambda_0 \le \lambda_1 \le \lambda_2 \le \ldots$, and

R3: $\lim_{k\to\infty} \lambda_k = +\infty$.

Defining quasi-derivatives:

$$
\begin{aligned}
u_k &= y^{(k-1)}, \qquad 1 \le k \le m,\\
v_1 &= (-1)^m p_m y^{(m)},\\
v_2 &= (-1)^m (p_m y^{(m)})' + (-1)^{m-1}(p_{m-1} y^{(m-1)}),\\
&\vdots\\
v_k &= (-1)^m (p_m y^{(m)})^{(k-1)} + (-1)^{m-1}(p_{m-1} y^{(m-1)})^{(k-2)} + \ldots + (-1)^{m-k+1}(p_{m-1} y^{(m-k+1)}),\\
&\vdots\\
v_2 &= p_2 y'' - (p_3 y''')' + (p_4 y^{(4)})'' + \ldots + (-1)^{m-2}(p_m y^{(m)})^{(m-2)},\\
&\vdots
\end{aligned}
$$

$$v_m = (-1)^{m-1}(p_m y^{(m)})^{(m-1)} + \ldots - (p_3 y''')'' + (p_2 y'')' - p_1 y',$$

the general, separated, self-adjoint boundary conditions can be written in the form [50]:

$$A_1 u(a) + A_2 v(a) = 0, \qquad B_1 u(b) + B_2 v(b) = 0, \tag{3.2}$$

with

1. A_1, A_2, B_1, B_2 are $m \times m$ real matrices
2. $A_1 A_2^T = A_2 A_1^T$, $B_1 B_2^T = B_2 B_1^T$
3. $m \times 2m$ matrices $(A_1 : A_2)$ and $(B_1 : B_2)$ have rank m.

Equation (3.1) in the matrix form is:

$$U' = G(x)U \tag{3.3}$$

$$AU(a) + BU(b) = 0 \tag{3.4}$$

with

$$G(x) = \left(\begin{array}{ccccc|ccccc} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \ddots & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/p_m(x) & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & p_{m-1}(x) & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \iddots & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & p_2(x) & 0 & 0 & 0 & 0 & 0 & \ddots & 0 \\ 0 & p_1(x) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \lambda w(x) - p_0(x) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right)_{2m\times 2m} \tag{3.5}$$

$$U = [u_1, u_2, \ldots, u_m, v_1, v_2, \ldots, v_m],$$

$$A = \begin{pmatrix} A_1 & A_2 \\ 0_m & 0_m \end{pmatrix}, \qquad B = \begin{pmatrix} 0_m & 0_m \\ B_1 & B_2 \end{pmatrix}.$$

3.2.2 Lie Algebra

The Lie group $(G, \cdot)$ was introduced by Lie [66] as a differentiable manifold with the algebraic structure of a group. The tangent space at identity is called the Lie algebra of G and denoted by $\mathbf{g}$. A binary operation $[\cdot, \cdot] : \mathbf{g} \times \mathbf{g} \to \mathbf{g}$ is called a Lie bracket. Let $x, y \in \mathbf{g}$, then the Lie bracket $[x, y]$ is defined as:

$$[x, y] = xy - yx.$$

Special Lie group is defined as

$$SL(F, n) = \{y \in F^{n \times n}, \quad \det(y) = 1\}.$$

A system of linear ordinary differential equations on $SL(F, n)$ has the form [54]:

$$y' = G(x)y, \qquad \operatorname{tr}(G(x)) = 0 \iff \det(y) = 1. \tag{3.6}$$

Letting $U(x) = Y(x)U(a)$, where $U(a)$ is the initial data, $Y(x)$ satisfies the differential equation

$$Y'(x) = G(x)Y(x), \qquad Y(a) = I \tag{3.7}$$

where I stands for the $n-$dimensional identity matrix [20]. Since $\operatorname{tr}(G(x)) = 0$ we have $\det Y(x) = 1$, so that (3.6) is on the Lie Group $SL(\mathbb{R}^4, 4)$, hence can be solved by Magnus expansion. Let $Y(x)$ be a solution of (3.7) for $x \geq a$, the solution of the system (3.3) with initial condition $U(a)$ is

$$U(x) = Y(x)U(a).$$

Thus, $U(b) = Y(b)U(a)$ and using boundary conditions (3.4):

$$AU(a) + BY(b)U(a) = 0 \implies [A + BY(b)]U(a) = 0.$$

Here, $U(a) \neq 0$. Hence, the eigenvalues λ of the SLP (3.1) are the roots of the characteristic equation:

$$F(\lambda) = \det(A + BY(b)) = 0. \tag{3.8}$$

As the problem (3.1) is self-adjoint, the roots of Equation (3.8) are simple [16], hence can be calculated using Bisection method or any other root finding method. Although an explicit expression for $F(\lambda)$ cannot be found, we can find the roots by computing $F(\lambda)$ for a real number λ. Besides, one can plot the function $F(\lambda)$ to get an idea of the locations of the roots, and to specify a smaller bracket for the bisection method, thus increasing the efficiency of the method. However, the slow convergence and high computational time requirements make it impossible to use bisection method especially when it comes to higher-order SLPs. As the explicit form of F (and it's derivatives) is unknown a Newton-like method cannot apply. To overcome these difficulties we propose a new root finding method to our best knowledge in the following paragraph.

3.2.3 Multisection method

A new root finding method, named: 'Multisection method' – a variant of Bisection method is proposed which will converge to the desired root faster. The idea is to divide the root interval into m subsections ($m = 2$ being bisection method) and locate the sign changing interval. Then, this interval is again refined into m subsections and the sign changing interval is located. This will continue until desired accuracy or the maximum number of iterations reached (See Example A.1 in Appendix A which illustrates the usage and the performance of multisection method).

This method overcomes drawbacks in bisection method, yet provides a simple and faster, derivative-free solution to the root-finding problem. This method is able to find even multiple roots in an interval, and given a sufficiently small subsection-length it is able to find roots which are cluster closer to each other (see Example 3.4). Unlike Bisection method, this method does not require the bracketing interval to have different signs at the endpoints. In the present paper, input to this method is a set of function values, as it does not require to specify the explicit functional form of F, which is another advantage of this method. Due to the faster convergence of the multisection method (even at the risk of more computational cost), it will be used to find the eigenvalues (or roots) of the characteristic function.

3.2.4 Magnus expansion

Magnus [69] proposes a solution to (3.6) as

$$y(x) = \exp \Omega(x)\, y(a),$$

and a series expansion for the exponent

$$\Omega(x) = \sum_{k=1}^{\infty} \Omega_k(x)$$

which is called the Magnus expansion, the first few terms of which can be written as

$$\Omega_1(x) = \int_0^x G(t_1)\, dt_1$$
$$\Omega_2(x) = \frac{1}{2} \int_0^x dt_1 \int_0^{t_2} dt_2\, [G(t_1), G(t_2)]$$
$$\Omega_3(x) = \frac{1}{6} \int_0^x dt_1 \int_0^{t_1} dt_2 \int_0^{t_2} dt_3\, [G(t_1), [G(t_2), G(t_3)]] + [G(t_3), [G(t_2), G(t_1)]]$$
$$\vdots = \vdots$$

where $[A, B] \equiv AB - BA$ is the matrix commutator of A and B.

3.2.5 Numerical Procedure

The Magnus series only converges locally, hence to calculate Ω, the interval $[a, b]$ should be divided into N steps such that the Magnus series converges in each subinterval $[x_{n-1}, x_n]$, $n = 1, \ldots, N$, with $x_N = b$.

Then the solution at x_N is represented by

$$Y(x_N) = \prod_{n=1}^{N} \exp(\Omega(x_{n-1}, x_n))\, Y_0,$$

and the series $\Omega(x_{n-1}, x_n)$ has to be appropriately truncated.

The three steps in the procedure are:

E1: Ω series is truncated at an appropriate order: For achieving an integration

method of order $2s(s > 1)$ only terms up to Ω_{2s-2} in the Ω series are required [20].

E2: The multivariate integrals in the truncated series $\Omega^{[p]} = \sum_{i=1}^{p} \Omega_p$ are replaced by conveniently chosen approximations: if their exact evaluation is not possible or is computationally expensive, a numerical quadrature may be used instead. Suppose that b_i, c_i, $(i = 1, \ldots, k)$, are the weights and nodes of a particular quadrature rule, say X of order p, respectively. Then,

$$G(0) = \int_{t_n}^{t_n+h} G(t)\, dt = h \sum_{i=1}^{k} b_i G_i + O(h^{p+1}),$$

with $G_i \equiv G(t_n + c_i h)$. By [55], systems with matrices of general size require a $\lambda-$dependent step size restriction of the form $h \leq O\left(|\lambda|^{-1/4}\right)$ in order to be defined.

E3: The exponential of the matrix $\Omega^{[p]}$ has to be computed. This will be the most expensive step of the method. For this, Matlab function `expm()` will be used which provides the value for machine accuracy.

The algorithm then provides an approximation for $Y(x_{n+1})$ starting from $Y_n \approx Y(x_n)$, with $x_{n+1} = x_n + h$.

In the present paper, Ω is truncated using a sixth-order Magnus series and the integrals are approximated using three point Gaussian integration method [53]. $\Omega^{[6]}$ is obtained in the following equations. Let,

$$P_1 = hG(c_1 h), \quad P_2 = hG(c_2 h), \quad P_3 = hG(c_3 h),$$

$$Q_1 = P_2, \quad Q_2 = \frac{\sqrt{15}}{3}(P_3 - P_1), \quad Q_3 = \frac{10}{3}(P_3 - 2P_2 + P_1),$$

$$R_1 = [Q_1, Q_2], \quad R_2 = [Q_1, 2Q_3 + R_1], \quad R_3 = [-20Q_1 - Q_3 + R_1, Q_2 - \frac{1}{60}R_2].$$

Then,

$$\Omega^{[6]}(h) = Q_1 + \frac{1}{12}Q_3 + \frac{1}{240}R_3,$$

where $c_1 = \frac{1}{2} - \frac{\sqrt{15}}{10}$, $c_2 = \frac{1}{2}$ and $c_3 = \frac{1}{2} + \frac{\sqrt{15}}{10}$ are the nodes of Gaussian quadrature.

3.3 Solutions of direct Sturm–Liouville problems

In this Section, some numerical examples of Sturm–Liouville problems of different orders are presented. Furthermore, the efficiency and accuracy of the proposed technique outlined in the previous section is investigated, numerically.

Here, we made use of the MATLAB 2014 package to code the algorithms: multi-section method (Listing B.1) and Magnus method (Listing B.2) for solving SLP. The codes executed on an Intel ® Core ™ i3 CPU with power 2.40 GHz, equipped with 8 GB RAM.

For the numerical results (unless otherwise specified), the parameter values are: $m = 100$, $n = 500$, $L = 5$ where n is the number of subdivisions in the interval $[a, b]$ and m is the number of subdivisions in the interval $[\lambda_0, \lambda^*]$, λ^* being the maximum eigenvalue searching, and L is the number of multisection steps used to calculate each eigenvalue in the characteristic function.

The performance of the Magnus method is measured by the absolute error E_k which is defined as

$$E_k = \left|\lambda_k^{(Exact)} - \lambda_k^{(Magnus)}\right|, \qquad k = 1, 2, \ldots \tag{3.9}$$

where $\lambda_k^{(Magnus)}$ indicates the kth eigenvalue obtained by Magnus method and $\lambda_k^{(Exact)}$ is the kth exact eigenvalue and the relative error ϵ_k which is defined as

$$\epsilon_k = \left|\frac{\lambda_k^{(Exact)} - \lambda_k^{(Magnus)}}{\lambda_k^{(Exact)}}\right|, \qquad k = 1, 2, \ldots. \tag{3.10}$$

There are several parameters in the Magnus method, which affect the performance and the accuracy of the method, namely:

1. eigenvalue index: k or eigenvalue: λ_k
2. number of subdivisions in λ: m
3. number of subdivisions in x: n

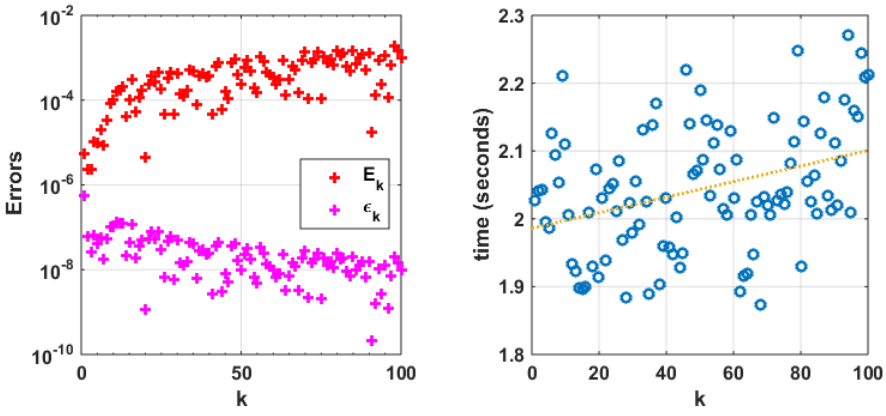
4. number of multisection iterations: L

The performance is measured by the computation time used, here the Matlab function `timeit()` will be used. `timeit` measures time required to run a function by calling the specified function multiple times, and computing the median of the measurements. It provides a robust measurement of the time required for function execution and the function provides a more vigorous estimate [77]. The accuracy is measured using the absolute and relative errors as defined in Equations (3.9) and (3.10), respectively. The Example 3.1 illustrates the affect of these variables on a SLP.

▶ **Example 3.1.** This eigenvalue problem is due to Chawla and Katti [32],

$$-y'' = \lambda y, \quad 0 < x < 1, \qquad y(0) = y(1) = 0, \tag{3.11}$$

having the exact eigenvalues $\lambda_k = (k\pi)^2$, $k = 1, 2, \ldots$. From Figure 3.1(a), the

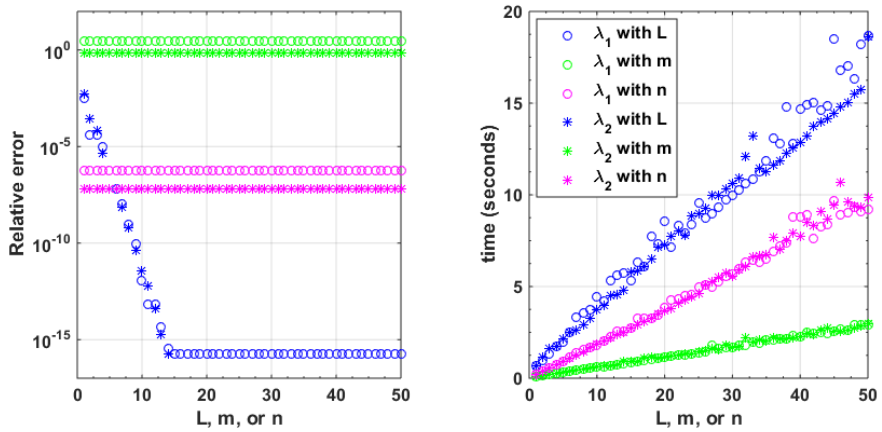


(a) (b)

Figure 3.1: (**a**) Log absolute error (red) and log relative error (magenta) and (**b**) computing time (in seconds) (blue circles) and linear trend line (orange dashed line) for the first 100 eigenvalues using Magnus method for the problem (3.11).

absolute error is increasing (as the eigenvalues are increasing quadratically), however, the relative error is (independent of λ) gradually decreasing as $\lambda \to \infty$. This shows the Magnus method's capability to find higher index eigenvalues with less relative error. (See Table A.1 in Appendix A for a detailed analysis of

the code). Also the computation time is very slowly increasing linearly with the index of the eigenvalue (Figure 3.1(b)). As expected accuracy and computation time increase with $m-$ the number of subdivisions of λ and $L-$ the number of multisection iteration steps (Figure 3.2). After around $L = 15$, the error doesn't improve, so it is sufficient to stop at 15 iterations. As Figure 3.2(b) shows the



(**a**) (**b**)

Figure 3.2: (**a**) Relative error and (**b**) computation time (in seconds) for the first two eigenvalues λ_1 (∘) and λ_2 (*) of the problem (3.11) using Magnus method: for $m = 10$, $L = 5$, and $n = 1, \ldots, 50$ (magenta); for $n = 10$, $m = 10$, and $L = 1, \ldots, 50$ (blue); for $n = 10$, $L = 1$, and $m = 1, \ldots, 50$ (green).

computation time increases with the number of subdivisions of x: $n-$, but there is no change in the error (Figure 3.2(a)). This is obvious, since in Equation (3.11) all coefficient functions are constant, hence G is independent of x and the number of subdivisions n. ◀

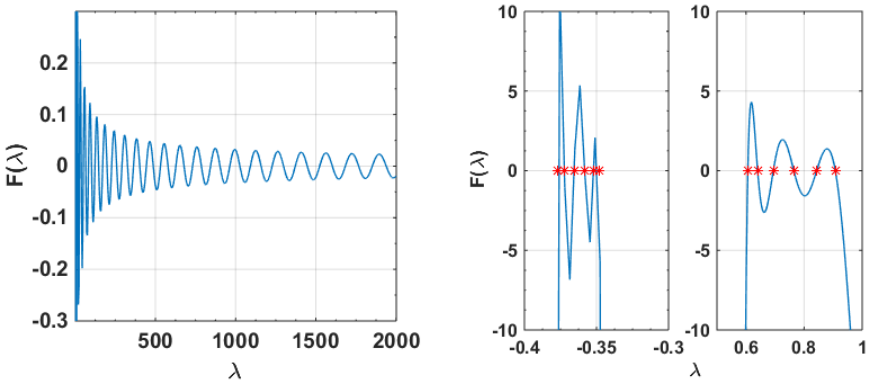
▶ **Example 3.2 (SL Regular problem).** This is an almost singular eigenvalue problem due to Paine et al. [91].

$$-y'' + (x + 0.1)^{-2} y = \lambda y, \quad 0 < x < \pi, \qquad y(0) = y(\pi) = 0. \tag{3.12}$$

Tables 3.1 and 3.2 list the first four eigenvalues of this problem and compare the relative errors with FDM [91, Table 4], NM [11, Table 3], FEM [10, Table

Table 3.1: First four eigenvalues of (3.12).

k	Eigenvalue	
	Exact [110, Table 4]	Magnus
1	1.5198658	1.519865820634
2	4.9433098	4.943309819778
3	10.2846630	10.284662639575
4	17.5599580	17.559957737891



(**a**) (**b**)

Figure 3.3: (**a**) Characteristic function of (3.12). (**b**) Parts of the characteristic equation (blue) and first 12 eigenvalues (red) for the version of Mathieu equation (3.13).

Table 3.2: Relative errors for the first four eigenvalues of (3.12). [Minimum relative error for each eigenvalue is in bold]

k	Relative error							
	Magnus	FDM [91]	NM [11]	FEM [10]	MNM [110]	FEMT [19]	FEMS [19]	LGM [84]
1	$1.21E-08$	$\mathbf{0.00E+00}$	$2.63E-06$	$1.91E-04$	$3.95E-06$	$1.32E-04$	$1.41E-03$	$2.74E-11$
2	$4.00E-09$	$8.09E-05$	$4.86E-06$	$3.03E-04$	$8.50E-06$	$2.16E-04$	$2.02E-04$	$\mathbf{2.27E-11}$
3	$3.50E-08$	$8.75E-05$	$7.10E-06$	$3.51E-04$	$8.75E-06$	$2.78E-04$	$2.37E-04$	$\mathbf{5.26E-11}$
4	$1.49E-08$	$9.68E-05$	$9.23E-06$	$3.59E-04$	$1.73E-05$	$3.17E-04$	$2.55E-04$	$\mathbf{1.15E-09}$

3], MNM [110, Table 5], FEMT and FEMS [19, Table 2], and LGM [84, Table 4] methods. Magnus method exhibits good relative errors compared with the other methods.

Figure 3.3(a) shows the characteristic function for this problem. It can be seen that the roots (eigenvalues) are simple and also the eigenvalues are further apart as the index increases. ◀

▶ **Example 3.3 (Singular SLP).** Magnus method is robust that it can handle even singular Sturm-Liouville Problems. For example, consider the Bessel equation of order $\frac{1}{2}$ [95, Test problem 19]:

$$xy'' + \frac{1}{4x}y = \lambda xy, \quad 0 < x < 1, \qquad y(1) = 0.$$

Here $p(0) = 0$ and the exact eigenvalues are $\lambda_k = (k\pi)^2$. The endpoint $a = 0$ is singular and a limit-circle non-oscillatory (LCN) point (Note that the assumptions A3 and A4 mentioned in section 3.2.1 no longer holds). Taking Friedrich's BCs: $f = 1$, $g = \ln x$, writing the boundary condition coefficients as:

$$\begin{pmatrix} 1 & \log(0) \\ 1 & \log(1) \end{pmatrix}\begin{pmatrix} y(0) & y(1) \\ y'(0) & y'(1) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

the eigenvalues can be easily obtained as: $\lambda_1 = 9.864557319999999$, $\lambda_2 = 39.458190680000001$, $\lambda_3 = 88.780786309999996$, and so on. ◀

▶ **Example 3.4.** This example – a version of Mathieu equation is also due to [95, Test problem 5].

$$y'' + (\cos x)y = \lambda y, \quad 0 < x < 40, \qquad y(0) = y(40) = 0. \tag{3.13}$$

Here the lower eigenvalues form clusters of 6 (see Figure 3.3(b)). Using a sufficiently small step size (< 0.004: the minimum difference between two consecutive eigenvalues) for the subdivision in $\lambda-$dimension, Magnus method is able to find even the lower eigenvalues with an accuracy comparable to Matslise's (see Table 3.3). ◀

▶ **Example 3.5 (FSLP: Simple BCs).** This example is due to [97, Example 1] :

$$y^{(4)} + y = \lambda y, \qquad y(0) = y''(0) = y(1) = y''(1) = 0. \tag{3.14}$$

Table 3.3: First 17 eigenvalues and errors for the version of Mathieu equation (3.13) using Matslise and Magnus method.

k	Matslise [61]	Magnus	Absolute difference
1	−0.376845881566946	−0.37684593940	$5.78330540124128E-08$
2	−0.372222021386702	−0.37222207135	$4.99632979988895E-08$
3	−0.365517698755253	−0.36551773700	$3.82447470359537E-08$
4	−0.358145409533808	−0.35814543445	$2.49161919985141E-08$
5	−0.351818307563677	−0.35181832065	$1.30863230252132E-08$
6	−0.348153086495489	−0.34815309260	$6.10451100779841E-09$
7	0.606260772683456	0.60626073600	$3.66834560505680E-08$
8	0.639995069435221	0.63999503360	$3.58352210128032E-08$
9	0.694009291118601	0.69400929280	$1.68139902001485E-09$
10	0.764487943746816	0.76448803840	$9.46531840684273E-08$
11	0.843278584575029	0.84327879680	$2.12224970930208E-07$
12	0.907400354525007	0.90740065280	$2.98274993038028E-07$
13	1.272925107989470	1.27292467200	$4.35989469860232E-07$
14	1.381819492267290	1.38181949440	$2.13271000859550E-09$
15	1.525973491208340	1.52597360640	$1.15191659988412E-07$
16	1.695868669968620	1.69586882560	$1.55631380005516E-07$
17	1.884251375610910	1.88425164800	$2.72389089950309E-07$

By Schueller [101], exact eigenvalues are: $\lambda_k = (k\pi)^4 + 1, k = 1, 2, \ldots$.

Table 3.4: The first five eigenvalues for (3.14).

k	Eigenvalue	
	Exact	Magnus
1	$9.84090910340024E+01$	$9.84090910340024E+01$
2	$1.55954545654404E+03$	$1.55954545654403E+03$
3	$7.89113637375420E+03$	$7.89113637375442E+03$
4	$2.49377273047046E+04$	$2.49377273046367E+04$
5	$6.08816818962515E+04$	$6.08816818943245E+04$

Table 3.5: The relative errors of the first five eigenvalues for (3.14). Relative errors for all the methods except Magnus method are calculated from the eigenvalues reported in [97, Table 5]. * denotes the methods with correction terms. [Minimum relative error for each eigenvalue is in bold]

k	Relative error					
	Magnus	FDM*	MNM*	BVM6*	BVM8*	BVM10*
1	**1.44E − 16**	$2.38E-09$	$5.11E-09$	$9.73E-08$	$4.71E-09$	$4.14E-08$
2	**5.54E − 15**	$1.88E-10$	$1.49E-09$	$1.04E-08$	$2.57E-11$	$1.52E-09$
3	**2.84E − 14**	$1.22E-10$	$4.43E-10$	$1.40E-09$	$1.10E-11$	$2.47E-10$
4	**2.72E − 12**	$2.31E-11$	$5.16E-11$	$7.11E-10$	$2.60E-11$	$6.07E-11$
5	$3.17E-11$	$3.44E-11$	$2.45E-11$	$1.96E-10$	**4.73E − 12**	$3.15E-11$

From Tables 3.4, 3.5 and Figure 3.4 it is clear that Magnus method is more accurate. However, unlike the other methods, the accuracy decreases with the index of the eigenvalue due to the increasing magnitude of the eigenvalue. In contrast, for the second-order SLP using Magnus method the relative error decreased with the index (Figure 3.1 **(a)**).

◀

▶ **Example 3.6 (FSLP: General BCs).** This example involving more general boundary conditions was taken from [97, Example 4] to illustrate the Magnus method's versatility.

$$y^{(4)} + (\sin(x) + 2)y = \lambda y, \tag{3.15}$$

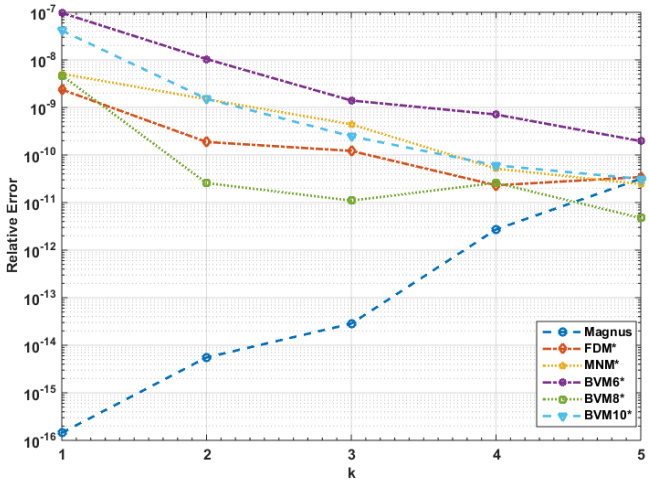


Figure 3.4: Log relative error of the first five eigenvalues of problem (3.14)

$$y(0) - y'(0) = 0, \qquad y(0) - y''(0) = 0,$$
$$y(1) - y'(1) = 0, \qquad y(1) - y''(1) = 0.$$

According to [97, pp.153], the exact eigenvalues of this problem could not be computed and there is no other method except matrix methods (MM) proposed by them to approximate the eigenvalues. Table 3.6 lists the first five eigenvalues

Table 3.6: First five eigenvalues of FSLP (3.15). * Eigenvalues (except for the Magnus method) were computed using the codes provided by F. Fischer, PhD student at the Potsdam University, Germany. For each eigenvalue computed using Magnus method, the closest ones by other methods are in bold.

k	Magnus	FDM*	MNM*	BVM6*	BVM8*	BVM10*
1	3.52240175682026	3.44479	2.60411	**3.52238**	**3.52238**	**3.52238**
2	503.00125581467400	502.56365	501.77704	502.96155	**502.96324**	502.96270
3	3805.98521478000000	3804.50674	3804.41400	3805.47622	**3805.49752**	3805.49069
4	14620.08332703000000	14614.34574	**14619.49377**	14617.34897	14617.46132	14617.42540
5	39946.25457020200000	39923.30006	**39950.76042**	39936.65422	39937.04152	39936.91814

of the problem and it can be seen the values by Magnus method are comparable

with values computed by matrix methods. From Figure 3.4, we see that the BVM8 method with correction terms performs better than the other methods considered (except Magnus) in that example. And from Table 3.6, it is clear that Magnus method's values are closer to the BVM8's values for this example. ◀

▶ **Example 3.7.** This example illustrates Magnus method's superiority over some of the other existing methods. This problem describes the free lateral vibration of a uniform clamped-hinged beam (CH) [70]:

$$y^{(4)} = \lambda y, \qquad y(0) = y'(0) = y(1) = y''(1) = 0. \tag{3.16}$$

From [70, Equation 45], the eigenvalues are the solutions of the equation:

$$\tanh \sqrt[4]{\lambda} = \tan \sqrt[4]{\lambda}. \tag{3.17}$$

Table 3.7: The first five eigenvalues of the problem (3.16)

k	Eigenvalue	
	Exact	Magnus
1	237.72106753	237.72106753
2	2496.48743786	2496.48743786
3	10867.58221698	10867.58221697
4	31780.09645408	31780.09645380
5	74000.84934916	74000.84934040

As Tables 3.7 and 3.8 point out, Magnus method has a comparable relative error to the other methods considered (CSCM [70, Table 4], ADM [13], HAM [1], PDQ and FDQ [112, Table 1], CM [44, Tables 1, 2], CDM [105, Table 2], VIM [104, Table 2], and SPSS [56]) with respect to the first five eigenvalues. ◀

More examples on Magnus methods for FSLP can be found in Mirzaei [83].

▶ **Example 3.8 (SSLP).**

$$-y^{(6)} = \lambda y, \quad 0 < x < \pi, \quad y(0) = y''(0) = y^{(4)}(0) = y(\pi) = y''(\pi) = y^{(4)}(\pi) = 0. \tag{3.18}$$

Table 3.8: Relative error of the first five eigenvalues for the problem (3.16) [Minimum relative error for each eigenvalue is in bold].

k	Relative error									
	Magnus	CSCM [70]	ADM [13]	HAM [1]	PDQ [112]	FDQ [112]	CM [44]	CDM [105]	VIM [104]	SPSS [56]
1	$4.6E-12$	$2.4E-6$	$4.7E-12$	$\mathbf{0.0E+00}$	$7.6E-9$	$3.2E-9$	$4.7E-12$	$2.0E-09$	$6.7E-11$	$4.6E-12$
2	$1.4E-12$	$9.8E-6$	$4.4E-12$	$4.0E-12$	$4.4E-8$	$4.8E-8$	$3.0E-12$	$7.9E-10$	$4.0E-10$	$\mathbf{1.2E-12}$
3	$1.2E-12$	$9.3E-6$	$1.0E-06$	$\mathbf{9.2E-13}$	$1.7E-8$	$2.2E-8$	$5.1E-12$	$2.3E-10$	$1.8E-09$	$2.1E-12$
4	$8.8E-12$	$9.3E-6$	$9.6E-03$	$\mathbf{6.0E-12}$	$2.4E-8$	$1.7E-8$	$8.6E-09$	$2.0E-11$	$2.0E-09$	$4.2E-11$
5	$1.2E-10$	$9.3E-6$	-	$\mathbf{3.5E-11}$	$3.0E-8$	$1.9E-8$	-	$7.5E-11$	$3.5E-09$	$2.3E-10$

This example is due to [50, Problem 1], having exact eigenvalues: $\lambda_k = k^6$, $k = 1, 2, \ldots$. Again from Table 3.9, Magnus method shows excellent accuracy with regard to this sixth-order SLP compared to other methods: Shooting method [50, Table 1], ADM [62, Table 2], CP [5, Table 2], VIM [104, Table 3]. ◀

As far as the authors concern there exists no algorithms for solving eighth-order SLP in the literature. However, proposed Magnus method can be successfully applied with literally no modification to the algorithm. Only the matrix G has to be modified using Equation (3.18).

▶ **Example 3.9 (Eighth-order Sturm Liouville problem).** Squaring the fourth-order SLP:

$$y^{(4)} = \lambda y, \qquad y(0) = y''(0) = y(1) = y''(1) = 0,$$

with exact eigenvalues $\lambda_k = (k\pi)^4, k = 1, 2, \ldots$, the following eighth-order SLP is obtained:

$$y^{(8)} = \lambda^2 y, \qquad y(0) = y''(0) = y^{(4)}(0) = y^{(6)}(0) = y(1) = y''(1) = y^{(4)}(1) = y^{(6)}(1) = 0, \tag{3.19}$$

with exact eigenvalues $\lambda_k^2 = (k\pi)^8, k = 1, 2, \ldots$. The first five eigenvalues have good accuracy although the error is steadily increasing with the index of the eigenvalue (see Table 3.10). This is because the error also depends on the magnitude of the eigenvalue. Specifically, the error in a pth-order method grows as $O(h^{p+1}\lambda^{p/2-1})$, and thus one expects poor approximations for large eigenvalues [20]. ◀

▶ **Example 3.10 (Singular SLP: infinite domain).** This example extends the Magnus method, as described in [92] to infinite domain. Consider the harmonic oscillator problem

$$u'' + (\lambda - x^2)u = 0, \quad x \in (-\infty, \infty) \tag{3.20}$$

together with Dirichlet and Neumann BCs, having exact eigenvalues $\lambda_k = 2k + 1$, $n = 0, 1, \ldots$ [106].

The domain is truncted to $[-l, l]$, and using the parameters $m = 10$, $n = 100$, $L = 10$ the first three eigenvalues are computed (see Table 3.11). It can be

Table 3.9: Eigenvalues and relative error for the problem (3.18) [Minimum relative error for each eigenvalue is in bold].

k	Eigenvalue		Relative error				
	Exact	Magnus	Magnus	Shooting [50]	ADM [62]	CP [5]	VIM [104]
1	1	0.999999999999844	$1.56E-13$	$2.00E-07$	**0.00E + 00**	**0.00E + 00**	$5.50E-09$
2	64	64.000000000001400	$2.20E-14$	$1.72E-07$	**0.00E + 00**	**0.00E + 00**	$7.32E-09$
3	729	728.999999999632000	$5.05E-13$	-	$0.00E+00$	**0.00E + 00**	$5.88E-09$
4	4096	4095.999999988370000	$2.84E-12$	$1.71E-07$	**0.00E + 00**	**0.00E + 00**	$1.61E-08$
5	15625	15624.999999968700000	$2.00E-12$	-	$1.28E-13$	**6.40E − 15**	$2.82E-09$
6	46656	46655.999979784000000	$4.33E-10$	$1.71E-07$	$1.00E-08$	**2.18E − 15**	$4.62E-09$
7	117649	117648.998941287000000	$9.00E-09$	-	$1.34E-04$	**6.50E − 11**	$2.94E-08$
8	262144	262143.988377187000000	$4.43E-08$	$1.53E-07$	$1.24E-01$	-	**9.50E − 09**
9	531441	531440.822457814000000	$3.34E-07$	-	-	-	**1.25E − 07**
10	1000000	1000000.004000010000000	**4.00E − 09**	$1.70E-07$			

Table 3.10: Eigenvalues and relative errors for the problem (3.19) using $m = 100$, $n = 100$, $L = 10$

k	Eigenvalue		Relative error
	Exact	Magnus	
1	$9.48853101607057E+03$	$9.48853101607059E+03$	$1.91704005653253E-15$
2	$2.42906394011407E+06$	$2.42906394011394E+06$	$5.21434895376847E-14$
3	$6.22542519964390E+07$	$6.22542519964604E+07$	$3.43361538321867E-13$
4	$6.21840368669201E+08$	$6.21840368667400E+08$	$2.89626411740934E-12$
5	$3.70645742815257E+09$	$3.70645742800000E+09$	$4.11625897732872E-11$

deduced that the Magnus method has comparable performance to other method. Moreover, fixed parameters can be changed to get better accuracy. ◀

Table 3.11: Absolute errors for the first three eigenvalues of Equation (3.20) from [106, Table 1] and using Magnus method (Minimum absolute error for each eigenvalue is shows in bold.)

λ	l	**Dirichlet error**		**Neumann error**	
		[106]	Magnus method	[106]	Magnus method
1	4.5	$1.70E-08$	$\mathbf{1.58E-08}$	$\mathbf{1.50E-08}$	$1.68E-08$
	6.0	$\mathbf{2.60E-11}$	$5.80E-10$	$\mathbf{2.00E-12}$	$5.80E-10$
	7.5	$\mathbf{2.00E-14}$	$2.20E-09$	$\mathbf{0.00E+00}$	$2.20E-09$
5	4.5	$\mathbf{1.88E-06}$	$1.10E-05$	$\mathbf{1.10E-05}$	$1.19E-05$
	6.0	$3.30E-08$	$\mathbf{8.02E-09}$	$\mathbf{2.73E-09}$	$8.00E-09$
	7.5	$\mathbf{5.50E-11}$	$3.07E-08$	$\mathbf{6.00E-12}$	$3.07E-08$
9	4.5	$9.99E-01$	$\mathbf{9.85E-04}$	$\mathbf{9.85E-04}$	$1.12E-03$
	6.0	$6.01E-06$	$\mathbf{3.06E-08}$	$6.11E-07$	$\mathbf{2.54E-08}$
	7.5	$\mathbf{1.70E-08}$	$1.08E-07$	$\mathbf{2.00E-09}$	$1.08E-07$

▶ **Example 3.11 (Singular SLP with asymmetric potential).** Consider the Morse potential problem

$$-u'' - (e^{-\alpha x} - 1)^2 u = \lambda u, \quad x \in (-\infty, \infty) \tag{3.21}$$

together with Dirichlet and Neumann BCs. When $\lambda \in (0,1)$ this has a finite number of EV: $\lambda_k = \beta_k(2-\beta_k)$, with $\beta_k = \frac{1}{2}\alpha(2k+1)$, $k = 0,1,\ldots,K$, where $K = \left\lfloor \frac{1}{\alpha} - \frac{1}{2} \right\rfloor$ [106].

The domain is truncated to $[-l,l]$, and using the parameters $m = 10$, $n = 100$, $L = 10$ and $\alpha = 0.02$ the first three eigenvalues are computed (see Table 3.12). Magnus method is more accurate when the truncated interval is small. Also it can be observed that initial eigenvalues are more accurate than the later ones.

Table 3.12: Absolute errors for the first three eigenvalues of Equation (3.21) from [106, Table 2] and using Magnus method (Minimum absolute error for each eigenvalue is shows in bold.)

λ	l	Dirichlet error		Neumann error	
		[106]	Magnus method	[106]	Magnus method
0.0199	40	$1.10E-08$	**5.60E − 12**	$1.90E-09$	**1.12E − 11**
	50	$6.07E-10$	**3.17E − 11**	$6.00E-11$	**3.17E − 11**
	60	**1.15E − 13**	$9.39E-11$	**2.48E − 14**	$9.39E-11$
0.0591	40	$3.56E-07$	**3.78E − 10**	$3.90E-08$	**3.34E − 10**
	50	$2.20E-08$	**1.12E − 10**	$5.97E-09$	**1.12E − 10**
	60	**5.00E − 12**	$3.37E-10$	**6.75E − 13**	$3.37E-10$
0.0975	40	$7.69E-07$	**1.85E − 08**	$1.13E-07$	**1.93E − 08**
	50	$6.65E-07$	**3.83E − 10**	$1.29E-07$	**3.82E − 10**
	60	**2.50E − 11**	$1.15E-09$	**3.18E − 12**	$1.15E-09$

◀

► **Example 3.12 (Singular SLP in half axis).** Consider the hydrogen atom equation

$$-u'' - \left(\frac{2}{x^2} - \frac{1}{x}\right)u = \lambda u, \quad x \in (0,\infty) \tag{3.22}$$

together with Dirichlet and Neumann BCs. This has finite number of eigenvalues lying between $-1 < \lambda < 0$, given by $\lambda_k = -\dfrac{1}{4(k+2)^2}$, with $k = 0,1,\ldots,$ [74].

The domain is truncated to $[0,100]$, and using the parameters $m = 100$, $n = 200$, $L = 10$ the first few eigenvalues are computed (see Table 3.13).

Table 3.13: Eigenvalues and absolute errors for the first five eigenvalues of Equation (3.22) using Magnus method.

k	Magnus value	Magnus error
1	$-6.2323550470257E-02$	$1.76E-04$
2	$-2.7717584125111E-02$	$6.02E-05$
3	$-1.5581393052633E-02$	$4.36E-05$
4	$-9.0660619438460E-03$	$9.34E-04$
5	$-1.6285043066920E-03$	$5.32E-03$

3.4 Discussion

It is clear that Magnus method has the potential to find the first few eigenvalues of a SLP of arbitrary (even) order with good accuracy.

Also, it is observed for the EVPs with constant coefficients (hence constant and mostly sparse G matrix) the errors are smaller ($n = 2$: Example 3.1, $n = 4$: Examples 3.5, 3.7, $n = 6$: Example 3.8 and $n = 8$: Example 3.9) irrespective of the order. However, when G is no longer a constant, the error is not as good even for the SLP of order 2 (see Example 3.2). This can be attributed to the matrix exponent calculation step and can be improved by using a more accurate method in place of `expm()` function.

Furthermore, it is capable of solving some singular SLPs on infinite domain, half–axis and for an asymmetric potential. In addition, the present method doesn't require transforming the singular SLP to a regular one or re-scaling the eigenvalues. It effectively finds the required eigenvalues, just by truncating the interval to a finite one.

4 Solutions of Inverse Sturm–Liouville Problems

This chapter is based on two of own publications [92] and [93].

4.1 Introduction

The inverse Sturm–Liouville theory was originated in 1929 by [6] and further developed in [57, 63, 64, 73, 107].

The $2m$th order, nonsingular, self-adjoint eigenvalue problem (EVP) is given by:

$$(-1)^m(p_0(x)y^{(m)})^{(m)} + (-1)^{m-1}(p_1(x)y^{(m-1)})^{(m-1)} + \ldots + (p_{m-2}(x)y'')'' - (p_{m-1}(x)y')' + p_m(x)y = \lambda w(x)y, \qquad a < x < b$$

together with some boundary conditions at a and b, the functions p_k, $(0 \leq k \leq m)$, and $w(x)$ being continuous on the finite closed interval $[a, b]$, and p_0 having a continuous derivative. In the inverse Sturm–Liouville problem (SLP), the coefficient functions p_k, $(0 \leq k \leq m)$ need to be reconstructed, given suitable valid spectral data.

For a discussion on numerical methods for inverse SLP refer [42]. [80] provides an overview of analytical methods for second and fourth-order inverse problems.

Iterative methods [16, 100], Rayleigh–Ritz method [51], finite difference approximation [90], Quasi-Newton method [99], shooting method [68], interval Newton's method [89], finite-difference method [41], boundary value methods [3, 4, 47, 48], Numerov's method [7, 8, 9], least-squares functional [98], generalized Rundell—Sacks algorithm [40, 96], spectral mappings [52], Lie-group estimation method [67], Broyden method [22, 23], decent flow methods [45], modified Numerov's method [46], Newton-type method [36], Fourier–Legendre series [58], and Chebyshev polynomials [88] are of particular importance among the existing methods to solve inverse SLP.

Numerical algorithms to solve the inverse fourth–order Sturm–Liouville problem (FSLP) are proposed in [17], and [78, 79].

The inverse problem is much harder as the restrictions on the spectral data should be placed to ensure the uniqueness. Barcilon [18] proved that in general, $m+1$ spectra associated with $m+1$ distinct 'admissible' boundary conditions are required to determine p_m's uniquely (with $p_m = w = 1$). Later McLaughlin and Rundell [81] proved that the measurement of a particular eigenvalue for an infinite set of different boundary conditions is sufficient to determine the unknown potential. Both these theorems require an infinite number of accurate eigenvalue measurements, which are hard to obtain in practice as the higher eigenvalues are usually more expensive to compute than lower ones [15]. However, the present technique can be applied to address some of the above difficulties in finding suitable set(s) of eigenvalues so that the inverse problem can be effectively solved.

Section 4.2 explains the inverse SLP algorithm and Section 4.3 presents some solutions of inverse SLPs.

4.2 Inverse SLP Algorithm for the general order

The Magnus method's ability to handle various types of boundary conditions, adaptability to extend to higher order problems, and accuracy with regard to higher index eigenvalues naturally allows it to be useful in generating suitable spectra for constructing and testing different inverse SLPs.

The Iterative Solution presented by Barcilon [16] was selected as the inverse SLP algorithm due to two reasons: first, there was no evidence of actual implementation of Barcilon's algorithm in the literature. Second, other numerical methods available for the inverse problem are not amenable to generalization for higher order equations or systems. Setting $p_0(x), w(x) \equiv 1$ in (3.1) and taking the domain $[0,1]$ the EVP reads as:

$$(y^{(m)})^{(m)} - (p_1 y^{(m-1)})^{(m-1)} + \ldots + p_m(x)y = \lambda y, \qquad 0 < x < 1\,. \tag{4.1}$$

According to [18], $m+1$ distinct spectra are required in order to determine $p_1(x), p_2(x), \ldots, p_m(x)$. In other words, $m+1$ different EVPs with distinct boundary conditions (BC) are needed. In addition, the spectra need to be interlaced, i.e.,

$$\ldots < \lambda_{n-1}^{(m+1)} < \lambda_n^{(1)} < \lambda_n^{(2)} < \lambda_n^{(3)} < \ldots < \lambda_n^{(m+1)} < \lambda_{n+1}^{(1)} < \ldots$$

Assume that m boundary conditions are common to all the given $m+1$ eigenvalue problems, say, conditions at the right end point $x = 1$.

1. Combine the $m+1$ eigenvalue problems into an equivalent linear differential equation:
   a) Define a vector function $\boldsymbol{\phi}(x, \lambda)$:
   $$\begin{aligned}\boldsymbol{\phi}^T(x,\lambda) &= [\varpi\chi, \varpi'\chi', \ldots, \varpi^{(m-1)}\chi^{(m-1)}] \\ &= [\phi_{(1)}, \phi_{(2)}, \ldots, \phi_{(m)}]\end{aligned}$$
   where $\varpi(x, \lambda)$ and $\chi(x, \lambda)$ are two solutions of equation (4.1).
   b) Differentiate each $\phi_{(k)}$, $2m$ times.
   c) Obtain $m(2m+1)$ linear equations for $\varpi^{(i)}\chi^{(j)} + \varpi^{(j)}\chi^{(i)}$, where $i \le j$, $j = 0, 1, \ldots, 2m-1$.
   d) Express $\varpi^{(i)}\chi^{(j)} + \varpi^{(j)}\chi^{(i)}$, $i \le j$, $j = 0, 1, \ldots, 2m-1$, in terms of $\phi_{(1)}, \phi_{(2)}, \ldots, \phi_{(m)}$ and their first $2m$ derivatives.
   e) Differentiate $\phi_{(i)}^{(2m)}$, $i = 1, 2, \ldots, m$ again, to obtain m linear coupled differential equations of order $2m+1$, i.e.,
   $$\mathcal{M}\boldsymbol{\phi} = \lambda\mathcal{N}\boldsymbol{\phi} \tag{4.2}$$
2. Obtain the corresponding boundary conditions for the linear differential equation using the boundary conditions of the $m+1$ eigenvalue problems:
   a) Assume that $\varpi(x, \lambda)$ and $\chi(x, \lambda)$ satisfy the m boundary conditions common to all the given $m+1$ eigenvalue problems.
   b) Then from 1.3, find the $m(3m+1)/2$ boundary conditions at $x = 1$, and the $m(m+1)/2$ boundary conditions at $x = 0$ for $\boldsymbol{\phi}(x, \lambda)$.
3. Solve equation (4.2) using above boundary conditions, to obtain the solutions $\boldsymbol{\phi}_n$.
4. Solve the adjoint system of equations to the above system, and denote the solutions by $\boldsymbol{\eta}_n$.
5. Find the bi-orthogonal set of functions $\{\boldsymbol{y}_n\}_1^\infty$ to $\{\boldsymbol{\phi}_n\}_1^\infty$, using the relation $\boldsymbol{y}_n(x) = \mathcal{N}^T\boldsymbol{\eta}_n(x)$.

With these pre-calculated values, Algorithm 1 can be used to solve the inverse SLP.

Algorithm 1: Solving inverse SLP.

Data: $m+1$ distinct spectra $\left\{\lambda_n^{(i)}\right\}_{n=1}^{\infty}$, $i = 1, 2, \ldots, m+1$ corresponding to $m+1$ different EVPs.

Result: the unknown potential functions
$\mathbf{p}^T(x) = [p_m(x), p_{m-1}(x), \ldots, p_1(x)]$

1 Coalesce the $m+1$ sequences of eigenvalues into a single sequence $\{\hat{\sigma}_n\}_1^{\infty}$ using

$$\hat{\sigma}_{(i-1)(m+1)+j} = \lambda_i^{(j)}, \quad j = 1, 2, \ldots, m+1, \qquad i = 1, 2, \ldots \tag{4.3}$$

Set: initial guess $\mathbf{p}^{(0)}$, $k = 1$;

2 Solve: EVPs using $\mathbf{p}^{(0)}$ and combine the solutions to $\left\{\sigma_n^{(0)}\right\}$;

3 **while** $\sum_n \left|\hat{\sigma}_n - \sigma_n^{(k-1)}\right| \neq 0$ **do**

4 Set

$$\hat{\mathbf{y}}_n(x) = \frac{\left\langle \mathbf{e}^T \boldsymbol{\phi}_n \right\rangle}{\left\langle \mathbf{y}_n^T \boldsymbol{\phi}_n \right\rangle} \mathbf{y}_n \tag{4.4}$$

and

$$\mathbf{p}^{(k)}(x) = \mathbf{p}^{(k-1)}(x) + \sum_{n=1}^{\infty} \left(\hat{\sigma}_n - \sigma_n^{(k-1)}\right) \hat{\mathbf{y}}_n(x) \tag{4.5}$$

Solve the EVPs using $\mathbf{p}^{(k)}$ and set the solutions to $\left\{\sigma_n^{(k)}\right\}$;

5 $k = k + 1$;

Some remarks on Algorithm 1 follows:

Line (Data:) Infinite eigenvalue (EV) sequence is replaced with the first N EVs. The truncated EV sequences $\left\{\lambda_n^{(i)}\right\}_{n=1}^{N}$, $i = 1, 2, \ldots, m+1$ are obtained, using the Magnus method [92] to solve Equation (4.1).

Line (Result:) Output $\mathbf{p}$: $m \times n$ matrix of function values, n: the number of subdivisions in the x–axis and m: the number of potentials.

Line (1) We take initial guess $\mathbf{p}^{(0)}$ as $\mathbf{0}_{m\times n}$.

Line (3) `while loop` exit condition is replaced with a maximum number of iterations: $kMAX$ and in each iteration the condition $\sum_{n=1}^{(m+1)N} \left|\hat{\sigma}_n - \sigma_n^{(k-1)}\right| < tol$ (tol: a fixed tolerance) is checked to exit the loop.

Line (4) $\langle\cdot\rangle$ denotes the integration w.r.t. x from 0 to 1 and $\mathbf{e}^T = [1, 0]$. $\mathbf{y}_n$ and $\boldsymbol{\phi}_n$ are kept fixed at $\mathbf{y}_0$ and $\boldsymbol{\phi}_0$, which are the solutions by setting $p_i = 0, \forall i$. Solutions $\mathbf{y}_0$ and $\boldsymbol{\phi}_0$ are calculated using Mathematica [75]. Matlab [76] built-in functions: `trapz()` and `griddedInterpolant()` with `pchip` (piecewise cubic Hermite interpolating polynomial) option are used to approximate integrals and to have p as a function, respectively. Latter is required since, p as a function is input to the Magnus method. Again, the eigenvalue sequence $\sigma^{(k)}$ is calculated using $\mathbf{p}^{(k)}$ and the iteration repeats.

4.2.1 Inverse SLP Algorithm of order 2

Consider the Sturm-Liouville problem

$$y'' + (\sigma - q(x))y = 0, \quad x \in (-1, 1) \qquad y(-1) = y(1) = 0, \tag{4.6}$$

for a symmetric and normalized potential $q(x)$, i.e.

$$q(-x) = q(x) \qquad \text{and} \qquad \int_{-1}^{1} q(x)dx = 0.$$

Then the spectrum $\{\sigma_n\}_1^\infty$ uniquely determines the symmetric potential $q(x)$ [24]. The eigenvalue problem (4.6) is equivalent to the following pair of Sturm-Liouville problems:

$$u'' + (\lambda - q(x))u = 0, \quad x \in (0, 1) \qquad u(0) = u(1) = 0, \tag{4.7}$$
$$v'' + (\mu - q(x))v = 0, \quad x \in (0, 1) \qquad v'(0) = v(1) = 0. \tag{4.8}$$

Also the two spectra $\{\lambda_n\}_1^\infty$, $\{\mu_n\}_1^\infty$, are interlaced, i.e.,

$$0 < \mu_1 < \lambda_1 < \mu_2 < \lambda_2 < \mu_3 < \dots. \tag{4.9}$$

Letting,

$$\begin{aligned} w_{2n}(x) &= u_n^2(x), \quad \nu_{2n} = 4\lambda_n \\ w_{2n-1}(x) &= v_n^2(x), \quad \nu_{2n-1} = 4\mu_n \end{aligned} \tag{4.10}$$

the equations (4.7) and (4.8) are combined into:

$$w''' - 4qw' - 2q'w = -\nu w', \qquad w'(0) = w(1) = w'(1) = 0. \tag{4.11}$$

Equation (4.11) is not self-adjoint and its adjoint is given by:

$$\omega''' - 4q\omega' - 2q'\omega = -\nu\omega', \qquad \omega(0) = \omega''(0) = \omega(1) = 0. \tag{4.12}$$

Then the inverse SLP procedure presented in Algorithm 2 (which is a simplified version of Algorithm 1) can be used to recover the unknown potential q.

For a symmetric potential a fixed set of eigenfunctions can be used in the updating formula (4.13) for $\hat{w}_n(x)$, for example, eigenfunctions for the case $q \equiv 0$ [16]. This is possible because,

> "...if $q \in L^2([a,b])$, the kth eigenvalue $\lambda_k(q,a,b)$ behaves asymptotically as
>
> $$\lambda_k(q,a,b) = \lambda_k(0,a,b) + \overline{q} + \delta_k(q,a,b), \tag{4.15}$$
>
> where $\lambda_k(0,a,b)$ is the kth eigenvalue of the SLP with the same BCs and zero potential, $\overline{q}$ is the mean value of q and $\delta_k(q,a,b)$ is the remainder for smooth potential. Moreover, the information that the given spectrum provides about the variation of the unknown potential are contained in the terms $\delta_k(q,a,b)$ and, in view of their behaviour, the first eigenvalues are the most important for the reconstruction of q...." [3, p. 2]

This implies that the required set can be restricted to the first few eigenvalues. In fact, when the number of input eigenvalues increases, the error also increases, which can be attributed to the approximation errors in the higher index eigenvalues (see Figure 4.3).

Implementation details of the steps of Algorithm 2 are as follows:

Algorithm 2: Solving inverse SLP

Data: Sequence $\{\hat{\nu}_n\}_1^\infty$, the combined eigenvalues of the two equations (4.7) and (4.8) using equation (4.10)

Result: the unknown potential function $q(x)$

1 Set an initial guess $q^{(0)}$, $k = 1$;

2 Solve EVPs (4.11), (4.12), and get the eigensolutions: $\left\{\nu_n^{(0)}, w_n^{(0)}(x), \omega_n^{(0)}(x)\right\}_1^\infty$;

3 **while** $\hat{\nu}_n \neq \nu_n^{(k-1)}$, $n = 1, 2, \ldots$ **do**

4 Set

$$\hat{w}_n(x) = \frac{\displaystyle\int_0^1 w_n^{(k-1)}(\xi)d\xi}{\displaystyle\int_0^1 w_n^{(k-1)}(\xi)\left(\frac{d\omega_n^{(k-1)}}{d\xi}\right)d\xi}\frac{d\omega_n^{(k-1)}}{dx} \tag{4.13}$$

and

$$q^{(k)}(x) = q^{(k-1)}(x) + \frac{1}{4}\sum_{n=1}^{\infty}\left(\hat{\nu}_n - \nu_n^{(k-1)}\right)\hat{w}_n(x) \tag{4.14}$$

Solve EVPs (4.11), (4.12), and get the eigensolutions: $\left\{\nu_n^{(k)}, w_n^{(k)}(x), \omega_n^{(k)}(x)\right\}_1^\infty$;

5 $k = k + 1$;

Line (Data:) Input eigenvalue sequence is limited to $N = 10$ to get a finite sequence. Using Magnus method on the two equations (4.7) and (4.8) the truncated eigenvalue sequences $\{\lambda_n\}_1^N$, $\{\mu_n\}_1^N$ are obtained, then using (4.10) are combined to obtain the interlaced set of eigenvalues $\{\hat{\nu}_n\}_1^{2N}$.

Line (Result:) The output is a finite vector of function values: $q_0, q_1, \ldots, q_n$, where n is the number of subdivisions in the $x-$axis.

Line (1) Initial guess $q^{(0)}$ will be set to $q \equiv 0$.

Line (2) By setting $q = 0$ in the equations (4.7) and (4.8), the following expressions are easily obtained:

- $\left\{\lambda_n^{(0)}\right\}_1^{\infty} = \left\{(n\pi)^2\right\}_1^{\infty}$,
- $\left\{\mu_n^{(0)}\right\}_1^{\infty} = \left\{\left((n-\frac{1}{2})\pi\right)^2\right\}_1^{\infty}$,
- $\left\{v_n^{(0)}\right\}_1^{\infty} = \left\{\lambda_n^{(0)}\right\}_1^{\infty} \bigcup \left\{\mu_n^{(0)}\right\}_1^{\infty}$,
- $\left\{u_n^{(0)}\right\}_1^{\infty} = \{\sin(n\pi x)\}_1^{\infty}$,
- $\left\{v_n^{(0)}\right\}_1^{\infty} = \left\{\cos\left(n-\frac{1}{2}\right)\pi x\right\}_1^{\infty}$,
- $\left\{w_n^{(0)}\right\}_1^{\infty} = \left\{\cos\left(\sqrt{v_n^{(0)}}\pi x\right)\right\}_1^{\infty}$,
- $\left\{\omega_n^{(0)}\right\}_1^{\infty} = \left\{\sin\left(\sqrt{v_n^{(0)}}\pi x\right)\right\}_1^{\infty}$.

For the rest of the algorithm, the eigenfunctions $\left\{w_n^{(k)}(x), \omega_n^{(k)}(x)\right\}_1^{2N}$ are kept fixed at $\left\{w_n^{(0)}(x), \omega_n^{(0)}(x)\right\}_1^{2N}$.

Line (3) The optimal while loop condition: $\hat{v}_n \neq v_n^{(k-1)}$, $n = 1, 2, \ldots$ may never reach due to various errors in the numerical procedure, and is relaxed to $\sum_{n=1}^{2N}\left|\hat{v}_n - v_n^{(k-1)}\right| < tol$, and/or $k < kMAX$ where tol is a fixed tolerance and $kMAX$ is a maximum number of iterations.

Line (4) The required derivatives and integrals in equation (4.13) are approximated using the Matlab built-in functions `gradient()` and `trapz()`, respectively. Furthermore, the Matlab built-in function `griddedInterpolant()` with `linear` interpolation method is used to obtain an explicit functional form of q from the set of points $q_0, q_1, \ldots, q_n$. This is necessary because in the next step, the explicit form of q should be the input to the Magnus method (which allows evaluating $q(x)$ at an arbitrary point). Just one of the EVPs (4.11) or (4.12), is need to be solved using Magnus method since only the eigenvalue sequence $\left\{v_n^{(k)}\right\}_1^{2N}$ is needed (as the eigenfunctions are kept fixed) in the subsequent iterations.

4.2.2 Inverse FSLP algorithm

Setting $m = 2$ in equation (4.1) we get FSLP:

$$\mathcal{L}u \equiv u^{(4)} - (pu')' + qu = \lambda u. \tag{4.16}$$

Three spectra $\{\lambda_n\}$, $\{\mu_n\}$ and $\{\nu_n\}$ required to reconstruct the unknown coefficients: p and q will be obtained from the following EVPs:

$$\mathcal{L}u_n = \lambda_n u_n, \qquad u_n(0) = u_n''(0) = u_n(1) = u_n''(1) = 0, \tag{4.17}$$

$$\mathcal{L}v_n = \mu_n v_n, \qquad v_n(0) = v_n'(0) = v_n(1) = v_n''(1) = 0, \tag{4.18}$$

$$\mathcal{L}w_n = \nu_n w_n, \qquad w_n'(0) = w_n''(0) = w_n(1) = w_n''(1) = 0. \tag{4.19}$$

Although, equations (4.17) and (4.18) are self-adjoint, equation (4.19) is not. It's adjoint equation is:

$$\mathcal{L}\omega_n = \nu_n \omega_n, \qquad \omega_n'(0) = \omega_n'''(0) - p(0)\omega_n'(0) = \omega_n(1) = \omega_n''(1) = 0. \tag{4.20}$$

The eigenvalues (for $p = q = 0$) λ_n, μ_n and ν_n are given by the solutions of the following equations:

$$\begin{aligned} \sin(s) = 0, &\qquad s^4 = \lambda \\ \tan(r) - \tanh(r) = 0, &\qquad r^4 = \mu \\ \tan(t) + \tanh(t) = 0, &\qquad t^4 = \nu \end{aligned}$$

Furthermore,

$$t \approx \left(n - \frac{1}{4}\right)\pi, \qquad s = n\pi, \qquad r \approx \left(n + \frac{1}{4}\right)\pi, \qquad n \in \mathbb{N}$$

so that the spectra $\{\lambda_n\}$, $\{\mu_n\}$ and $\{\nu_n\}$ are interlaced, i.e.,

$$0 < \nu_1 < \lambda_1 < \mu_1 < \nu_2 < \lambda_2 < \mu_2 < \dots$$

Since the eigenvalues are interlaced, they can be coalesced into a single one by defining

$$\left\{\sigma_k, \begin{bmatrix} \phi_k \\ \psi_k \end{bmatrix}\right\}_1^\infty = \begin{cases} \left\{\nu_n, \begin{bmatrix} w_n\omega_n \\ w_n'\omega_n' \end{bmatrix}\right\}_1^\infty & \text{for } k = 3n-2, \\ \left\{\lambda_n, \begin{bmatrix} u_n^2 \\ u_n'^2 \end{bmatrix}\right\}_1^\infty & \text{for } k = 3n-1, \qquad n = 1, 2, \ldots . \\ \left\{\mu_n, \begin{bmatrix} v_n^2 \\ v_n'^2 \end{bmatrix}\right\}_1^\infty & \text{for } k = 3n, \end{cases} \tag{4.21}$$

Denote:

$$\mathbf{q} = \begin{bmatrix} q \\ p \end{bmatrix}, \qquad \boldsymbol{\phi}_n = \begin{bmatrix} \phi_n \\ \psi_n \end{bmatrix}.$$

Let $\chi(x,\lambda)$ and $\varpi(x,\lambda)$ be two solutions of (4.16) satisfying the common boundary conditions of equations (4.17)-(4.20), namely, ($u(1) = u''(1) = 0$, etc.)

$$\chi(1,\lambda) = \varpi(1,\lambda) = \chi''(1,\lambda) = \varpi''(1,\lambda) = 0\,. \tag{4.22}$$

Guided by (4.21), define

$$\begin{cases} \phi(x,\lambda) = \varpi(x,\lambda)\,\chi(x,\lambda) \\ \psi(x,\lambda) = \varpi'(x,\lambda)\,\chi'(x,\lambda)\,. \end{cases}$$

By differentiating ϕ and ψ five times each, and eliminating ϖ, and χ, 9th order linear differential equation for ϕ (or a coupled system of linear differential equations of order 5 for ϕ and ψ) can be obtained. When $p = q = 0$, this reduces to

$$\phi^{(9)}(x) - 12\lambda\phi^{(5)}(x) = 64\lambda^2\phi^{(1)}(x) \tag{4.23}$$

or

$$\begin{pmatrix} 3D^5 & -10D^3 \\ 0 & D^5 \end{pmatrix}\begin{pmatrix} \phi \\ \psi \end{pmatrix} = \lambda\begin{pmatrix} 8D & 0 \\ 10D^3 & -24D \end{pmatrix}\begin{pmatrix} \phi \\ \psi \end{pmatrix} \tag{4.24}$$

in the form of the equation (4.2), and the boundary conditions reads as:

$$\phi = \phi''' = \psi' = 0 \quad \text{at} \quad x = 0, \tag{4.25}$$

$$\phi = \phi' = \phi'' - 2\psi = \phi''' = \phi'''' - 4\psi'' = \psi' = \psi''' = 0 \quad \text{at} \quad x = 1\,. \tag{4.26}$$

The solution of the system of equations (4.24)-(4.26) using Mathematica is:

$$\begin{aligned}
&\phi(x) \\
&= \frac{\csc(\mu)}{8(2\cosh(\mu)+\cosh(2\mu)(-3\cos(\mu)+\cos(3\mu)-\cos((2+i)\mu)+\cos((1+2i)\mu)+\cosh((2+i)\mu)-\cosh((1+2i)\mu)))} \\
&\times (i\cos((1+i)((1+i)x+1)\mu) - i\cos((1+i)((1+i)x\mu+\mu)) + 16\sin(2\mu) - 4\sin(4\mu) - (10+10i)\sin((1+i)\mu) \\
&+ (2-2i)\sin((3+i)\mu) - (4+4i)\sin((2+2i)\mu) - (2-2i)\sin((1+3i)\mu) + (2+2i)\sin((3+3i)\mu) \\
&- 2i\sin((1+i)((3+i)-x)\mu) - (6+6i)\sin((1+i)(x-2)\mu) + (2+2i)\sin((1+i)(x+2)\mu) \\
&+ 6i\sin((1+i)((-1-i)+x)\mu) + 2\sin((1+i)((-3+i)+x)\mu) + 4\sin((1+i)(-2i+x)\mu) + 4i\sin((1+i)(2i+x)\mu) \\
&- (2+4i)\sin((1+i)((1+i)+x)\mu) - 2i\sin((1+i)((-1+3i)+x)\mu) + (2-2i)\sin((1+i)((-2-2i)+x)\mu) \\
&- 2\sin((1+i)((-1-3i)+x)\mu) + 2i\sin(((1+i)+2x)\mu) - 2i\sin(((1+3i)+2x)\mu) - 6i\sin((2i+(1-i)x)\mu) \\
&+ (2+2i)\sin((4i+(1-i)x)\mu) - 2\sin(((2+4i)+(1-i)x)\mu) + 2\sin(2(ix+1)\mu) + 4i\sin((1+i)(ix+2)\mu) \\
&+ 8\sin(((1-i)+2ix)\mu) - 2\sin(((3+i)+2ix)\mu) + 2\sin(((1-3i)+2ix)\mu) - 2\sin(((3-3i)+2ix)\mu) \\
&+ \sin((1+i)((1+i)x+1)\mu) - (4+2i)\sin(((1+i)x+2)\mu) + 8i\sin(((1+i)-2x)\mu) + 2i\sin(((3+i)-2x)\mu) \\
&- 2i\sin(((3+3i)-2x)\mu) - 6\sin((2-(1-i)x)\mu) + (2+2i)\sin((4-(1-i)x)\mu) + 2\sin(2(1-ix)\mu) \\
&+ 4\sin((1+i)(2-ix)\mu) - 8\sin(2((1+i)-ix)\mu) + 2\sin(2((2+i)-ix)\mu) + 2\sin(2((1+2i)-ix)\mu) \\
&+ 2\sin(((1-i)-2ix)\mu) - 2\sin(((3-i)-2ix)\mu) + 8\sin(((1+i)-2ix)\mu) + 2\sin(((1+3i)-2ix)\mu) \\
&- 2\sin(((3+3i)-2ix)\mu) - 6\sin((2-(1+i)x)\mu) + (2-2i)\sin((4-(1+i)x)\mu) - (4-2i)\sin((1-i)x\mu \\
&+ 2\mu) + \sin((1+i)((1+i)x\mu+\mu)) - 8i\sin((2+2i)\mu-2x\mu) + 2i\sin((4+2i)\mu-2x\mu) \\
&+ 2i\sin((2+4i)\mu-2x\mu) - 16\sinh(2\mu) + 4\sinh(4\mu) + (10+10i)\sinh((1+i)\mu) \\
&- (2-2i)\sinh((3+i)\mu) + (4+4i)\sinh((2+2i)\mu) + (2-2i)\sinh((1+3i)\mu) - (2+2i)\sinh((3+3i)\mu) \\
&+ (6+6i)\sinh((1+i)(x-2)\mu) - (2+2i)\sinh((1+i)(x+2)\mu) - 2i\sinh((1+i)((-3-i)+x)\mu) + 2i\sinh(2((-2-i)+x)\mu) \\
&- 8i\sinh(2((-1-i)+x)\mu) - 2\sinh((1+i)((-3+i)+x)\mu) + 2i\sinh(2((-1-2i)+x)\mu) + 2\sinh((1+i)((-1-3i)+x)\mu) \\
&- 2\sinh(2(ix+1)\mu) + 2\sinh(((3+i)+2ix)\mu) - 2\sinh((1+i)((1+i)x+1)\mu) + (4+2i)\sinh(((1+i)x+2)\mu) \\
&- 2\sinh(2(1-ix)\mu) + 8\sinh(2((1+i)-ix)\mu) - 2\sinh(2((2+i)-ix)\mu) - 2\sinh(2((1+2i)-ix)\mu) \\
&- 8\sinh(((1+i)-2ix)\mu) - 2\sinh(((1+3i)-2ix)\mu) + 2\sinh(((3+3i)-2ix)\mu))
\end{aligned}$$

$$\begin{aligned}
&\psi(x) \\
&= \frac{\mu^2\csc(\mu)}{8(2\cosh(\mu)+\cosh(2\mu)(-3\cos(\mu)+\cos(3\mu)-\cos((2+i)\mu)+\cos((1+2i)\mu)+\cosh((2+i)\mu)-\cosh((1+2i)\mu)))} \\
&\times (i\cos((1+i)((1+i)x+1)\mu) - i\cos((1+i)((1+i)x\mu+\mu)) - 16\sin(2\mu) + 4\sin(4\mu) + (10-10i)\sin((1+i)\mu) \\
&- (2+2i)\sin((3+i)\mu) + (4-4i)\sin((2+2i)\mu) + (2+2i)\sin((1+3i)\mu) - (2-2i)\sin((3+3i)\mu) \\
&- 2\sin((1+i)((3+i)-x)\mu) - (6-6i)\sin((1+i)(x-2)\mu) + (2-2i)\sin((1+i)(x+2)\mu) + 6\sin((1+i)((-1-i)+x)\mu) \\
&- 2i\sin((1+i)((-3+i)+x)\mu) - 4i\sin((1+i)(-2i+x)\mu) + 4\sin((1+i)(2i+x)\mu) - (4-2i)\sin((1+i)((1+i)+x)\mu) \\
&- 2\sin((1+i)((-1+3i)+x)\mu) - (2+2i)\sin((1+i)((-2-2i)+x)\mu) + 2i\sin((1+i)((-1-3i)+x)\mu) - 2i\sin(((1+i)+2x)\mu) \\
&+ 2i\sin(((1+3i)+2x)\mu) + 6\sin((2i+(1-i)x)\mu) - (2-2i)\sin((4i+(1-i)x)\mu) - 2i\sin(((2+4i)+(1-i)x)\mu)
\end{aligned}$$

$$
\begin{aligned}
&+2\sin(2(ix+1)\mu)-4\sin((1+i)(ix+2)\mu)+8\sin(((1-i)+2ix)\mu)-2\sin(((3+i)+2ix)\mu)+2\sin(((1-3i)+2ix)\mu)\\
&-2\sin(((3-3i)+2ix)\mu)+\sin((1+i)((1+i)x+1)\mu)-(2-4i)\sin(((1+i)x+2)\mu)-8i\sin(((1+i)-2x)\mu)\\
&-2i\sin(((3+i)-2x)\mu)+2i\sin(((3+3i)-2x)\mu)-6i\sin((2-(1-i)x)\mu)-(2-2i)\sin((4-(1-i)x)\mu)+2\sin(2(1-ix)\mu)\\
&+4i\sin((1+i)(2-ix)\mu)-8\sin(2((1+i)-ix)\mu)+2\sin(2((2+i)-ix)\mu)+2\sin(2((1+2i)-ix)\mu)+2\sin(((1-i)-2ix)\mu)\\
&-2\sin(((3-i)-2ix)\mu)+8\sin(((1+i)-2ix)\mu)+2\sin(((1+3i)-2ix)\mu)-2\sin(((3+3i)-2ix)\mu)+6i\sin((2-(1+i)x)\mu)\\
&-(2+2i)\sin((4-(1+i)x)\mu)-(2+4i)\sin((1-i)x\mu+2\mu)+\sin((1+i)((1+i)x\mu+\mu))+8i\sin((2+2i)\mu-2x\mu)\\
&-2i\sin((4+2i)\mu-2x\mu)-2i\sin((2+4i)\mu-2x\mu)-16\sinh(2\mu)+4\sinh(4\mu)+(10-10i)\sinh((1+i)\mu)\\
&-(2+2i)\sinh((3+i)\mu)+(4-4i)\sinh((2+2i)\mu)+(2+2i)\sinh((1+3i)\mu)-(2-2i)\sinh((3+3i)\mu)-(6-6i)\sinh((1+i)(x-2)\mu)\\
&+(2-2i)\sinh((1+i)(x+2)\mu)+2\sinh((1+i)((-3-i)+x)\mu)+2i\sinh(2((-2-i)+x)\mu)-8i\sinh(2((-1-i)+x)\mu)\\
&-2i\sinh((1+i)((-3+i)+x)\mu)+2i\sinh(2((-1-2i)+x)\mu)+2i\sinh((1+i)((-1-3i)+x)\mu)+2\sinh(2(ix+1)\mu)\\
&-2\sinh(((3+i)+2ix)\mu)+2\sinh((1+i)((1+i)x+1)\mu)-(2-4i)\sinh(((1+i)x+2)\mu)+2\sinh(2(1-ix)\mu)\\
&-8\sinh(2((1+i)-ix)\mu)+2\sinh(2((2+i)-ix)\mu)+2\sinh(2((1+2i)-ix)\mu)+8\sinh(((1+i)-2ix)\mu)+2\sinh(((1+3i)-2ix)\mu)\\
&-2\sinh(((3+3i)-2ix)\mu))
\end{aligned}
$$

with $\mu^4=\lambda$. The adjoint of system of equations (4.24)-(4.26) is

$$
\begin{pmatrix} 3D^5 & 0 \\ -10D^3 & D^5 \end{pmatrix}\begin{pmatrix} \eta \\ \zeta \end{pmatrix}=\lambda\begin{pmatrix} 8D & 10D^3 \\ 0 & -24D \end{pmatrix}\begin{pmatrix} \eta \\ \zeta \end{pmatrix} \tag{4.27}
$$

with

$$
\eta=\eta''=\eta'''=\zeta=\zeta'=\zeta''=\zeta''''=0 \quad \text{at} \quad x=0, \tag{4.28}
$$

$$
\zeta=\zeta''+2\eta=\zeta''''-4\eta''=0 \quad \text{at} \quad x=1\,. \tag{4.29}
$$

The solutions are

$$
\begin{aligned}
&\eta(x)\\
&=\cos(2\mu x)+2\cos((1+i)\mu x)+\cosh(2\mu x)+2\cosh((1+i)\mu x)\\
&\quad+\frac{(2-2i)(\sin(\mu)-\sinh(\mu))(\sin(\mu)\sinh(2\mu)-\sin(2\mu)\sinh(\mu))}{(\sin(2\mu)+\sinh(2\mu))(\sin(\mu)\cosh(\mu)-\cos(\mu)\sinh(\mu))}(\sin((1+i)\mu x)+\sinh((1+i)\mu x))\\
&\quad-\frac{(\cos(2\mu)+\cosh(2\mu)-2)}{\sin(2\mu)+\sinh(2\mu)}(\sin(2\mu x)+\sinh(2\mu x))-6
\end{aligned}
$$

$$
\begin{aligned}
&\zeta(x)\\
&=\frac{1}{\mu^2}\Bigg(-\cos(2\mu x)-2i\cos((1+i)\mu x)+\cosh(2\mu x)+2i\cosh((1+i)\mu x)\\
&\quad+\frac{(2+2i)(\sin(\mu)-\sinh(\mu))(\sin(\mu)\sinh(2\mu)-\sin(2\mu)\sinh(\mu))}{(\sin(2\mu)+\sinh(2\mu))(\sin(\mu)\cosh(\mu)-\cos(\mu)\sinh(\mu))}(\sinh((1+i)\mu x)-\sin((1+i)\mu x))\\
&\quad+\frac{(\cos(2\mu)+\cosh(2\mu)-2)}{\sin(2\mu)+\sinh(2\mu)}(\sin(2\mu x)-\sinh(2\mu x))\Bigg).
\end{aligned}
$$

The bi-orthogonal set for $\{\boldsymbol{\phi}\}$ given by $\mathbf{y} = \mathcal{N}^T\boldsymbol{\eta}$ is:

$$
\begin{aligned}
y(x) = {} & \frac{24(1+i)\mu}{(\sin(2\mu)+\sinh(2\mu))(\sin(\mu)\cosh(\mu)-\cos(\mu)\sinh(\mu))} \\
& \times \big(-(2-2i)\sin^2(\mu)\sinh(2\mu)\cos(\mu x)\cosh(\mu x) \\
& + \cos(\mu)\sinh(\mu)(-\sin(2\mu)(\sin((1+i)\mu x)-(2-2i)\sin(2\mu x))+(2-2i)(\cos(2\mu)-2)\cos(2\mu x) \\
& + \sin(2\mu)(\sinh((1+i)\mu x)-(2-2i)\sinh(2\mu x))+\sinh(2\mu)((2-2i)\sin(2\mu x)-\sin((1+i)\mu x) \\
& -(2-2i)\sinh(2\mu x)+\sinh((1+i)\mu x))+(2-2i)\sin^2(\mu)\cos((1+i)\mu x) \\
& +(2-2i)(\cos(2\mu)-2)\cosh(2\mu x)+(2-2i)\cosh(2\mu)(\cos(2\mu x) \\
& + \cosh(2\mu x))+(2-2i)\sin^2(\mu)\cosh((1+i)\mu x)\big) \\
& + \sin(\mu)\cosh(\mu)(\sin(2\mu)(\sin((1+i)\mu x)-(2-2i)\sin(2\mu x))+(-2+2i)(\cos(2\mu)-2)\cos(2\mu x) \\
& + \sinh(2\mu)(-(2-2i)\sin(2\mu x)+\sin((1+i)\mu x)+(2-2i)\sinh(2\mu x)-\sinh((1+i)\mu x)) \\
& - \sin(2\mu)(\sinh((1+i)\mu x)-(2-2i)\sinh(2\mu x))-(2-2i)(\cos(2\mu)-2)\cosh(2\mu x) \\
& -(2-2i)\cosh(2\mu)(\cos(2\mu x)+\cosh(2\mu x))+(4-4i)\sinh^2(\mu)\cos(\mu x)\cosh(\mu x))\big) \\
& +(-2+2i)\sin(2\mu)\sinh^2(\mu)\cos(\mu x)\cosh(\mu x))\big)
\end{aligned}
$$

$$
\begin{aligned}
z(x) = {} & -\frac{48}{\mu}\Bigg(\sin(2\mu x)+(-1+i)\sin((1+i)\mu x)+\sinh(2\mu x)-(1-i)\sinh((1+i)\mu x) \\
& +\frac{(\cos(2\mu)+\cosh(2\mu)-2)}{\sin(2\mu)+\sinh(2\mu)}(\cos(2\mu x)-\cosh(2\mu x)) \\
& +\frac{2i(\sin(\mu)-\sinh(\mu))(\sin(\mu)\sinh(2\mu)-\sin(2\mu)\sinh(\mu))}{(\sin(2\mu)+\sinh(2\mu))(\sin(\mu)\cosh(\mu)-\cos(\mu)\sinh(\mu))}(\cosh((1+i)\mu x)-\cos((1+i)\mu x))\Bigg).
\end{aligned}
$$

4.3 Results of Inverse Sturm–Liouville problems

Here we present some numerical examples of direct and inverse SLPs of orders 2 and 4.

Algorithm 2 was implemented using MATLAB (2014) [76]. The reference solutions of the EVPs are computed using Wolfram Mathematica 11 [75].

For the numerical calculations, n is the number of subdivisions in the interval $[a,b]$, m is the number of subdivisions in the interval $[\lambda_0,\lambda^*]$; λ^* being the maximum eigenvalue searching, L is the number of multisection steps used to calculate each eigenvalue in the characteristic function and M is the number of inverse algorithm steps.

Error values are stated using max-norm which is defined as $||\mathbf{x}||_\infty := \max_n(|x_n|)$.

4.3.1 Inverse SLP of order 2

▶ **Example 4.1 (Symmetric potential).** Consider the EVP

$$u'' + (\lambda - q(x))u = 0, \quad x \in (0, 1)$$

Suppose $q(x) = \cos(\pi x)$, which is symmetric and normalized. The truncated eigenvalue sequences at $N = 10$, for Dirichlet boundary conditions (DDBCs): $u(0) = u(1) = 0$ and Dirichlet- Neumann boundary conditions (DNBCs): $v'(0) = v(1) = 0$ are calculated using Magnus method (using a tolerance $1e - 13$), and combined into $\{\hat{\nu}_n\}_1^{2N}$. This will be the input to the algorithm. Now using an intial guess, say $q = 0$, the eigenvalue sequences for DDBCs and DNBCs are calculated again using Magnus method and the sequence $\left\{\nu_n^{(0)}\right\}_1^{2N}$ is constructed. Then q is updated using the equations, (4.5) and (4.4). Again this cycle repeats. Iteration stops when the desired accuracy is reached. Figures 4.1 and 4.2

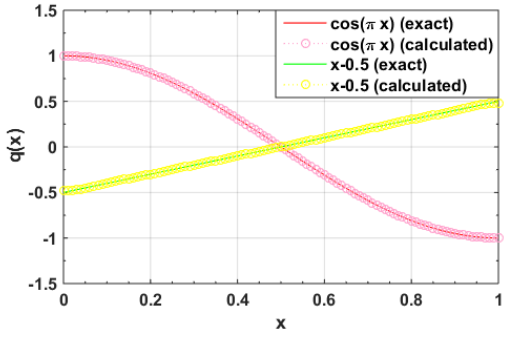
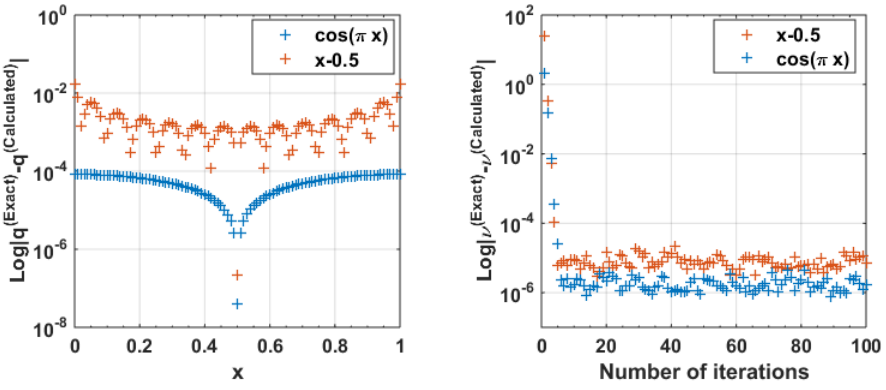


Figure 4.1: Exact and reconstructed potentials $q(x) = \cos(\pi x)$ and $q(x) = x - 0.5$.

show the exact, and reconstructed potential q and the log absolute errors in the reconstructed potential and the eigenvalue sequences, respectively. The supremum norm difference between the reconstructed q and the actual q is $8.229454025521221e - 05$. It is clear that the error is larger at the end-points than in the middle. In the mid-point $q(x)$ is 0 and from equation (4.15), this implies a minimum error, see Figure 4.2(a). The supremum norm difference between the reconstructed eigenvalues and the actual ones is $1.302601049246732e - 07$. Also after about 15 iterations the differences in the two sets of eigenvalues are



(a) (b)

Figure 4.2: (a) Log absolute error in the reconstructed q, and **(b)** log absolute error sum of the difference of the eigenvalue sequences $\{\hat{\nu}_n\}$ and $\{\nu_n^{(k-1)}\}$ for reconstructing the potentials $q(x) = \cos(\pi x)$ (blue) and $q(x) = x - 0.5$ (red).

not improving, just oscillating, implying that more iterations may try to overfit (Figure 4.2(b)).

When the number of eigenvalues increase the error also increases, due to the errors in approximating higher index eigenvalues (see Figure 4.3). This verifies Aceto et al. [3]'s claim that the first eigenvalues are the most important for the reconstruction of q, so the required set should be restricted to the first few eigenvalues. Also, after about 6 iterations, the error does not improves, and even increases for some values of N. So it's advisable to stop the iteration procedure by checking the error. ◀

► **Example 4.2 (Non-symmetric potential).** For a second example, consider $q(x) = x - 0.5$, which is non-symmetric but normalized. Figures 4.1 and 4.2 show the exact, and reconstructed potential q and the log absolute errors in the reconstructed potential and the reconstructed eigenvalue sequence, respectively. The supremum norm difference between the reconstructed q and the actual q is 0.016647548176497. Although not as good as for the symmetric potential this can be improved if the exact eigenvalues for the exact potential (here we are calculating them using the Magnus method) are known. The supremum norm difference between the reconstructed eigenvalues sequence and the actual ones

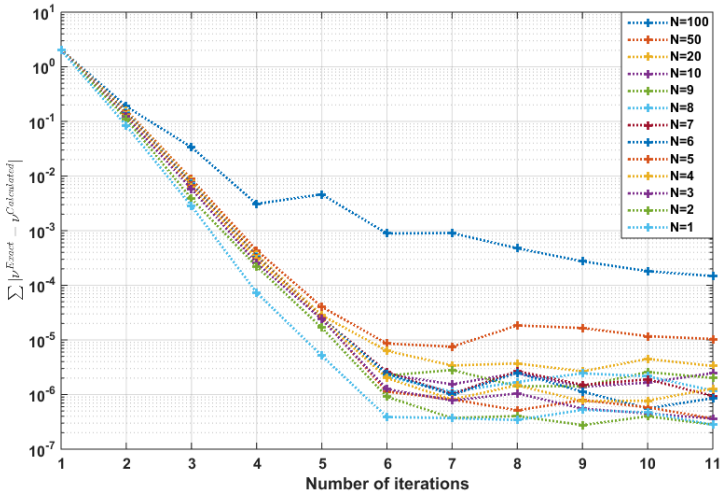


Figure 4.3: Sum of the absolute errors in the reconstructed eigenvalues: $\sum |v^{Exact} - v^{Calculated}|$ using Magnus method vs. the iteration number, using eigenvalue sets of sizes $N = 1, \ldots, 10, 20, 50, 100$ for reconstructing the potential $q(x) = \cos(\pi x)$.

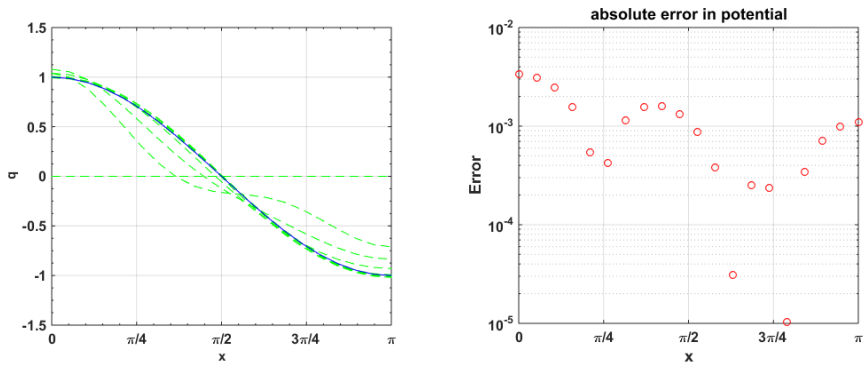
is 0.004796140075996. Similar to previous example, the error is larger at the end-points than in the middle (Figure 4.2(a)). ◀

▶ **Example 4.3 (Inverse SLP in a different domain).** This example extends Barcilon's algorithm in [17] by changing the domain $[0, 1]$ to $[0, \pi]$:

$$u'' + (\lambda - \cos(x))u = 0, \quad x \in (0, \pi)$$

Here $p(x)$ is symmetric and normalized. In Algorithm 2, $w_n^{(0)}(x)$ and $\omega_n^{(0)}(x)$ need to be changed into $\left\{w_n^{(0)}\right\}_1^\infty = \{\cos(nx) - \cos(n\pi)\}_1^\infty$, $\left\{\omega_n^{(0)}\right\}_1^\infty = \{\sin(nx)\}_1^\infty$ which are the basic solutions in the new domain. The truncated eigenvalue sequences at $N = 6$, for Dirichlet boundary conditions (DDBCs) and Dirichlet–Neumann boundary conditions (DNBCs) are calculated using the Magnus method (with $m = 20$, $n = 20$, $L = 15$) and $M = 10$ inverse algorithm steps are used.

Figure 4.4 shows the reconstructed and exact potential, and the log absolute errors in the reconstructed potential, respectively. From Figure 4.4(a), it is obvious that the potential is converging towards the exact one. The max-norm



(a) (b)

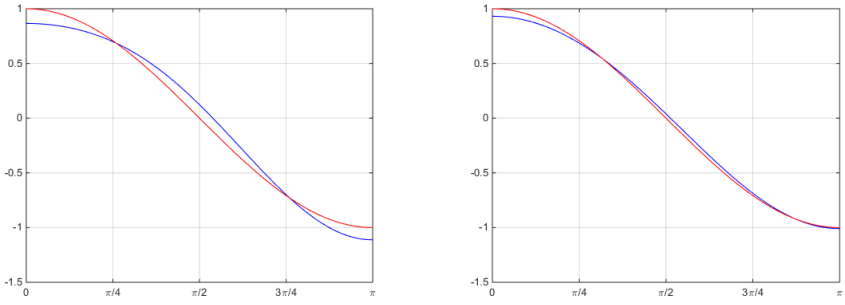
Figure 4.4: (**a**) Iteratively reconstructed potential (green dashed line) and exact potential (blue solid line) $p(x) = \cos(x)$ (**b**) Log absolute error in the reconstructed p.

of difference between the reconstructed p and the actual p is $\approx 3.35 \times 10^{-3}$ and the max-norm of difference between the reconstructed eigenvalues and the actual ones is $\approx 1.62 \times 10^{-3}$.

Figure 4.5 shows reconstructed p starting with perturbed eigenvalues

$$\sigma_k^\delta = \sigma_k + \delta \cdot \sigma_k \cdot \text{rand}(\text{size}(\sigma_k)), \quad k = 1, \ldots, 2N$$

where δ is the noise level and $N = 5$, $m = 20$, $n = 100$, $L = 5$, $M = 6$.



(a) (b)

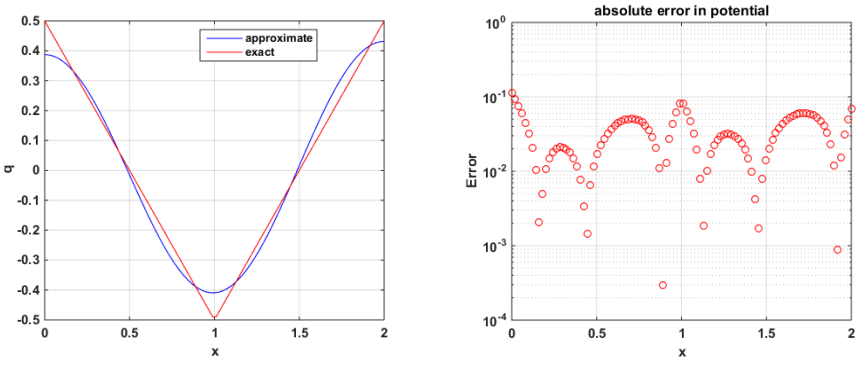
Figure 4.5: Reconstructed potential (blue) and exact potential (red) $p(x) = \cos(x)$ (**a**) $\delta = 0.1$ (**b**) $\delta = 0.05$.

It is obvious that reconstruction is possible even in the presence of a significant noise. ◀

▶ **Example 4.4 (Inverse SLP with a non-smooth potential).** This example extends Barcilon's algorithm [17] by reconstructing a non-smooth potential:

$$u'' + (\lambda - p(x))u = 0, \quad x \in (0, 2)$$

with $p(x) = |1 - x| - 0.5$: symmetric and normalized, but non-smooth. The truncated eigenvalue sequences at $N = 4$, for DDBCs and DNBCs are calculated using the Magnus method (with $m = 10$, $n = 100$, $L = 5$), and used in the reconstruction of the potential.



(a) (b)

Figure 4.6: (**a**) Reconstructed potential (blue) and exact potential (red) $p(x) = |1-x|-0.5$ (**b**) Log absolute error in the reconstructed p.

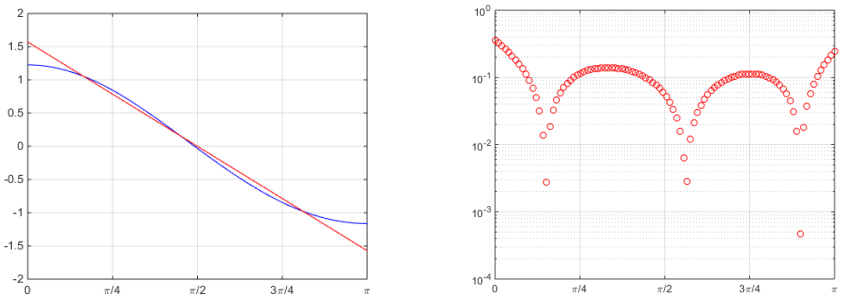
Figure 4.6 shows the reconstructed and exact potential, and the log absolute errors in the reconstructed potential, respectively, using one inverse algorithm step. From Figure 4.6(a), it is obvious that the potential is converging towards the exact one. The max-norm of difference between the reconstructed p and the actual p is $\approx 1.13 \times 10^{-1}$ and the max-norm of difference between the reconstructed eigenvalues and the actual ones is $\approx 3.15 \times 10^{1}$. As anticipated, the error is maximum at the point of non-differentiability and at the boundaries. ◀

4.3.2 Inverse FSLP

▶ **Example 4.5 (FSLP).** Consider

$$y^{(4)} - (p_1 y')' + p_2 y = \lambda y, \qquad 0 < x < \pi$$

with $p_1 \equiv 0$ and $p_2(x) = \pi/2 - x$. Using the three spectra corresponding to the sets of BCs: $\{y(0) = y''(0) = y(1) = y''(1) = 0\}$, $\{y(0) = y'(0) = y(1) = y''(1) = 0\}$ and $\{y'(0) = y''(0) = y(1) = y''(1) = 0\}$ two potential functions p_1, p_2 are reconstructed starting with zero initial guesses and $N = 4$, $m = 20$, $n = 100$, $L = 5$, $M = 6$ (Figure 4.7). The max-norm of difference between the reconstructed p_2 and actual p_2 is $\approx 3.5 \times 10^{-1}$ and the max-norm of difference between the reconstructed eigenvalues and the actual ones is $\approx 2.1 \times 10^{-1}$.



(a) **(b)**

Figure 4.7: (**a**) Reconstructed potential (blue) and exact potential (red) $p_2(x) = \pi/2 - x$ (**b**) Log absolute error in the reconstructed p_2.

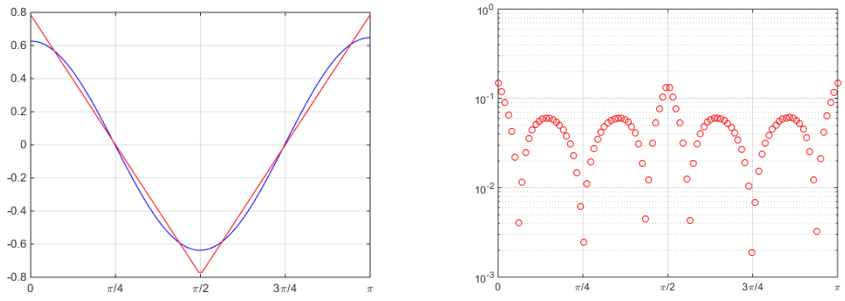
◀

▶ **Example 4.6 (FSLP).** Consider

$$y^{(4)} - (p_1 y')' + p_2 y = \lambda y, \qquad 0 < x < \pi$$

with $p_1(x) = |x - \pi/2| - \pi/4$ and $p_2 \equiv 0$. Using the three spectra corresponding to the sets of BCs: $y(0) = y''(0) = y(1) = y''(1) = 0$, $y(0) = y'(0) = y(1) = y''(1) = 0$ and $y'(0) = y''(0) = y(1) = y''(1) = 0$ two potential functions p_1, p_2 are reconstructed starting with zero initial guesses and $N = 4$, $m = 20$, $n = 100$, $L = 15$, $M = 6$ (Figure 4.8). The max-norm of difference between the

reconstructed p_1 and the actual p_1 is $\approx 1.4 \times 10^{-1}$ and the max-norm of difference between the reconstructed eigenvalues and the actual ones is $\approx 7.5 \times 10^{-1}$. ◀



(a) **(b)**

Figure 4.8: (**a**) Reconstructed potential (blue) and exact potential (red) $p_1(x) = |x - \pi/2| - \pi/4$ (**b**) Log absolute error in the reconstructed p_1.

4.4 Discussion

Here, the inverse SLP is solved with smooth and non-smooth potential, even in the presence of noise. It is observed that the method is successful even in the presence of significant noise, provided that the assumptions of the algorithm is satisfied.

Last few examples solve the inverse FSLP using the Barcilon's algorithm with the initial knowledge of three spectra. A simplified FSLP is solved keeping one of the unknown potential functions zero.

According to Andrew [9] there are three major sources of error with the inverse problem:

E4: attempting to compute q using only a finite (and often quite small) number of eigenvalues although the complete infinite set is required for the determination of q,

E5: only approximations of the eigenvalues are available and errors in the given eigenvalues may be especially serious for higher eigenvalues,

E6: all errors in the numerical solution of the direct problem cause errors in the solution of the inverse problem.

The last one is the only error source which is affected by the choice of numerical method and which can be reduced by using a high accuracy method such as Magnus.

5 Discussion and Conclusion

There are several places where approximation errors occur in the Magnus method. For the direct problem, they are (see Section 3.2.5 also):

E1: truncation of the Magnus series

E2: calculation of multivariate integrals

E3: computing the matrix exponent

and in the case of the inverse problem (see also section 4.4):

E4: using a finite number of eigenvalues

E5: using approximate eigenvalues

E6: errors in the direct problem

E7: computing the explicit form of q by an interpolation method

E8: approximating derivatives and integrals using numerical methods

so that the accumulated error would be high for the direct and inverse SLPs. For the direct SLP, we can address the error E1, by using a higher order Magnus method, error E2, by using higher order quadrature methods, and E3 by using a more accurate method from the methods proposed by Moler and Van Loan [85]. For the inverse SLP, the error E4 is no longer a problem as Aceto et al. [3] argues, that the first eigenvalues are the most important for the reconstruction of the unknown potential. And the errors E5, and E6 can be also reduced by reducing the errors in the direct problem (E1, E2, and E3). E8 can be eliminated by using exact derivatives and integrals when possible. So it turn outs that the error in the inverse problem solely depends on the errors in the direct problem and the error E7 (interpolating q from a set of points).

By construction, Magnus method leads to a global error of order $O(h^p)$ if a pth-order Magnus method is applied, yet it turns out that the error also depends

on the magnitude of the eigenvalue. Specifically, the error in a pth-order method grows as $O(h^{p+1}\lambda^{p/2-1})$, and thus one expects poor approximations for larger eigenvalues [20]. This method has the disadvantage that the cost of integration, while increasing very slowly as a function of λ, will be $O\left(\left(\begin{array}{c} 2n \\ n \end{array}\right)^2\right)$ for a problem arising from a $2n$th order Sturm–Liouville problem [55].

It is clear, the Magnus method is able to deal with general kind of boundary conditions, since the boundary conditions are presented in a matrix form. Here, it is shown that the proposed technique has the ability to solve regular and some singular Sturm–Liouville problems of any even order, efficiently. By Ledoux et al. [60] a modified Magnus method can be used with Lobatto points to improve the error. Although in theory, this method can be used for solving higher order SLPs, for it to be effective in practise, the various errors in the approximation steps, should be addressed adequately.

More than 40 years after Barcilon's paper [17], this work gives a concrete implementation of the inverse SLP algorithm proposed therein, to our knowledge for the first time. Furthermore, computational feasibility and applicability of this algorithm for solving inverse SLPs of higher order is verified successfully in this paper (for $n = 2$ and $n = 4$).

The Magnus method was tested with finite interval SLPs of even orders (upto 8) along with regular and a finite singular endpoint BC, and infinite intervals. Good accuracy with regard to first few eigenvalues is obtained. It is an open problem whether this method can be extend to solve other types of EVPs such as: different finite singular endpoints, and multiple eigenvalues. In a future work, it is possible to extend the results to solve even higher order SLPs.

In a future work, the algorithm for inverse SLP may be modified to include general finite interval problems (i.e. problems in the domain (a, b)): for different sets of BCs (provided the spectra are interlaced), infinite BCs, and different classes of potential functions, such as: non-smooth, discontinuous, oscillating, etc. Also a more suitable interpolation method for reconstructing q may be examine.

In conclusion, this work provides a method that can be adapted successfully for solving a direct (regular/singular) or inverse SLP of an arbitrary order with arbitrary BCs.

A Appendix

► **Example A.1 (Multi-section method).** Figure A.1 compares the log errors for the bisection and multisection methods for approximating the root of $f(x) = x^3 - x - 2$ in the interval $[1, 2]$ using $m = 3, 10, 100$ subdivisions. Even after more than 20 iterations, bisection method reports an error of $1.5E - 08$ where as after only 7 iterations multisection method with $m = 100$ reached an error $4.5E - 14$. Even $m = 3$ multisection method converged within 19 iterations to the same accuracy. Obviously, the computation time increases and error decreases with the number of steps m and tolerance (Figure A.2). ◄

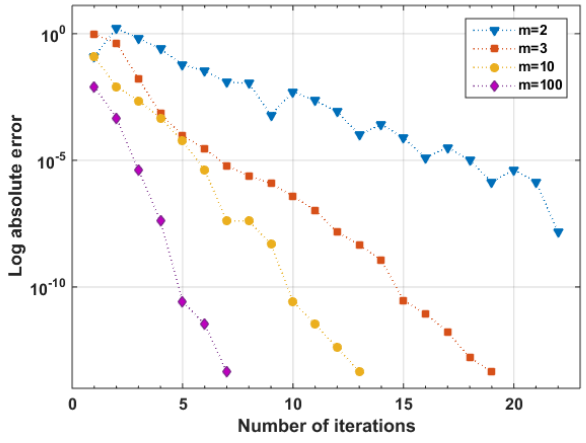
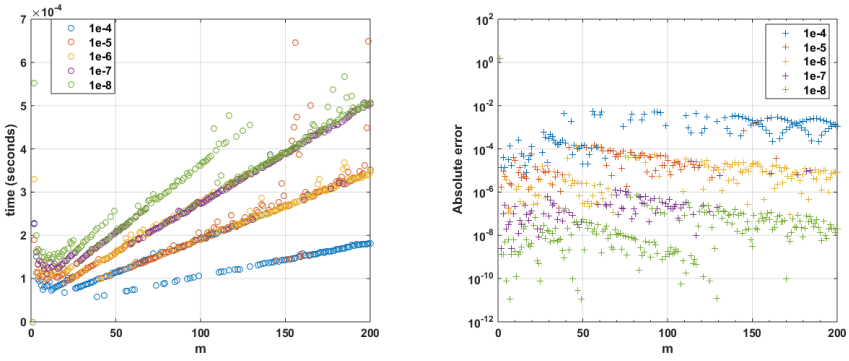


Figure A.1: Log errors for the bisection ($m = 2$) and multisection method for approximating the root of $f(x) = x^3 - x - 2$ in the interval $[1, 2]$ using $m = 3, 10, 100$ subdivisions.

Table A.1, shows the Profile summary of the Matlab implementation of Magnus method. It can be seen that the most time spent on calculating Ω, and most function calls are for the functions G, p, w, and q (which are the coefficients matrix and the coefficients of the equation, respectively). The most self time is utilized by the function G. Also, problem (B) requires more computational

time than for problem (A) with other factors remaining fixed, as the latter have a constant G matrix.



(a) **(b)**

Figure A.2: (a) Time and **(b)** log absolute error for the multisection method for approximating the root of $f(x) = x^3 - x - 2$ in the interval $[1, 2]$ using $m = 2, \ldots, 200$ subdivisions with tolerance in $\{1e-4, \ldots, 1e-8\}$.

Table A.1: Profile summary of the `slp` function generated by Matlab with $m = 10$, $n = 10$, $L = 5$ for finding the eigenvalue in the **(A)** interval $[0, 10]$ for $y'' + \lambda y = 0, y(0) = y(1) = 0$ and **(B)** interval $[0, 2]$ for $y'' + (x + 0.1)^{-2} y = \lambda y, y(0) = y(\pi) = 0$. *Self time is the time spent in a function excluding the time spent in its child functions. Self time also includes overhead resulting from the process of profiling. All time values are in seconds.

Function Name	Calls	Total Time		Self Time*	
		A	B	A	B
`slp`	1	2.442 s	2.461 s	0.028 s	0.031 s
`Omega`	660	2.323 s	2.292 s	0.024 s	0.030 s
`R3`	660	2.163 s	2.200 s	0.056 s	0.032 s
`G`	44880	1.454 s	1.480 s	0.948 s	1.014 s
`R2`	1320	1.336 s	1.404 s	0.040 s	0.063 s
`R1`	3960	1.114 s	1.307 s	0.096 s	0.095 s
`Q2`	9240	0.843 s	0.900 s	0.140 s	0.124 s
`Q3`	4620	0.736 s	0.548 s	0.096 s	0.078 s
`G2`	17160	0.711 s	0.653 s	0.140 s	0.110 s
`G1`	13860	0.602 s	0.516 s	0.104 s	0.078 s
`Q1`	12540	0.527 s	0.623 s	0.076 s	0.048 s
`G3`	13860	0.481 s	0.654 s	0.096 s	0.156 s
`p`	44880	0.216 s	0.171 s	0.216 s	0.171 s
`w`	44880	0.180 s	0.141 s	0.180 s	0.141 s
`q`	44880	0.110 s	0.155 s	0.110 s	0.155 s
`expm`	660	0.091 s	0.139 s	0.008 s	0.031 s
`expm>PadeApproximantOfDegree`	660	0.067 s	0.108 s	0.067 s	0.092 s
`expm>expmchk`	660	0.016 s	0.000 s	0.016 s	0.000 s
`expm>getPadeCoefficients`	660	0.000 s	0.016 s	0.000 s	0.016 s

B Appendix: Important Coding

Listing B.1: Multisection method

```
function y=multisection(f,a,b,tol,m)
%% Multisection method for finding a root in an interval
% Input:
%_f_ (function handle)
%_a_ (double) left end point of the interval
%_b_ (double) right end point of the interval
%_tol_ (double) tolerance
%_m_ (integer) # maximum iterations
% Output:
%_y_ (double) approximate root of $f$ in the interval [a, b] with a
    tolerance of 'tol'
% and maximum number of iterations m
% Author: Upeksha Perera (April 2019)
% Supplementary Material for the article titled
% Solutions of Direct and Inverse Even-order Sturm-Liouville problems using
    Magnus expansion
% by Upeksha Perera and Christine Bockmann
% Correspondence: upeksha@kln.ac.lk; Department of Mathematics, University
    of Kelaniya, 11600 Kelaniya, Sri Lanka;
% Current address: Institut fur Mathematik, Universitat Potsdam, 14476
    Potsdam, Germany; bodhiyabadug@uni-potsdam.de
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % USAGE:
% % Example: Finding the  root  in the  interval [1,2]  for  the  function $
    f=x^3-x-2$
% % with a tolerance 1e-7 and maximum  of 10 iterations
% % a=1;b=2;
% % f=@(x) x.^3-x-2;
% % tol=1e-7;
% % m=10;
% % y=multisection(f,a,b,tol,m)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

h=(b-a)/m;   % step -size
x=a:h:b; % sequence  of  numbers: x1=a , x2=a+h , ... , xm=b

```

```
values=zeros(1,m);  % store function values at each subdivision point

err = 1;
while err > tol
    for k=1:m+1
        values(k)=f(x(k));  % calculate function value at each point k
    end

    scinter = find(diff(sign(values))); % find the sign changing position of
         f

    % reset the right and left end points to bracket the sign changing
        position
    b=x(scinter(1)+1);
    a=x(scinter(1));

    % redefine the step size
    h=(b-a)/m;

    % refine the root interval
    x=a:h:b;

    err=abs(f(x));
end % end of while loop (multisection)
y=(a+b)/2; % approximate root
```

Listing B.2: Magnus method

```
function ev=magnus_slp(a,b,lambda0,lambdaS,p,q,w,m,n,LM,A1,A2,B1,B2)

%% Magnus method for Sturm-Liouville problems of the form
% $$ (p(x)y')'+q(x)y=\lambda w(x) y $$
% Input:
%_a_ (double) left end point of the x-domain
%_b_ (double) right end point of the x-domain
%_lambdaS_ (double)  right end of eigenvalue interval
%_lambda0_ (double) left end of eigenvalue interval
%_q_ (function) coefficient of y
%_p_ (function) coefficient of y''
%_w_ (function) coefficient of lambda y term
%_m_ (integer) # divisions for the eigenvalue interval
%_n_ (integer) # divisions for the [a,b]
%_LM_ (integer) #  multisection iteration steps
%_A1_ (0 or 1) coefficient of boundary condition y(a)
%_A2_ (0 or 1) coefficient of boundary condition y'(a)
```

```
%_B1_ (0 or 1) coefficient of boundary condition y(b)
%_B2_ (0 or 1) coefficient of boundary condition y'(b)
%% note that A1,A2,B1,B2  selected such that
% A1*A2'=A2*A1', B1*B2'=B2*B1' and (A1,A2) and (B1,B2) have rank 1
% Output:
%_ev_ (double) approximate eigenvalue in the interval [lambda0, lambdaS]
% Author: Upeksha Perera (April 2019)
% Supplementary Material for the article titled
% Solutions of Direct and Inverse Even-order Sturm-Liouville problems using Magnus expansion
% by Upeksha Perera and Christine Bockmann
% Correspondence: upeksha@kln.ac.lk; Department of Mathematics, University of Kelaniya, 11600 Kelaniya, Sri Lanka;
% Current address: Institut fur Mathematik, Universitat Potsdam, 14476 Potsdam, Germany; bodhiyabadug@uni-potsdam.de
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % USAGE:
% % Example 1: Finding the  eigenvalue  in the  interval [0, 10]  for  the  problem
% % y''+lambda y=0, y(0)=y(1)=0
%
% % a=0; b=1;
% % lambda0=0; lambdaS=10;
% % q=@(x) 0;
% % p=@(x) 1;
% % w=@(x) 1;
% % m=100; n=100;
% % LM=5;
% % A1=1; A2=0;
% % B1=1; B2=0;
% % ev=magnus_slp(a,b,lambda0,lambdaS,p,q,w,m,n,LM,A1,A2,B1,B2);

% % Example 2: Paine problem 2 in Pryce [1993] from Paine et al. [1981].
% % y''+1/(x+0.1)^2 y=lambda y, y(0)=y(pi)=0
% a=0; b=pi;
% lambda0=0; lambdaS=2;
% q=@(x) 1./(x+0.1).^2;
% p=@(x) -1;
% w=@(x) 1;
% m=10; n=10; L=5;
% A1=1; A2=0;
% B1=1; B2=0;
% ev=magnus_slp(a,b,lambda0,lambdaS,p,q,w,m,n,L,A1,A2,B1,B2)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```
h1=(lambdaS-lambda0)/m;  % step-size of eigenvalue interval
lambda=lambda0:h1:lambdaS; % sequence of numbers: lambda0, lambda1, ... ,
    lambdaS
h=(b-a)/n; % step-size for the domain
x=a:h:b; % sequence of numbers a=x0, x1, ..., xn=b

Y=eye(2);  % initial value of Y(0)

A=[A1,A2; 0 0];
B=[0 0;B1,B2];

G=@(x,lambda)[0  1./p(x) ;  lambda.*w(x)-q(x) 0]; % matrix G

c1=1/2-sqrt(15)/10; % Gaussian point c1
c3=1/2+sqrt(15)/10; % Gaussian point c3

G1=@(x,lambda) G(x+c1.*h,lambda); % G(c1*h)
G2=@(x,lambda) G(x+h/2,lambda); % G(c2*h), c2=1/2
G3=@(x,lambda) G(x+c3.*h,lambda); % G(c3*h)

Q1=@(x,lambda) h.*G2(x,lambda); % Q1=h*G(c1*h)
Q2=@(x,lambda) (sqrt(15)*h/3).*(G3(x,lambda)-G1(x,lambda));
Q3=@(x,lambda) (10*h/3).*(G3(x,lambda)-2*G2(x,lambda)+G1(x,lambda));

R1=@(x,lambda)Q1(x,lambda)*Q2(x,lambda)-Q2(x,lambda)*Q1(x,lambda);
% R1=[Q1,Q2]=Q1*Q2-Q2*Q1

R2=@(x,lambda) Q1(x,lambda)*(2.*Q3(x,lambda)+R1(x,lambda)) ...
    -(2.*Q3(x,lambda)+R1(x,lambda))*Q1(x,lambda);  % R2=[Q1,2Q3+R1]

R3=@(x,lambda)(-20*Q1(x,lambda)-Q3(x,lambda)+R1(x,lambda))*(Q2(x,lambda) ...
    -R2(x,lambda)./60)-(Q2(x,lambda)-R2(x,lambda)./60)*(-20*Q1(x,lambda) ...
    -Q3(x,lambda)+R1(x,lambda));

sigma=@(x,lambda) Q1(x,lambda)+Q3(x,lambda)./12+R3(x,lambda)./240;

%%%%%%%%%%% the multisection method starts %%%%%%%%%
L=0;
while L<LM % # multisection steps
    for k=1:m+1
        Y=eye(2);
        for j=1:n
            Y=expm(sigma(x(j),lambda(k)))*Y; % Calculate Y(b) iteratively
        end
```

```
        F(k)=det(A+B*Y);  % calculate characteristic function at each point
            k
    end
    plot(F)

    scinter = find(diff(sign(F))); % find the sign changing position of F

    % reset the right and left end points to bracket the sign changing
        position
    lambdaS=lambda(scinter(1)+1);
    lambda0=lambda(scinter(1));

    % redefine the step size
    h1=(lambdaS-lambda0)/m;

    % refine the possible lambda values
    lambda=lambda0:h1:lambdaS;

    L=L+1; % increase the multisection counter
end % end of while loop (multisection)
%%%%%%%%%%%%%%%% end of multisection method %%%%%%%%%%

ev=(lambda0+lambdaS)/2; % approximate eigenvalue
```

Listing B.3: Inverse SLP algorithm

```
function [q0,nu0]=inverse_slp(a,b,m,n,L,LL,nue,lambda0,lambdaS,M1,A11, A21,
    B11,B21,A12 ,A22, B12, B22)
%% Barcilon's inverse SLP method using Magnus method for Sturm-Liouville
    problems of the form
% as described in the paper:
% Iterative solution of the inverse Sturm--Liouville problem
% Journal of Mathematical Physics 15, 429 (1974); https://doi.org
    /10.1063/1.1666664
% Victor Barcilon
% $$ (p(x)y')'+q(x)y=\lambda w(x) y $$
% p=w=1 and sets of BCs: y(0)=y(1)=0 and y'(0)=y(1)=0
%% Input:
%_a_ (double) left end point of the x-domain
%_b_ (double) right end point of the x-domain
%_lambdaS_ (double)  right end of eigenvalue interval
%_lambda0_ (double) left end of eigenvalue interval
%_m_ (integer) # divisions for the eigenvalue interval
%_n_ (integer) # divisions for the [a,b]
%_L_ (integer) #  multisection iteration steps
```

```
%_LL_ (integer) #  length of exact eigenvalue sequence
%_M1_ (integer) #  inverses iteration steps
%_nue_ (array) #  exact eigenvalue sequence
%% Output:
%_q0_ (function) reconstructed potential
%_nu0_ (array) reconstructed eigenvalue sequence
% Author: Upeksha Perera (April 2019)
% Supplementary Material for the article titled
% Solutions of Direct and Inverse Even-order Sturm-Liouville problems using Magnus expansion
% by Upeksha Perera and Christine Bockmann
% Correspondence: upeksha@kln.ac.lk; Department of Mathematics, University of Kelaniya, 11600 Kelaniya, Sri Lanka;
% Current address: Institut fur Mathematik, Universitat Potsdam, 14476 Potsdam, Germany; bodhiyabadug@uni-potsdam.de
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% USAGE:
% % Example: Reconstructing the potential for  the  problem
% % -y''+cos(t)*y=lambda y,
% % given the corresponding eigenvalue sequences for y(0)=y(1)=0 and y'(0)=y(1)=0
% %
% a=0; b=pi; % end points
% lambda0=0; lambdaS=20;   % eigenvalue search interval
% m=10;n=10; L=2; % parameters for magnus method
% A11=1; A21= 0; B11=1 ; B21=0 ; % first set of BCs
% A12=0  ; A22= 1; B12= 1 ; B22=0; % second set of BCs
% q0e=@(t) cos(t); % exact potential
% p=@(t) -ones(size(t)); w0=@(t) ones(size(t)); % these two coefficients are fixed
% M1=2; % number of inverse algorithm steps

% lambdae=magnus_slp(a,b,lambda0,lambdaS,p,q0e,w0,m,n,L,A11,A21,B11,B21);
% mue=magnus_slp(a,b,lambda0,lambdaS,p,q0e,w0,m,n,L,A12,A22,B12,B22);
% LL=min([length(lambdae),length(mue)]);
% for k=1:LL % interlacing
%     nue(2*k-1,:)=4*mue(k);
%     nue(2*k,:)=4*lambdae(k);
% end
% % nue: exact eigenvalues
%
% [q0,nu0]=inverse_slp(a,b,m,n,L,LL,nue,lambda0,lambdaS,M1,A11, A21, B11,B21 ,A12 ,A22, B12, B22);
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A11=1; A21= 0; B11=1 ; B21=0 ; % first set of BCs
```

```
A12=0  ; A22= 1; B12= 1 ; B22=0; % second set of BCs

x=linspace(a,b,n); % partitioning x-domain
sum1=zeros(M1,length(x)); % store the error terms

%% constructing w and omega for q=0
for k=1:2*lambdaS
    cc1=sqrt(2/(3*b));
    w(k,:)=(cos(k.*x*pi/b)-cos(k*pi))*cc1;
    dw(k,:)=-(k*pi/b)*sin(k.*x*pi/b)*cc1;

    cc2=sqrt(2/b);
    omega(k,:)=sin(k.*x*pi/b)*cc2;
    domega(k,:)=(k*pi/b)*cos(k.*x*pi/b)*cc2;

    int1(k)=-sqrt(2*b/3)*cos(k*pi);
    int2(k)=k*pi/(b*sqrt(3));
    c(k)=int1(k)./int2(k);
    dw1(k,:)=c(k)*domega(k,:);
end

p=@(t) -ones(size(t)); w0=@(t) ones(size(t)); % these two coefficients are
    fixed
q0=@(t) zeros(size(t));% initial guess

lambda=magnus_slp(a,b,lambda0,lambdaS,p,q0,w0,m,n,L,A11,A21,B11,B21);
mu=magnus_slp(a,b,lambda0,lambdaS,p,q0,w0,m,n,L,A12,A22,B12,B22);

for k=1:LL % interlacing
    nu0(2*k-1,:)=4*mu(k);
    nu0(2*k,:)=4*lambda(k);
end
% nu0: approximate eigenvalues
%%
for M=1:M1 % iteration loop

    for kk=1:LL
        sum1(M,:)=sum1(M,:)+dw1(kk,:).*(nue(kk)-nu0(kk));
    end
    sum1(M,:)=0.25*sum1(M,:);

    Q0=q0(x)+sum1(M,:); % updating q
    pp=griddedInterpolant(x',Q0','pchip'); % getting the functional form of
        q
    q0=@(t) pp(t);
```

```
99
100     % calculating new eigenvalues using updated q
101     lambda=[]; mu=[]; nu0=[];
102     lambda=magnus_slp(a,b,lambda0,lambdaS,p,q0,w0,m,n,L,A11,A21,B11,B21);
103     mu=magnus_slp(a,b,lambda0,lambdaS,p,q0,w0,m,n,L,A12,A22,B12,B22);
104     for k=1:LL % interlacing
105         nu0(2*k-1,:)=4*mu(k);
106         nu0(2*k,:)=4*lambda(k);
107     end
108 end
```

Bibliography

[1] S Abbasbandy and A Shirzadi. **A new application of the homotopy analysis method: Solving the Sturm–Liouville problems**. *Communications in Nonlinear Science and Numerical Simulation* 16:1 (2011), 112–126. DOI: 10.1016/j.cnsns.2010.04.004.

[2] L Aceto, P Ghelardoni, and C Magherini. **Boundary Value Methods as an extension of Numerov's method for Sturm–Liouville eigenvalue estimates**. *Applied Numerical Mathematics* 59:7 (2009), 1644–1656. DOI: 10.1016/j.apnum.2008.11.005.

[3] L Aceto, P Ghelardoni, and C Magherini. **Boundary value methods for the reconstruction of Sturm–Liouville potentials**. *Applied Mathematics and Computation* 219:6 (2012), 2960–2974. DOI: 10.1016/j.amc.2012.09.021.

[4] L Aceto, P Ghelardoni, and C Magherini. **BVMs for Sturm-Liouville eigenvalue estimates with general boundary conditions**. *Journal of Numerical Analysis,Industrial and Applied Mathematics* 4 (2009), 113–127.

[5] M Allame, H Ghasemi, and MT Kajani. **An appropriate method for eigenvalues approximation of sixth-order Sturm-Liouville problems by using integral operation matrix over the Chebyshev polynomials**. In: *AIP Conference Proceedings*. Vol. 1684. AIP Publishing. 2015, 090001. DOI: 10.1063/1.4934326.

[6] V Ambarzumian. **Über eine frage der eigenwerttheorie**. *Zeitschrift für Physik A Hadrons and Nuclei* 53:9 (1929), 690–695.

[7] AL Andrew. **Computing Sturm–Liouville potentials from two spectra**. *Inverse Problems* 22:6 (2006), 2069–2081. DOI: 10.1088/0266-5611/22/6/010.

[8] AL Andrew. **Numerical solution of inverse Sturm–Liouville problems**. *ANZIAM Journal* 45 (2004), 326–337. DOI: 10.21914/anziamj.v45i0.891.

[9] AL Andrew. **Numerov's method for inverse Sturm–Liouville problems**. *Inverse Problems* 21:1 (2005), 223–238. DOI: 10.1088/0266-5611/21/1/014.

[10] AL Andrew and JW Paine. **Correction of finite element estimates for Sturm-Liouville eigenvalues**. *Numerische Mathematik* 50:2 (1986), 205–215. DOI: 10.1007/BF01390430.

[11] AL Andrew and JW Paine. **Correction of Numerov's eigenvalue estimates**. *Numerische Mathematik* 47:2 (1985), 289–300. DOI: 10.1007/BF01389712.

[12] MT Atay and S Kartal. **Computation of Eigenvalues of Sturm-Liouville Problems using Homotopy Perturbation Method**. *International Journal of Nonlinear Sciences and Numerical Simulation* 11:2 (2010). DOI: 10.1515/ijnsns.2010.11.2.105.

[13] BS Attili and D Lesnic. **An efficient method for computing eigenelements of Sturm-Liouville fourth-order boundary value problems**. *Applied mathematics and computation* 182:2 (2006), 1247–1254. DOI: 10.1016/j.amc.2006.05.011.

[14] PB Bailey, WN Everitt, and A Zettl. **Computing eigenvalues of singular Sturm-Liouville problems**. *Results in Mathematics* 20:1-2 (1991), 391–423. DOI: 10.1007/BF03323182.

[15] PB Bailey, MK Gordon, and LF Shampine. **Automatic solution of the Sturm-Liouville problem**. *ACM Transactions on Mathematical Software (TOMS)* 4:3 (1978), 193–208. DOI: 10.1145/355791.355792.

[16] V Barcilon. **Iterative solution of the inverse Sturm-Liouville problem**. *Journal of Mathematical Physics* 15:4 (1974), 429–436. DOI: 10.1063/1.1666664.

[17] V Barcilon. **On the solution of inverse eigenvalue problems of high orders**. *Geophysical Journal International* 39:1 (1974), 143–154. DOI: 10.1111/j.1365-246X.1974.tb05444.x.

[18] V Barcilon. **On the uniqueness of inverse eigenvalue problems**. *Geophysical Journal of the Royal Astronomical Society* 38:2 (1974), 287–298. DOI: 10.1111/j.1365-246x.1974.tb04121.x.

[19] GV Berghe and H De Meyer. **A finite-element estimate with trigonometric hat functions for Sturm–Liouville eigenvalues**. *Journal of computational and applied mathematics* 53:3 (1994), 389–396. DOI: 10.1016/0377-0427(94)90066-3.

[20] S Blanes, F Casas, JA Oteo, and J Ros. **The Magnus expansion and some of its applications**. *Physics reports* 470:5-6 (2009), 151–238. DOI: 10.1016/j.physrep.2008.11.001.

[21] RP Boas. **Entire functions**. Academic Press, 2011.

[22] C Böckmann and A Kammanee. **Broyden method for inverse non-symmetric Sturm-Liouville problems**. *BIT Numerical Mathematics* 51:3 (2011), 513–528. DOI: 10.1007/s10543-011-0317-5.

[23] C Böckmann and A Rattana. **An inverse fourth order sturm-liouville problem**. In: *AIP Conference Proceedings*. Vol. 1648. 1. AIP Publishing LLC. 2015, 850030. DOI: 10.1063/1.4913085.

[24] G Borg. **Eine Umkehrung der Sturm-Liouvilleschen Eigenwertaufgabe: Bestimmung der Differentialgleichung durch die Eigenwerte**. *Acta Mathematica* 78 (1946), 1–96. DOI: 10.1007/BF02421600.

[25] I Çelik. **Approximate computation of eigenvalues with Chebyshev collocation method**. *Applied Mathematics and Computation* 168:1 (2005), 125–134. DOI: 10.1016/j.amc.2004.08.024.

[26] B Chanane. **Accurate solutions of fourth order Sturm–Liouville problems**. *Journal of Computational and Applied Mathematics* 234:10 (2010), 3064–3071. DOI: 10.1016/j.cam.2010.04.023.

[27] B Chanane. **Eigenvalues of fourth order Sturm-Liouville problems using Fliess series**. *Journal of computational and applied mathematics* 96:2 (1998), 91–97. DOI: 10.1016/s0377-0427(98)00086-7.

[28] B Chanane. **Fliess series approach to the computation of the eigenvalues of fourth-order Sturm-Liouville problems**. *Applied mathematics letters* 15:4 (2002), 459–463. DOI: 10.1016/s0893-9659(01)00159-8.

[29] B Chanane. **Sturm–Liouville problems with parameter dependent potential and boundary conditions**. *Journal of Computational and Applied Mathematics* 212:2 (2008), 282–290. DOI: 10.1016/j.cam.2006.12.006.

[30] MM Chawla. **A new fourth-order finite-difference method for computing eigenvalues of fourth-order two-point boundary-value problems**. *IMA Journal of Numerical Analysis* 3:3 (1983), 291–293. DOI: 10.1093/imanum/3.3.291.

[31] MM Chawla and CP Katti. **A new symmetric five-diagonal finite difference method for computing eigenvalues of fourth-order two-point boundary value problems**. *Journal of Computational and Applied Mathematics* 8:2 (1982), 135–136. DOI: 10.1016/0771-050X(82)90069-9.

[32] MM Chawla and CP Katti. **On Noumerov's method for computing eigenvalues**. *BIT Numerical Mathematics* 20:1 (1980), 107–109. DOI: 10.1007/BF01933592.

[33] L Chen and H Ma. **Approximate solution of the Sturm–Liouville problems with Legendre–Galerkin–Chebyshev collocation method**. *Applied Mathematics and Computation* 206:2 (2008), 748–754. DOI: 10.1016/j.amc.2008.09.038.

[34] EA Coddington and N Levinson. **Theory of ordinary differential equations**. Tata McGraw-Hill Education, 1955.

[35] R Darzi and B Agheli. **The Chebyshev Collocation Method for Finding the Eigenvalues of Fourth-Order Sturm-Liouville Problems**. *Bulletin of Mathematical Sciences and Applications* 15 (2016), 62–68. DOI: 10.18052/www.scipress.com/bmsa.15.62.

[36] MC Drignei. **A Newton-type method for solving an inverse Sturm-Liouville problem**. *Inverse Problems in Science and Engineering* 23:5 (2014), 851–883. DOI: 10.1080/17415977.2014.947478.

[37] MC Drignei. **Constructibility of an $L^2_R(0, a)$ solution to an inverse Sturm–Liouville problem using three Dirichlet spectra**. *Inverse Problems* 26:2 (2009), 025003. DOI: 10.1088/0266-5611/26/2/025003.

[38] MC Drignei. **Inverse Sturm-Liouville problems using multiple spectra**. PhD thesis. Iowa State University, 2008. URL: https://lib.dr.iastate.edu/rtd/15694.

[39] MC Drignei. **Numerical reconstruction in a three-spectra inverse Sturm–Liouville problem with mixed boundary conditions**. *Inverse Problems in Science and Engineering* 21:8 (2013), 1368–1391. DOI: 10.1080/17415977.2013.764603.

[40] L Efremova and G Freiling. **Numerical solution of inverse spectral problems for Sturm-Liouville operators with discontinuous potentials**. *Open Mathematics* 11:11 (2013), 2044–2051. DOI: 10.2478/s11533-013-0301-1.

[41] RH Fabiano, R Knobel, and BD Lowe. **A finite-difference algorithm for an inverse Sturm-Liouville problem**. *IMA journal of numerical analysis* 15:1 (1995), 75–88. DOI: 10.1093/imanum/15.1.75.

[42] G Freiling and VA Yurko. **Inverse Sturm-Liouville problems and their applications**. NOVA Science Publishers New York, 2001.

[43] M El-Gamel, MS El-Azab, and M Fathy. **An efficient technique for finding the eigenvalues and the eigenelements of fourth-order Sturm-Liouville problems**. *SeMA Journal* 74:1 (2016), 37–56. DOI: 10.1007/s40324-016-0079-8.

[44] M El-Gamel and M Sameeh. **An Efficient Technique for Finding the Eigenvalues of Fourth-Order Sturm-Liouville Problems**. *Applied Mathematics* 03:08 (2012), 920–925. DOI: 10.4236/am.2012.38137.

[45] Q Gao. **Decent flow methods for inverse Sturm–Liouville problem**. *Applied Mathematical Modelling* 36:9 (2012), 4452–4465. DOI: 10.1016/j.apm.2011.11.070.

[46] Q Gao, X Cheng, and Z Huang. **Modified Numerov's method for inverse Sturm–Liouville problems**. *Journal of Computational and Applied Mathematics* 253 (2013), 181–199. DOI: 10.1016/j.cam.2013.04.025.

[47] P Ghelardoni. **Approximations of Sturm-Liouville eigenvalues using boundary value methods**. *Applied Numerical Mathematics* 23:3 (1997), 311–325. DOI: 10.1016/s0168-9274(96)00073-6.

[48] P Ghelardoni and C Magherini. **BVMs for computing Sturm–Liouville symmetric potentials**. *Applied Mathematics and Computation* 217:7 (2010), 3032–3045. DOI: 10.1016/j.amc.2010.08.036.

[49] L Greenberg and M Marletta. **Algorithm 775: The code SLEUTH for solving fourth-order Sturm-Liouville problems**. *ACM Transactions on Mathematical Software (TOMS)* 23:4 (1997), 453–493. DOI: 10.1145/279232.279231.

[50] L Greenberg and M Marletta. **Oscillation Theory and Numerical Solution of Sixth Order Sturm–Liouville Problems**. *SIAM journal on numerical analysis* 35:5 (1998), 2070–2098. DOI: 10.1137/s0036142997316451.

[51] OH Hald. **The inverse Sturm-Liouville problem and the Rayleigh-Ritz method**. *Mathematics of Computation* 32:143 (1978), 687–705. DOI: 10.1090/S0025-5718-1978-0501963-2.

[52] M Ignatiev and V Yurko. **Numerical Methods for Solving Inverse Sturm–Liouville Problems**. *Results in Mathematics* 52:1 (2008), 63–74. DOI: 10.1007/s00025-007-0276-y.

[53] A Iserles. **Magnus expansions and beyond**. *Combinatorics and physics* 539 (2008), 171–186. DOI: 10.1090/conm/539/10634.

[54] A Iserles and SP Norsett. **On the solution of linear differential equations in Lie groups**. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 357:1754 (1999), 983–1019. DOI: 10.1098/rsta.1999.0362.

[55] L Jódar and M Marletta. **Solving ODEs arising from non-selfadjoint Hamiltonian eigenproblems**. *Advances in Computational Mathematics* 13:3 (2000), 231–256. DOI: 10.1023/A:1018954227133.

[56] KV Khmelnytskaya, VV Kravchenko, and JA Baldenebro-Obeso. **Spectral parameter power series for fourth-order Sturm–Liouville problems**. *Applied mathematics and computation* 219:8 (2012), 3610–3624. DOI: 10.1016/j.amc.2012.09.055.

[57] VV Kravchenko. **Direct and inverse Sturm-Liouville problems: A method of solution**. Springer Nature, 2020.

[58] VV Kravchenko. **On a method for solving the inverse Sturm–Liouville problem**. *Journal of Inverse and Ill-posed Problems* 27:3 (2019), 401–407. DOI: 10.1515/jiip-2018-0045.

[59] C Lanczos. **An iteration method for the solution of the eigenvalue problem of linear differential and integral operators**. United States Governm. Press Office Los Angeles, CA, 1950.

[60] V Ledoux, MV Daele, and GV Berghe. **Efficient numerical solution of the one-dimensional Schrodinger eigenvalue problem using Magnus integrators**. *IMA Journal of Numerical Analysis* 30:3 (2009), 751–776. DOI: 10.1093/imanum/drn062.

[61] V Ledoux, MV Daele, and GV Berghe. **MATSLISE: A MATLAB package for the numerical solution of Sturm-Liouville and Schrödinger equations**. *ACM Transactions on Mathematical Software (TOMS)* 31:4 (2005), 532–554. DOI: 10.1145/1114268.1114273.

[62] D Lesnic and BS Attili. **An efficient method for sixth-order Sturm-Liouville problems**. *International Journal of Science & Technology* 2:2 (2007), 109–114.

[63] N Levinson. **The Inverse Sturm-Liouville Problem**. *Matematisk Tidsskrift. B* (1949), 25–30. URL: http://www.jstor.org/stable/24527827.

[64] BM Levitan. **Inverse Sturm-Liouville Problems**. Walter de Gruyter GmbH & Co KG, 2018.

[65] BM Levitan, IS Sargsian, and Sargsjan. **Introduction to spectral theory: self-adjoint ordinary differential operators: Selfadjoint Ordinary Differential Operators**. Vol. 39. American Mathematical Soc., 1975.

[66] S Lie. **Theorie der transformationsgruppen**. Vol. 1. Leipzig, B.G. Teubner, 1970.

[67] C-S Liu. **Solving an inverse Sturm-Liouville problem by a Lie-group method**. *Boundary Value Problems* 2008:1 (2008), 749865. DOI: 10.1155/2008/749865.

[68] BD Lowe, M Pilant, and W Rundell. **The recovery of potentials from finite spectral data**. *SIAM journal on mathematical analysis* 23:2 (1992), 482–504. DOI: 10.1137/0523023.

[69] W Magnus. **On the exponential solution of differential equations for a linear operator**. *Communications on pure and applied mathematics* 7:4 (1954), 649–673. DOI: 10.1002/cpa.3160070404.

[70] N Mai-Duy. **An effective spectral collocation method for the direct solution of high-order ODEs**. *Communications in numerical methods in engineering* 22:6 (2006), 627–642. DOI: 10.1002/cnm.841.

[71] V Makarov and N Romaniuk. **Exponentially convergent symbolic algorithm of the functional-discrete method for the fourth order Sturm–Liouville problems with polynomial coefficients**. *Journal of Computational and Applied Mathematics* 358 (2019), 405–423. DOI: 10.1016/j.cam.2019.03.024.

[72] VL Makarov and YV Klymenko. **Application of the FD-method to the solution of the Sturm-Liouville problem with coefficients of special form**.

Ukrainian Mathematical Journal 59:8 (2007), 1264–1273. DOI: 10.1007/s11253-007-0086-0.

[73] V Marčenko. **Sturm-Liouville operators and their applications**. *Izd. Nauk. Dumka, Kiev* 8 (1977).

[74] M Marletta and JD Pryce. **Automatic solution of Sturm-Liouville problems using the Pruess method**. *Journal of Computational and Applied Mathematics* 39:1 (1992), 57–78. DOI: 10.1016/0377-0427(92)90222-J.

[75] **Mathematica**. Wolfram Research, Inc., 2016.

[76] **Matlab**. The MathWorks, Inc., 2014.

[77] B McKeeman. *MATLAB Performance Measurement - File Exchange - MATLAB Central*. Ed. by 1994-2019 The MathWorks, Inc. 2008. URL: https://in.mathworks.com/matlabcentral/fileexchange/18510-matlab-performance-measurement.

[78] JR McLaughlin. **An Inverse Eigenvalue Problem of Order Four**. *SIAM Journal on Mathematical Analysis* 7:5 (1976), 646–661. DOI: 10.1137/0507050.

[79] JR McLaughlin. **An Inverse Eigenvalue Problem of Order Four—An Infinite Case**. *SIAM Journal on Mathematical Analysis* 9:3 (1978), 395–413. DOI: 10.1137/0509026.

[80] JR McLaughlin. **Analytical methods for recovering coefficients in differential equations from spectral data**. *SIAM review* 28:1 (1986), 53–72. DOI: 10.1137/1028003.

[81] JR McLaughlin and W Rundell. **A uniqueness theorem for an inverse Sturm–Liouville problem**. *Journal of mathematical physics* 28:7 (1987), 1471–1472. DOI: 10.1063/1.527500.

[82] QM Al-Mdallal and MI Syam. **The Chebyshev collocation-path following method for solving sixth-order Sturm–Liouville problems**. *Applied Mathematics and Computation* 232 (2014), 391–398. DOI: 10.1016/j.amc.2014.01.083.

[83] H Mirzaei. **Computing the eigenvalues of fourth order Sturm-Liouville problems with Lie Group method**. *Iranian Journal of Numerical Analysis and Optimization* 7:1 (2017), 1–12. DOI: 10.22067/ijnao.v7i1.44788.

[84] PC Moan. **Efficient approximation of Sturm-Liouville problems using Lie-group methods**. *University of Cambridge Department of Applied Mathematics & Theoretical Physics–Report–DAMPT NA* (1998).

[85] C Moler and C Van Loan. **Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later**. *SIAM review* 45:1 (2003), 3–49. DOI: 10.1137/S00361445024180.

[86] Mark A Naimark. **Linear differential operators. Part I: Elementary theory of linear differential operators**. Frederick Ungar Publishing Company, 1967.

[87] Mark A Naimark. **Linear differential operators. Part II: Linear differential operators in Hilbert space**. Frederick Ungar Publishing Company, 1968.

[88] A Neamaty, Sh Akbarpoor, and E Yilmaz. **Solving Symmetric Inverse Sturm–Liouville Problem Using Chebyshev Polynomials**. *Mediterranean Journal of Mathematics* 16:3 (2019), 74. DOI: 10.1007/s00009-019-1330-1.

[89] M Neher. **Enclosing solutions of an inverse Sturm-Liouville problem with finite data**. *Computing* 53:3 (1994), 379–395. DOI: 10.1007/BF02307388.

[90] J Paine. **A numerical method for the inverse Sturm–Liouville problem**. *SIAM Journal on scientific and statistical computing* 5:1 (1984), 149–156. DOI: 10.1137/0905011.

[91] JW Paine, FR de Hoog, and RS Anderssen. **On the correction of finite difference eigenvalue approximations for Sturm-Liouville problems**. *Computing* 26:2 (1981), 123–139. DOI: 10.1007/BF02241779.

[92] U Perera and C Böckmann. **Solutions of Direct and Inverse Even-Order Sturm-Liouville Problems Using Magnus Expansion**. *Mathematics* 7:6 (2019). DOI: 10.3390/math7060544.

[93] U Perera and C Böckmann. **Solutions of Sturm-Liouville Problems**. *Mathematics* 8:11 (2020), 2074. DOI: 10.3390/math8112074.

[94] S Pruess and CT Fulton. **Mathematical software for Sturm-Liouville problems**. *ACM Transactions on Mathematical Software (TOMS)* 19:3 (1993), 360–376. DOI: 10.1145/155743.155791.

[95] JD Pryce. **A Test Package for Sturm-Liouville Solvers**. *ACM Transactions on Mathematical Software (TOMS)* 25 (1 1999), 21–57. DOI: 10.1145/305658.287651.

[96] M Rafler and C Böckmann. **Reconstruction method for inverse Sturm–Liouville problems with discontinuous potentials**. *Inverse Problems* 23:3 (2007), 933–946. DOI: 10.1088/0266-5611/23/3/006.

[97] A Rattana and C Böckmann. **Matrix methods for computing eigenvalues of Sturm–Liouville problems of order four**. *Journal of Computational and Applied Mathematics* 249 (2013), 144–156. DOI: 10.1016/j.cam.2013.02.024.

[98] N Röhrl. **A least-squares functional for solving inverse Sturm–Liouville problems**. *Inverse Problems* 21:6 (Nov. 2005), 2009–2017. DOI: 10.1088/0266-5611/21/6/013.

[99] W Rundell and PE Sacks. **Reconstruction techniques for classical inverse Sturm-Liouville problems**. *Mathematics of Computation* 58:197 (1992), 161–183. DOI: 10.1090/S0025-5718-1992-1106979-0.

[100] PE Sacks. **An iterative method for the inverse Dirichlet problem**. *Inverse Problems* 4:4 (Oct. 1988), 1055–1069. DOI: 10.1088/0266-5611/4/4/009.

[101] AWI Schueller. **Eigenvalue asymptotics for self-adjoint, fourth-order, ordinary differential operators.** PhD thesis. University of Kentucky, Lexington, 1997.

[102] HI Siyyam and MI Syam. **An Efficient Technique for Finding the Eigenvalues of Sixth-Order Sturm-Liouville Problems**. *Applied Mathematical Sciences* 5:49 (2011), 2425–2436.

[103] MI Syam and HI Siyyam. **An efficient technique for finding the eigenvalues of fourth-order Sturm–Liouville problems**. *Chaos, Solitons & Fractals* 39:2 (2009), 659–665. DOI: 10.1016/j.chaos.2007.01.105.

[104] AHS Taher and A Malek. **An efficient algorithm for solving high order Sturm-Liouville problems using variational iteration method**. *Fixed Point Theory* 14 (2013), 193–210.

[105] AHS Taher, A Malek, and SH Momeni-Masuleh. **Chebyshev differentiation matrices for efficient computation of the eigenvalues of fourth-order Sturm–Liouville problems**. *Applied Mathematical Modelling* 37:7 (2013), 4634–4642. DOI: 10.1016/j.apm.2012.09.062.

[106] H Taşeli. **Accurate numerical bounds for the spectral points of singular Sturm–Liouville problems over $-\infty < x < \infty$**. *Journal of Computational and Applied Mathematics* 115:1 (2000), 535–546. DOI: 10.1016/S0377-0427(99)00302-7.

[107] AN Tikhonov. **On the uniqueness of the solution of the problem of electromagnetic sounding**. In: *Dokl. Akad. Nauk SSSR*. Vol. 69. 1949, 797–800.

[108] RA Usmani. **Some new finite difference methods for computing eigenvalues of two-point boundary-value problems**. *Computers & mathematics with applications* 11:9 (1985), 903–909. DOI: 10.1016/0898-1221(85)90093-8.

[109] RA Usmani and RP Agarwal. **New symmetric finite difference methods for computing eigenvalues of a boundary-value problem**. *Communications in applied numerical methods* 1:6 (1985), 305–309. DOI: 10.1002/cnm.1630010609.

[110] G Vanden Berghe and H De Meyer. **A modified Numerov method for higher Sturm-Liouville eigenvalues**. *International Journal of Computer Mathematics* 37:1-2 (1990), 63–77. DOI: 10.1080/00207169008803935.

[111] U Yücel. **Approximations of Sturm–Liouville eigenvalues using differential quadrature (DQ) method**. *Journal of Computational and Applied Mathematics* 192:2 (2006), 310–319. DOI: 10.1016/j.cam.2005.05.008.

[112] U Yücel and K Boubaker. **Differential quadrature method (DQM) and Boubaker polynomials expansion scheme (BPES) for efficient computation of the eigenvalues of fourth-order Sturm–Liouville problems**. *Applied Mathematical Modelling* 36:1 (2012), 158–167. DOI: 10.1016/j.apm.2011.05.030.

[113] X Zhang. **Mapped barycentric Chebyshev differentiation matrix method for the solution of regular Sturm–Liouville problems**. *Applied Mathematics and Computation* 217:5 (2010), 2266–2276. DOI: 10.1016/j.amc.2010.07.027.

[114] Z Zhang. **How Many Numerical Eigenvalues Can We Trust?** *Journal of Scientific Computing* 65:2 (2015), 455–466. DOI: 10.1007/s10915-014-9971-5.

List of Publications

Articles in Refereed Journals

[1] **Biological aging modeled with stochastic differential equations**. *Communications in Applied Analysis* 22:2 (2018), 271–293. DOI: 10.12732/caa.v22i2.8. Joint work with EJ Allen.

[2] **Solutions of Direct and Inverse Even-Order Sturm-Liouville Problems Using Magnus Expansion**. *Mathematics* 7:6 (2019). DOI: 10.3390/math7060544. Joint work with C Böckmann.

[3] **Solutions of Sturm-Liouville Problems**. *Mathematics* 8:11 (2020), 2074. DOI: 10.3390/math8112074. Joint work with C Böckmann.

[4] **Assessing the Impact of S&P SL20 Index Construction on Listed Companies in Colombo Stock Exchange (CSE)**. *International Journal of Economics and Finance* 8:7 (2016), 159–159. DOI: 10.5539/ijef.v8n7p159. Joint work with R Dissanayake and M Jayasundara.

[5] **Eisenberg's Duality in Homogeneous Programming, Shephard's Duality and Economic Analysis**. *Metroeconomica* 68:4 (2017), 816–832. DOI: 10.1111/meca.12144. Joint work with T Fujimoto.

[6] **A proof of the Farkas–Minkowski theorem by a tandem method**. *Metroeconomica* 69:1 (2018), 142–150. DOI: 10.1111/meca.12173. Joint work with T Fujimoto and G Giorgi.

[7] **Nonlinear Generalizations of Farkas-Minkowski's Theorem**. *The Kagawa University Economic Review* 88:1 (2015), 1–17. Joint work with T Fujimoto, G Giorgi, and RR Ranade.

[8] **Relationship between journal-ranking metrics for a multidisciplinary set of journals**. *portal: Libraries and the Academy* 18:1 (2018), 35–58. DOI: 10.1353/pla.2018.0003. Joint work with M Wijewickrema.