A Matlab Spectral Integration Suite

Gokul Hariharan,[1] Mihailo R. Jovanović,[2] and Satish Kumar[1]

[1] *Department of Chemical Engineering and Materials Science, University of Minnesota, Minneapolis, MN 55455, USA*

[2] *Ming Hsieh Department of Electrical and Computer Engineering, University of Southern California, Los Angeles, CA 90089, USA*

I. INTRODUCTION

We develop a customized Matlab spectral integration suite that is based on the method described by Greengard [6] and Du [4]. In order to facilitate application to nth order differential equations, our implementation introduces minor modifications that we describe next.

I.1. An example: A reaction-diffusion equation

Let us consider a second-order linear differential equation,

$$\mathrm{D}^2\phi(y) \,-\, \epsilon^2\phi(y) \;=\; d(y), \tag{1a}$$

with either homogeneous Dirichlet,

$$\phi(\pm 1) \;=\; 0, \tag{1b}$$

or Neumann boundary conditions,

$$\mathrm{D}\,\phi(\pm 1) \;=\; 0. \tag{1c}$$

Here, ϕ is the field of interest, d is an input, $\epsilon \in \mathbb{R}$ is a given constant, $y \in [-1, 1]$, and $\mathrm{D} := \mathrm{d}/\mathrm{d}y$.

In the spectral integration method, the highest derivative in a differential equation is expressed in a basis of Chebyshev polynomials. In particular, for equation (1a),

$$\mathrm{D}^2\phi(y) \;=\; {\sum_{i\,=\,0}^{\infty}}' \phi_i^{(2)} T_i(y) \;=:\; \mathbf{t}_y^T \mathbf{\Phi}, \tag{2}$$

where $\sum'$ denotes a summation with the first term halved, $\mathbf{\Phi} := [\,\phi_0^{(2)} \;\; \phi_1^{(2)} \;\; \phi_2^{(2)} \;\; \cdots\;]^T$ is the infinite vector of spectral coefficients $\phi_i^{(2)}$, and $\mathbf{t}_y$ is the vector of Chebyshev polynomials of the first kind $T_i(y)$,

$$\mathbf{t}_y^T \;:=\; \left[\, \tfrac{1}{2}T_0(y) \;\; T_1(y) \;\; T_2(y) \;\; \cdots \;\right]. \tag{3}$$

Integration of (2) in conjunction with the recurrence relations for integration of Chebyshev polynomials is used to determine spectral coefficients corresponding to lower derivatives of ϕ. For example, indefinite integration of (2) yields

$$\mathrm{D}\,\phi(y) \;=\; {\sum_{i\,=\,0}^{\infty}}' \phi_i^{(1)} T_i(y) \,+\, c_1 \;=:\; \mathbf{t}_y^T \mathbf{\Phi}^{(1)} \,+\, c_1, \tag{4}$$

where c_1 is a constant of integration and

$$\phi_i^{(1)} \;=\; \begin{cases} \frac{1}{2}\,\phi_1^{(2)}, & i \,=\, 0, \\ \frac{1}{2i}(\phi_{i-1}^{(2)} \,-\, \phi_{i+1}^{(2)}), & i \,\geq\, 1. \end{cases} \tag{5}$$

These expressions for $\phi_i^{(1)}$ are derived in Appendix A.

Similarly, indefinite integration of $\mathrm{D}\,\phi$ allows us to express ϕ as

$$\phi(y) \;=\; {\sum_{i\,=\,0}^{\infty}}{}' \,\phi_i^{(0)} T_i(y) \,+\, \tilde{c}_0 \,+\, c_1 y \;=:\; \mathbf{t}_y^T \mathbf{\Phi}^{(0)} \,+\, \tilde{c}_0 \,+\, c_1 y,$$

where $\tilde{c}_0$ and c_1 are integration constants and

$$\phi_i^{(0)} \;=\; \begin{cases} \frac{1}{2}\,\phi_1^{(1)}, & i \;=\; 0, \\ \frac{1}{2i}(\phi_{i-1}^{(1)} \,-\, \phi_{i+1}^{(1)}), & i \;\geq\; 1. \end{cases}$$

Equation (5) provides a recursive relation that is used to determine spectral coefficients of lower derivatives from the spectral coefficients of the highest derivative of the variable ϕ,

$$\mathbf{\Phi}^{(1)} \;=\; \mathbf{J}\mathbf{\Phi}, \;\; \mathbf{\Phi}^{(0)} \;=\; \mathbf{J}^2\mathbf{\Phi},$$

where $\mathbf{J}^2 := \mathbf{J}\mathbf{J}$, and

$$\mathbf{J} \;:=\; \begin{bmatrix} 0 & \frac{1}{2} & 0 & \cdots & & & & \\ \frac{1}{2} & 0 & -\frac{1}{2} & 0 & \cdots & & & \\ 0 & \frac{1}{4} & 0 & -\frac{1}{4} & 0 & \cdots & & \\ 0 & 0 & \frac{1}{6} & 0 & -\frac{1}{6} & 0 & \cdots & \\ \vdots & \vdots & & \ddots & \ddots & \ddots & & \end{bmatrix}. \tag{6}$$

Since $T_0(y) = 1$ and $T_1(y) = y$, we let $c_0 := 2\,\tilde{c}_0$ and represent integration constants in the basis expansion of u and $\mathrm{D}\,u$ in terms of Chebyshev polynomials,

$$u(y) \;=\; \mathbf{t}_y^T \mathbf{J}^2 \mathbf{u} \,+\, \begin{bmatrix} \frac{1}{2}T_0(y) & T_1(y) \end{bmatrix} \overbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}^{\mathbf{K}^0} \begin{bmatrix} c_0 \\ c_1 \end{bmatrix}, \tag{7}$$

$$\mathrm{D}\,u(y) \;=\; \mathbf{t}_y^T \mathbf{J} \mathbf{u} \,+\, \begin{bmatrix} \frac{1}{2}T_0(y) & T_1(y) \end{bmatrix} \underbrace{\begin{bmatrix} 0 & 2 \\ 0 & 0 \end{bmatrix}}_{\mathbf{K}^1} \begin{bmatrix} c_0 \\ c_1 \end{bmatrix}. \tag{8}$$

By introducing the vector of integration constants $\mathbf{c} := \begin{bmatrix} c_0 & c_1 \end{bmatrix}^T$, we can represent u, $\mathrm{D}u$, and $\mathrm{D}^2 u$ as

$$u(y) \;=\; \mathbf{t}_y^T (\mathbf{J}^2 \mathbf{u} \,+\, \mathbf{R}_2 \mathbf{c}), \tag{9}$$

$$\mathrm{D}u(y) \;=\; \mathbf{t}_y^T (\mathbf{J}^1 \mathbf{u} \,+\, \mathbf{R}_1 \mathbf{c}), \tag{10}$$

$$\mathrm{D}^2 u(y) \;=\; \mathbf{t}_y^T (\mathbf{J}^0 \mathbf{u} \,+\, \mathbf{R}_0 \mathbf{c}), \tag{11}$$

where $\mathbf{J}^0 = \mathbf{I}$ is an infinite identity matrix, and

$$\mathbf{R}_i \;:=\; \begin{bmatrix} \mathbf{K}^{2-i} \\ \mathbf{0} \end{bmatrix}, \;\; i \;=\; 0, 1, 2,$$

are matrices with an infinite number of rows and two columns, and $\mathbf{K}^2 = \mathbf{K}\,\mathbf{K}$ results in a 2×2 zero matrix. Thus, in the basis of Chebyshev polynomials, we can express differential equation (1a) without its boundary conditions as

$$\mathbf{t}_y^T \big((\mathbf{I} \,-\, k^2 \mathbf{J}^2) \mathbf{u} \,+\, (\mathbf{R}_0 \,-\, k^2 \mathbf{R}_2) \mathbf{c} \big) \;=\; \mathbf{t}_y^T \mathbf{f}, \tag{12}$$

where $\mathbf{f}$ is the vector of spectral coefficients associated with the input f in (1a). Using (9), we can write

Dirichlet boundary conditions (1b) as

$$\begin{aligned} \mathbf{t}_{+1}^T\,\mathbf{J}^2\,\mathbf{u} \;+\; \mathbf{t}_{+1}^T\,\mathbf{R}_2\,\mathbf{c} \;&=\; 0, \\ \mathbf{t}_{-1}^T\,\mathbf{J}^2\,\mathbf{u} \;+\; \mathbf{t}_{-1}^T\,\mathbf{R}_2\,\mathbf{c} \;&=\; 0, \end{aligned} \tag{13}$$

and the use of (10) brings Neumann boundary conditions (1c) into,

$$\begin{aligned} \mathbf{t}_{+1}^T\,\mathbf{J}\,\mathbf{u} \;+\; \mathbf{t}_{+1}^T\,\mathbf{R}_1\,\mathbf{c} \;&=\; 0, \\ \mathbf{t}_{-1}^T\,\mathbf{J}\,\mathbf{u} \;+\; \mathbf{t}_{-1}^T\,\mathbf{R}_1\,\mathbf{c} \;&=\; 0. \end{aligned} \tag{14}$$

Hence, the infinite-dimensional representation of differential equation (1a) with Dirichlet boundary conditions is given by

$$\begin{bmatrix} \mathbf{I} - k^2\,\mathbf{J}^2 & \mathbf{R}_0 - k^2\,\mathbf{R}_2 \\ \mathbf{t}_{+1}^T\,\mathbf{J}^2 & \mathbf{t}_{+1}^T\,\mathbf{R}_2 \\ \mathbf{t}_{-1}^T\,\mathbf{J}^2 & \mathbf{t}_{-1}^T\,\mathbf{R}_2 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{c} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \\ 0 \end{bmatrix}, \tag{15}$$

and the representation for Neumann boundary conditions is obtained by replacing the last two rows in (15) with those in (14). We obtain the finite-dimensional approximation from (15) by truncating the infinite spectral coefficients to $N+1$ spectral coefficients using the projection operator [7, Eq. 2.9, also last paragraph of Section 2.4],

$$\mathbf{P} \;=\; [\,\mathbf{I}_{N+1} \;\; \mathbf{0}\,], \tag{16}$$

where $\mathbf{I}_{N+1}$ is an identity matrix of dimension $N+1$, and $\mathbf{P}$ is a matrix with $N+1$ rows and infinite columns so that

$$\begin{aligned} &\begin{bmatrix} \mathbf{P} & 0 \\ 0 & \mathbf{I}_2 \end{bmatrix} \begin{bmatrix} \mathbf{I} - k^2\,\mathbf{J}^2 & \mathbf{R}_0 - k^2\,\mathbf{R}_2 \\ \mathbf{t}_{+1}^T\,\mathbf{J}^2 & \mathbf{t}_{+1}^T\mathbf{R}_2 \\ \mathbf{t}_{-1}^T\,\mathbf{J}^2 & \mathbf{t}_{-1}^T\mathbf{R}_2 \end{bmatrix} \begin{bmatrix} \mathbf{P}^T & 0 \\ 0 & \mathbf{I}_2 \end{bmatrix} \begin{bmatrix} \mathbf{P} & 0 \\ 0 & \mathbf{I}_2 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{c} \end{bmatrix} = \begin{bmatrix} \mathbf{P} & 0 \\ 0 & \mathbf{I}_2 \end{bmatrix} \begin{bmatrix} \mathbf{f} \\ 0 \\ 0 \end{bmatrix}, \\ &\Rightarrow \begin{bmatrix} \mathbf{P}\left(\mathbf{I} - k^2\,\mathbf{J}^2\right)\mathbf{P}^T & \mathbf{P}\,\left(\mathbf{R}_0 - k^2\,\mathbf{R}_2\right) \\ \mathbf{t}_{+1}^T\,\mathbf{J}^2\,\mathbf{P}^T & \mathbf{t}_{+1}^T\mathbf{R}_2 \\ \mathbf{t}_{-1}^T\,\mathbf{J}^2\,\mathbf{P}^T & \mathbf{t}_{-1}^T\mathbf{R}_2 \end{bmatrix} \begin{bmatrix} \hat{\mathbf{u}} \\ \mathbf{c} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{f}} \\ 0 \\ 0 \end{bmatrix}. \end{aligned} \tag{17}$$

The variables with a cap represent the finite-dimensional truncation of spectral coefficients, e.g., $\mathbf{P}\,\mathbf{u} = \hat{\mathbf{u}}$.

Notice in the left-bottom corner in the matrix in (17) has a matrix multiplication of $\mathbf{t}_{-1}^T\,\mathbf{J}^2\,\mathbf{P}^T$, which is a quantity that involves multiplying matrices of dimensions $(1\times\infty)\times(\infty\times\infty)\times(\infty\times(N+1)\,)$ which results in a vector of size $1\times(N+1)$. The product $(1\times\infty)\times(\infty\times\infty)$ is needed to ensure proper truncation, see [7, last paragraph of Section 2.4]. For a practical implementation some large value greater than $N+1$ works in the place of a matrix dimension of ∞. In our codes we use $N+1+2n$, where n is the order of the differential equation (e.g., for the reaction-diffusion equation (1a) we use $N+1+4 = N+5$) as the ∞ as we found that results are the same if we use a value greater than $N+1+2n$.

I.2. Eigenvalues

We now consider the eigenvalues of the transient reaction-diffusion equation with Neumann boundary conditions,

$$\mathrm{D}^2 u(y) \;-\; k^2 u(y) \;=\; \lambda\, u(y), \tag{18a}$$

$$\mathrm{D}\,u(\pm) \;=\; 0. \tag{18b}$$

Using the relations from (9) and (11), the differential equation (18a) is expressed in an infinite Chebyshev basis as,

$$\mathbf{t}_y^T \, \left(\left(\mathbf{I} \,-\, k^2 \mathbf{J}^2 \right) \mathbf{u} \,+\, \left(\mathbf{R}_0 - k^2 \, \mathbf{R}_2 \right) \mathbf{c} \right) \;=\; \lambda \, \mathbf{t}_y^T \, \left(\mathbf{J}^2 \, \mathbf{u} + \mathbf{R}_2 \, \mathbf{c} \right). \tag{19}$$

The boundary conditions are the same as (14). The infinite-dimensional representation from equating terms of the same basis and appending boundary conditions is given by

$$\begin{bmatrix} \mathbf{I} \,-\, k^2 \, \mathbf{J}^2 & \mathbf{R}_0 \,-\, k^2 \, \mathbf{R}_2 \\ \mathbf{t}_{+1}^T \, \mathbf{J}^1 & \mathbf{t}_{+1}^T \, \mathbf{R}_1 \\ \mathbf{t}_{-1}^T \, \mathbf{J}^1 & \mathbf{t}_{-1}^T \, \mathbf{R}_1 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{c} \end{bmatrix} \;=\; \lambda \begin{bmatrix} \mathbf{J}^2 & \mathbf{R}_2 \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{c} \end{bmatrix}. \tag{20}$$

The differential equation in (18a) has an eigenvalue λ, but the boundary conditions in (18b) do not. Thus the eigenvalue problem with padded zeros as on the right-hand side of (20) can result in extra eigenvalues that increase in value with an increase in N in certain problems [1]. This zero padding in (20) can be removed by using the procedure described in § IIIB.2 in the accompanying paper. The projection operator (16) is used on (20) to reduce it to a finite dimensional approximation in the same way as (17). Eigenvalues of (20) are computed in Code 1.

I.3. Frequency responses

The system for frequency responses of the reaction diffusion equation is given by (see § IIB in the accompanying paper),

$$\left(i\omega I \,-\, \mathrm{D}^2 \,+\, \epsilon^2 I \right) u(y) \;=\; d(y), \tag{21a}$$

$$\mathrm{D}\, u(\pm 1) \;=\; 0, \tag{21b}$$

and the adjoint system by,

$$\left(-i\omega I \,-\, \mathrm{D}^2 \,+\, \epsilon^2 I \right) v(y) \;=\; g(y), \tag{21c}$$

$$\mathrm{D}\, v(\pm 1) \;=\; 0. \tag{21d}$$

The feedback interconnected system to compute frequency responses is given by (see § IIIA in the accompanying paper),

$$\begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} \begin{bmatrix} u(y) \\ v(y) \end{bmatrix} \;=\; \lambda \begin{bmatrix} (i\omega + \epsilon^2) I \,-\, \mathrm{D}^2 & 0 \\ 0 & (-i\omega + \epsilon^2) I \,-\, \mathrm{D}^2 \end{bmatrix} \begin{bmatrix} u(y) \\ v(y) \end{bmatrix}, \tag{22a}$$

$$\begin{bmatrix} \mathcal{Q}(+1, \mathrm{D}) & 0 \\ \mathcal{Q}(-1, \mathrm{D}) & 0 \\ 0 & \mathcal{Q}(+1, \mathrm{D}) \\ 0 & \mathcal{Q}(-1, \mathrm{D}) \end{bmatrix} \begin{bmatrix} u(y) \\ v(y) \end{bmatrix} \;=\; 0, \tag{22b}$$

where $\mathcal{Q}(a, L)$ is an evaluation operator that evaluates the action of the linear operator L on a variable at a point $y = a$, and (22b) specifies homogeneous Neumann boundary conditions on $u(y)$ and $v(y)$ (see (21)).

The expression for the identity operators in (22) is given in (9), and for the second derivative operators in (10). The infinite-dimensional representation for the differential equation of this system is given by,

$$
\begin{aligned}
&\begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{J}^2 & \mathbf{R}_2 \\ \mathbf{J}^2 & \mathbf{R}_2 & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{c}^{(u)} \\ \mathbf{v} \\ \mathbf{c}^{(v)} \end{bmatrix} = \\
&\lambda \begin{bmatrix} (i\omega + \epsilon^2)\mathbf{J}^2 - \mathbf{I} & (i\omega + \epsilon^2)\mathbf{R}_2 - \mathbf{R}_0 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & (-i\omega + \epsilon^2)\mathbf{J}^2 - \mathbf{I} & (-i\omega + \epsilon^2)\mathbf{R}_2 - \mathbf{R}_0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{c}^{(u)} \\ \mathbf{v} \\ \mathbf{c}^{(v)} \end{bmatrix},
\end{aligned}
\tag{23a}
$$

where $\mathbf{c}^{(u)}$ and $\mathbf{c}^{(v)}$ are the vector of constants of integration corresponding to $u(y)$ and $v(y)$ respectively. The expressions for Neumann boundary conditions follow from (14), and are given by

$$
\begin{bmatrix} \mathbf{t}_{+1}^T\,\mathbf{J} & \mathbf{t}_{+1}^T\,\mathbf{R}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{t}_{-1}^T\,\mathbf{J} & \mathbf{t}_{-1}^T\,\mathbf{R}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{t}_{+1}^T\,\mathbf{J} & \mathbf{t}_{+1}^T\,\mathbf{R}_1 \\ \mathbf{0} & \mathbf{0} & \mathbf{t}_{-1}^T\,\mathbf{J} & \mathbf{t}_{-1}^T\,\mathbf{R}_1 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{c}^{(u)} \\ \mathbf{v} \\ \mathbf{c}^{(v)} \end{bmatrix} = \mathbf{0}. \tag{23b}
$$

The finite-dimensional projection of the infinite-dimensional representation for the differential equation (23a) and boundary conditions (23b) is obtained using (16) in the same manner as (17). The finite-dimensional approximation to (23) is solved to compute $\lambda = \pm\sigma$, and $\sigma_{\max}$ is the resolvent norm of (21). The last part of Code 1 solves for frequency responses of this system.

II. ARBITRARY ORDER LINEAR DIFFERENTIAL EQUATION WITH NON-CONSTANT COEFFICIENTS

In the previous section we considered the reaction-diffusion equation, and discussed how spectral integration is used to solve for a forcing, for eigenvalues, and for frequency responses. In this section, we illustrate this process for an nth order linear differential equation with non-constant coefficients. Steps that follow are similar to what was discussed in § I.

Consider a general representation of an nth order linear differential equation with non-constant coefficients,

$$
\sum_{k=0}^{n} a^{(k)}(y)\,\mathrm{D}^k u(y) \;=\; f(y), \tag{24a}
$$

$$
\sum_{k=0}^{n-1} b^{(k,\mathbf{p})}\mathrm{D}^k\, u(\mathbf{p}) \;=\; \mathbf{q}, \tag{24b}
$$

where $a^{(k)}$ are the non-constant coefficients, and f is an input, $b^{(k,\mathbf{p})}$ are constant coefficients associated with boundary constraints (a general case of mixed boundary conditions), at a vector of evaluation points, $\mathbf{p}$, and corresponding values at the boundaries, $\mathbf{q}$.

II.1. Differential equation

In the same manner as the second derivative in (2) for the reaction-diffusion equation (1a), the highest derivative of the variable $u(y)$ in (24a) is expressed in a basis of Chebyshev polynomials as

$$
\mathrm{D}^n u(y) \;=\; {\sum_{i=0}^{\infty}}' u_i^{(n)} T_i(y) \;=:\; \mathbf{t}_y^T\mathbf{u}, \tag{25}
$$

where, $\mathbf{u} = [\, u_0^{(n)} \;\; u_1^{(n)} \;\; \cdots \;\; u_\infty^{(n)} \,]^T$. The lower derivatives are expressed as,

$$\mathrm{D}^i\, u(y) \;=\; \mathbf{t}_y^T \left(\mathbf{J}^{n-i}\mathbf{u} \,+\, \mathbf{R}_{n-i}\, \mathbf{c} \right), \tag{26}$$

where, $\mathbf{J}$ is defined in (6), $\mathbf{c} = [\, c_0 \;\; c_1 \;\; \cdots \;\; c_{n-1} \,]^T$ are the n constants of integration that result from integrating the highest derivative (25), and

$$\mathbf{R}_i \;:=\; \begin{bmatrix} \mathbf{K}^{n-i} \\ \mathbf{0} \end{bmatrix},$$

are matrices with n columns and infinite number of rows, where [6, Eq. 10]

$$\mathbf{K} = \begin{bmatrix} 0 & 2 & 0 & 6 & 0 & 10 & \cdots \\ 0 & 0 & 4 & 0 & 8 & 0 & \ddots \\ 0 & 0 & 0 & 6 & 0 & 10 & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}, \tag{27}$$

is a matrix of dimension $n \times n$.

II.2. Spatially varying coefficients

The multiplication operator for the product of two Chebyshev series is used to account for non-constant coefficients $a^{(k)}$ in (24a). For a function $a(y)$ in the basis of Chebyshev polynomials,

$$a(y) \;=\; {\sum_{i=0}^{\infty}}{}' \; a_i\, T_i(y), \tag{28}$$

the multiplication operator is given by [4, Section 3]

$$\mathcal{M}[a] \;=\; \frac{1}{2} \begin{bmatrix} a_0 & a_1 & a_2 & a_3 & \cdots \\ a_1 & a_0 & a_1 & a_2 & \ddots \\ a_2 & a_1 & a_0 & a_1 & \ddots \\ a_3 & a_2 & a_1 & a_0 & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix} \;+\; \frac{1}{2} \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots \\ a_1 & a_2 & a_3 & a_4 & \cdots \\ a_2 & a_3 & a_4 & a_5 & \ddots \\ a_3 & a_4 & a_5 & a_6 & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}. \tag{29}$$

II.3. Infinite-dimensional representation

The differential equation in (24a) is expressed in a Chebyshev basis using (26) and (29) as

$$\mathbf{t}_y^T \left(\left[\sum_{k=0}^{n} \mathcal{M}[a^{(k)}]\; \mathbf{J}^{n-k} \right] \mathbf{u} \;+\; \left[\sum_{k=0}^{n-1} \mathcal{M}[a^{(k)}]\; \mathbf{R}_{n-k} \right] \mathbf{c} \right) \;=\; \mathbf{t}_y^T\, \mathbf{f}, \tag{30}$$

and the boundary conditions (24b) using (26) as

$$\mathbf{t}_{\mathbf{p}}^T \left(\left[\sum_{k=0}^{n-1} b^{(k,\mathbf{p})}\; \mathbf{J}^{n-k} \right] \mathbf{u} \;+\; \left[\sum_{k=0}^{n-1} b^{(k,\mathbf{p})}\; \mathbf{R}_{n-k} \right] \mathbf{c} \right) \;=\; \mathbf{q}. \tag{31}$$

Thus the infinite-dimensional representation based on equating the terms of the same basis for the differential equation (24) using (30) and appending boundary conditions in (31) is given by,

$$\underbrace{\begin{bmatrix} \sum_{k=0}^{n} \mathcal{M}[a^{(k)}]\, \mathbf{J}^{n-k} & \sum_{k=0}^{n-1} \mathcal{M}[a^{(k)}]\, \mathbf{R}_{n-k} \\ \sum_{k=0}^{n-1} b^{(k,\mathbf{p})}\, \mathbf{t}_{\mathbf{p}}^{T}\, \mathbf{J}^{n-k} & \sum_{k=0}^{n-1} b^{(k,\mathbf{p})}\, \mathbf{t}_{\mathbf{p}}^{T}\, \mathbf{R}_{n-k} \end{bmatrix}}_{\mathbf{M}} \begin{bmatrix} \mathbf{u} \\ \mathbf{c} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{q} \end{bmatrix}. \tag{32}$$

II.4. Finite-dimensional approximation

We use the projection operator (16) to truncate the infinite spectral coefficients in $\mathbf{u}$ and $\mathbf{f}$ in (32) [7, Eq. 2.9, also last paragraph of Section 2.4] in the same way as (17)

$$\begin{bmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_n \end{bmatrix} \mathbf{M} \begin{bmatrix} \mathbf{P}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_n \end{bmatrix} \begin{bmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_n \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{c} \end{bmatrix} = \begin{bmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_n \end{bmatrix} \begin{bmatrix} \mathbf{f} \\ \mathbf{q} \end{bmatrix},$$

to arrive at

$$\begin{bmatrix} \mathbf{P} \sum_{k=0}^{n} \mathcal{M}[a^{(k)}]\, \mathbf{J}^{n-k}\, \mathbf{P}^T & \mathbf{P} \sum_{k=0}^{n-1} \mathcal{M}[a^{(k)}]\, \mathbf{R}_{n-k} \\ \sum_{k=0}^{n-1} b^{(k,\mathbf{p})}\, \mathbf{t}_{\mathbf{p}}^{T}\, \mathbf{J}^{n-k}\, \mathbf{P}^T & \sum_{k=0}^{n-1} b^{(k,\mathbf{p})}\, \mathbf{t}_{\mathbf{p}}^{T}\, \mathbf{R}_{n-k} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{u}} \\ \mathbf{c} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{f}} \\ \mathbf{q} \end{bmatrix}. \tag{33}$$

We represent the two rows of the matrix in (33) separately for the sake of brevity,

$$\mathbf{L} = \begin{bmatrix} \mathbf{P} \sum_{k=0}^{n} \mathcal{M}[a^{(k)}]\, \mathbf{J}^{n-k}\, \mathbf{P}^T & \mathbf{P} \sum_{k=0}^{n-1} \mathcal{M}[a^{(k)}]\, \mathbf{R}_{n-k} \end{bmatrix}, \tag{34}$$

$$\mathbf{B} = \begin{bmatrix} \sum_{k=0}^{n-1} b^{(k,\mathbf{p})}\, \mathbf{t}_{\mathbf{p}}^{T}\, \mathbf{J}^{n-k}\, \mathbf{P}^T & \sum_{k=0}^{n-1} b^{(k,\mathbf{p})}\, \mathbf{t}_{\mathbf{p}}^{T}\, \mathbf{R}_{n-k} \end{bmatrix}, \tag{35}$$

where $\mathbf{L}$ is the discrete approximation for the differential equation (24a) and $\mathbf{B}$ is the discrete approximation for the boundary conditions (24b). The final discrete expression to solve (24) is given by,

$$\begin{bmatrix} \mathbf{L} \\ \mathbf{B} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{u}} \\ \mathbf{c} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{f}} \\ \mathbf{q} \end{bmatrix}. \tag{36}$$

III. THE SPECTRAL INTEGRATION SUITE

The spectral integration suite is a set of routines to compute the finite-dimensional approximation in (36) to linear operators and block-matrix operators. An example of this is presented in Code 1 where we solve (1a) with boundary conditions in (1b) with a forcing and compare our solution with results from Chebfun [3].

In summary, we use the following routines (and these are sufficient for most problems to compute eigenvalues of or solve for inputs to linear differential equations or block-matrix operators):

Note: Our implementation needs that N (where $N+1$ is the number of basis functions) is an odd number.

- sety(N): Sets points in physical space, $y_i = \cos(\pi\,(i+0.5)/(N+1))$ over $N+1$ points. These points are such that when we take a discrete cosine transform, we have an array that represents spectral coefficients of a Chebyshev basis [8, Eq. 12.4.16-17].
- Discretize(n,N,L): Produces $\mathbf{L}$ in (36) by taking inputs as the highest differential order of the variable, n, N, and the linear operator L (linear operators are specified using cells in Matlab, e.g., the operator $a\,\mathrm{D}^2 + b\,\mathrm{D} + c$ is represented by a 3×1 cell with values $L\{1\} = a$, $L\{2\} = b$, and $L\{3\} = c$).
- BcMat(n,N,eval,L): Generates a matrix of boundary evaluations ($\mathbf{B}$ in (36)) given the highest order of the linear differential equation, n, N, the evaluation point, eval, and the linear operator to be applied at that point (Dirichlet, Neumann or mixed).

- phys2cheb.m: Takes points in physical space (we refer to this as phys-space in this text and our codes) and converts them to an array of spectral coefficients in the basis of Chebyshev polynomials of the first kind (we refer to this as cheb-space in this text and our codes) using [8, Eq. 12.4.16-17].

- cheb2phys.m: Takes an array of spectral coefficients in the basis of Chebyshev polynomials of the first kind, and coverts them to points in phys-space using [8, Eq. 12.4.16-17].

- Matgen(n,N): The solution we solve for, $\hat{\mathbf{u}}$ in (36), is a vector of spectral coefficients of the highest derivatives in the differential equation (25). However, in most cases we need spectral coefficients of the lowest derivative, i.e., the variable itself ((26) with $i = 0$). We use matrices generated from this routine to get to lower derivatives. Suppose we generate [J,K,E] = Matgen(n,N), then J contains matrices corresponding to (6), K corresponding to (27), and the matrices in E are the ones needed to go to lower derivatives (E uses both J and K). For instance, to integrate n times, we multiply the solution $[\hat{\mathbf{u}}^T \;\; \mathbf{c}^T]^T$ from (36) with E{n+1}; see Code 1.

In addition to these primary functions, we provide the following auxiliary functions that are useful in certain applications:

- ChebMat2CellMat Takes a matrix of size $m\,N \times n$, and returns a cell of arrays of size $m \times n$, each element in the cell is a vector representing a function in y.

- AdjointFormal Takes a linear operator or a block matrix operator and returns the formal adjoint.

- keepConverged Takes in eigenvalues, eigenvectors, and N, and returns those eigenvalues and vectors that have converged to machine precision.

- MultOps Gives the composition of two linear (block) matrix operators of compatible dimensions.

- integ Integrates a function in phys-space.

- ChebEval Evaluates a function in cheb-space at points in the domain.

Code 1: Problems with the reaction-diffusion equation, solving for a forcing to the system and eigenvalues and frequency responses of the system.

```
%% Using discretization matrices

% Solving the reaction diffusion equation with a forcing

clear;
clc;
close all;

% STEP 1:
N = 63;
[J,K,D] = Matgen(2,N);
y = sety(N);

% Set eps in reaction-diffusion equation:
eps = 1;
eps2 = eps*eps;

% STEP 2:
% For differential equation:
Delta = D{1} - eps2*D{3};

% For Dy operator is given by 1.0 Dy + 0.0
Dy = cell(2,1); Dy{1} = 1.0; Dy{2} = 0;

bcs = [BcMat(2,N,1,Dy);...
       BcMat(2,N,-1,Dy)];

% Finite-dimensional approximation:
A = [Delta;...
       bcs];
% d, forcing in physical space
d = 1 + y + y.^2;
```

```
% Convert to Chebyshev space:
d = phys2cheb(d);

% STEP 4:
% Solve, pad two zeros as values are boundaries are zero:
D2solutionAndIntConst = A\[d;0;0];
solution = D{3}*D2solutionAndIntConst;

% Convert solution to physical space:
solution = cheb2phys(solution);
plot(y,solution);

%%
% Check with Chebfun:

Aop = chebop([-1 1]);
Aop.op = @(y,u) diff(u,2) - eps2*u;
Aop.lbc = @(u) diff(u);
Aop.rbc = @(u) diff(u);
yc = chebfun('y');
solution = Aop\(1 + yc + yc^2);
hold on;
plot(solution,'--r');
hold off;
%% Using discretizing functions

% STEP 2
Delta = cell(3,1);
Delta{1} = 1.0; Delta{2} = 0; Delta{3} = -eps2;

% Discretize takes arguments: differential order, N, and the operator
Delta = Discretize(2,N,Delta);

% STEP 3
bcOps = {Dy;...
       Dy};
bcEvals =[1;...
         -1];
bcs = BcMat(2,N,bcEvals,bcOps);

% Finite-dimensional approximation:
A = [Delta;...
       bcs];
% forc in physical space
forc = 1 + y + y.^2;

% Convert to Chebyshev space:
forc = phys2cheb(forc);

% STEP 4:
% Solve, pad two zeros as values are boundaries are zero:
D2solutionAndIntConst = A\[forc;0;0];
solution = D{3}*D2solutionAndIntConst;

% Convert solution to physical space:
solution = cheb2phys(solution);
plot(y,solution);

%%
% Check with Chebfun:

Aop = chebop([-1 1]);
Aop.op = @(y,u) diff(u,2) - eps2*u;
Aop.lbc = @(u) diff(u);
Aop.rbc = @(u) diff(u);
yc = chebfun('y');
solution = Aop\(1 + yc + yc^2);
hold on;
plot(solution,'--r');
xlabel('$y$');
ylabel('$\phi (y)$');
hh = legend('Chebfun','SISMatlab')
print('-painters','-dpng','../pics/rnd1');
print('-painters','-dsvg','../pics/rnd1');
```

```
hold off;

%% Solve for Eigen-system

bcN = null(bcs);

I  = cell(1,1); I{1} =1; I = Discretize(2,N,I);
Delta = Delta*bcN;
I = I*bcN;
% V parametrizes null-space
[V,lam] = eig(Delta,I);

% Find the eigenvectors:
V = bcN*V;

% Get to lowest derivatives:
V = D{3}*V;

% Convert to a cell of functions:
V = ChebMat2CellMat(V,N);

% Keep only eigenvalues that converge to machine precision:
[V,lam] = keepConverged(V,lam,N);
lam = diag(lam);

% Got to physical space:
V = cheb2phys(V);

% Sort them:
[lam, ind] = sort(lam,'descend');
V = V(:,ind);

plot(y,V{1,2}/val_rbc(V{1,2}));

%%
Aop = chebop([-1 1]);
Aop.op = @(y,u) diff(u,2) - eps2*u;
Aop.lbc = @(u) diff(u);
Aop.rbc = @(u) diff(u);
y = chebfun('y');
[V,lam] = eigs(Aop);
hold on;
plot(V.blocks{5},'--r');
xlabel('$y$');
ylabel('$\phi (y)$');
hh = legend('Chebfun','SISMatlab','location','Northwest')
print('-painters','-dpng','../pics/rnd2');
print('-painters','-dsvg','../pics/rnd2');
hold off;


%% Solve for Frequency responses
omega = 0;

I = cell(1,1); I{1} = 1;
Z = cell(1,1); Z{1} = 0;


% The operator -1*Dyy + 0*Dy + (1i*omega + eps2)
A = cell(3,1); A{1} = -1; A{2} = 0, A{3} = 1i*omega + eps2;
% Derive its adjoint:
Aad = AdjointFormal(A);
F = {Z,I;...
    I,Z};
E = {A,Z;...
    Z,Aad};
E = Discretize([2 2],N,E);
F = Discretize([2 2],N,F);

BcOps = {Dy,Z;...
       Dy,Z;...
       Z,Dy;...
       Z,Dy};
BcEvals = [1 0
         -1  0
         0  1
         0 -1];
n = [2 2]; % vector that specifies that \phi and \psi have an order 2.
bcs = BcMat(n,N,BcEvals,BcOps);
%%
%
% $$e^{\pi i} + 1 = 0$$
%

y = sety(N);
bcsN = null(bcs);
E = E*bcsN;
F = F*bcsN;
[V,lam] = eig(F,E);
V = bcsN*V;

% Bring to lower derivatives
V =  [D{3}, zeros(size(D{3}));...
    zeros(size(D{3})), D{3}] * V;
% Convert to matrices of vectors;
V = ChebMat2CellMat(V,N);
% Sort and plot
lam = diag(lam);
[lam,indices] = sort(lam,'descend','ComparisonMethod','real');
V = V(:,indices);
V = cheb2phys(V);
plot(y,V{1,2}/val_rbc(V{1,2}));
ylabel('$\phi (y)$');
xlabel('$y$');
print('-painters','-dpng','../pics/rnd3')
print('-painters','-dsvg','../pics/rnd3')
```

III.1. Spatially varying coefficients

Spatially varying coefficients are accounted using an expression for product of Chebyshev polynomials detailed in (29) (see [7, Section 2.2]). Spatially varying coefficients are accounted using MultMat.m, which generates a matrix for the expression in (29). For example, let us consider the solution of the following system:

$$\mathrm{D}u(y) + \frac{1}{y^2+1}u(y) \;=\; 0, \tag{37a}$$

$$u(-1) = 1. \tag{37b}$$

Anaytical solution is given by [7, Eq. 2.12]:

$$u(y) = \exp\left(-\tan^{-1}(y) \,-\, \tan^{-1}(1)\right). \tag{37c}$$

Code 2 solves for $u(y)$ in (37a) and (37b) using spectral integration.

Code 2: Solving (37a) with the boundary condition (37b)

```
N = 63;
k = 1;
k2 = k*k;
y = sety(N);
[~,~,D] = Matgen(1,N); % 1st order linear differential equation

% Using discretization matrices:
mat = D{1} + MultMat(1./(y.^2 + 1))*D{2};

% Or use the Descretize function:
% L = cell(2,1); L{1} = 1; L{2} = 1./(y.^2 + 1);
% mat = Discretize(1,N,L);


% Operator 1 for Bc:
```

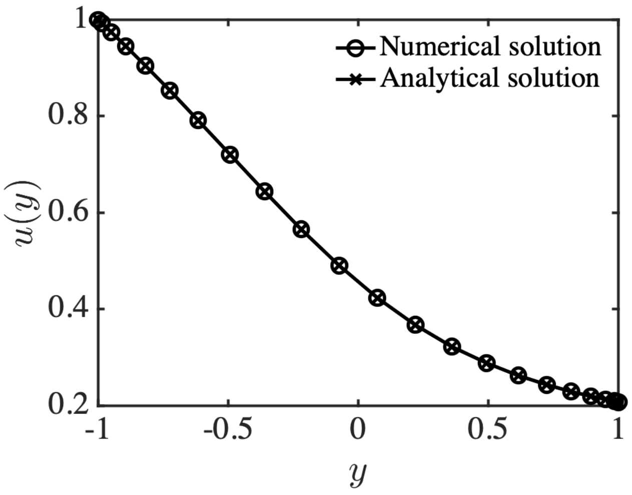


Figure 1: Comparing analytical and numerical solutions to (37) from Code 2.

```
I = cell(1,1); I{1} = 1.0;
bcs = BcMat(1,N,-1,I);

% Finite-dimensional approximation:
mat2 = [mat;...
       bcs];

% Make forcing:
forc = 0*y;

% Convert to Chebyshev space:
forc = phys2cheb(forc);

% Solve, pad 1, i.e., value at boundary:
solution = mat2\[forc;1];

% Note that the above solution is for the highest derivative, need
% to get to lower derivatives:
solution = D{2}*solution;

% Convert solution to physical space:
solution = cheb2phys(solution);

% Analytical solution:
anal_sol = exp(-(atan(y) + atan(1.0)));

% Plot:
plot(y(1:3:end),solution(1:3:end),'-ok',y(1:3:end),anal_sol(1:3:end),'-xk');
ylabel('$u(y)$','FontSize',28);
xlabel('$y$','FontSize',28);
hh = legend('Numerical solution', 'Analytical solution');

print('-painters','-dpng','../pics/Code2');
```

III.2. The linearized Navier-Stokes equations in the descriptor form

We consider the eigenvalues of the linearized Navier-Stokes equations (see § IID and § IIIC.1 in the accompanying paper). This system is given by,

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{\mathcal{E}} \partial_t \begin{bmatrix} u \\ v \\ w \\ p \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{\Delta}{Re} - i\,k_x\,U & -U' & 0 & -i\,,k_x \\ 0 & \frac{\Delta}{Re} - i\,k_x\,U & 0 & -\partial_y \\ 0 & 0 & \frac{\Delta}{Re} - i\,k_x\,U & -i\,k_z \\ i\,k_x & \partial_y & i\,k_z & 0 \end{bmatrix}}_{\mathcal{F}} \begin{bmatrix} u \\ v \\ w \\ p \end{bmatrix}, \tag{38a}$$

$$+ \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}}_{\mathcal{B}} \begin{bmatrix} d_x \\ d_y \\ d_z \end{bmatrix}, \tag{38b}$$

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\mathcal{C}} \begin{bmatrix} u \\ v \\ w \\ p \end{bmatrix}, \tag{38c}$$

with boundary conditions $u(\pm 1) = v(\pm 1) = w(\pm 1) = v'(\pm 1) = 0$ (the boundary condition $v'(\pm 1) = 0$ is derived in § IIIC.1 in the accompanying paper). Note that pressure is expressed with a highest derivative of 2 as discussed in § IIIC.1 in the accompanying paper. The boundary conditions are expressed as

$$\begin{bmatrix} \mathcal{Q}(1,I) & \mathcal{Q}(1,Z) & \mathcal{Q}(1,Z) & \mathcal{Q}(1,Z) \\ \mathcal{Q}(1,Z) & \mathcal{Q}(1,I) & \mathcal{Q}(1,Z) & \mathcal{Q}(1,Z) \\ \mathcal{Q}(1,Z) & \mathcal{Q}(1,Z) & \mathcal{Q}(1,I) & \mathcal{Q}(1,Z) \\ \mathcal{Q}(1,Z) & \mathcal{Q}(1,\mathrm{D}) & \mathcal{Q}(1,Z) & \mathcal{Q}(1,Z) \\ \mathcal{Q}(-1,I) & \mathcal{Q}(-1,Z) & \mathcal{Q}(-1,Z) & \mathcal{Q}(-1,Z) \\ \mathcal{Q}(-1,Z) & \mathcal{Q}(-1,I) & \mathcal{Q}(-1,Z) & \mathcal{Q}(-1,Z) \\ \mathcal{Q}(-1,Z) & \mathcal{Q}(-1,Z) & \mathcal{Q}(-1,I) & \mathcal{Q}(-1,Z) \\ \mathcal{Q}(-1,Z) & \mathcal{Q}(-1,\mathrm{D}) & \mathcal{Q}(-1,Z) & \mathcal{Q}(-1,Z) \end{bmatrix} \begin{bmatrix} u(y) \\ v(y) \\ w(y) \\ p(y) \end{bmatrix} = 0, \tag{38d}$$

where $\mathcal{Q}(a, L)$ is an evaluation operator that evaluates the action of the linear operator L on a variable at a point $y = a$. The generalized eigenvalues of the operators $(\mathcal{F}, \mathcal{E})$ yield eigenvalues of (38). As discussed in § IIIA of the accompanying paper, the norm of the resolvent operator, $\mathcal{A}(\omega)^{-1} = (i\,\omega\mathcal{E} - \mathcal{F})^{-1}$ can be computed using a feedback interconnected system,

$$\begin{bmatrix} 0 & \mathcal{B}\mathcal{B}^{\dagger} \\ \mathcal{C}^{\dagger}\mathcal{C} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\phi} \\ \boldsymbol{\psi} \end{bmatrix} = \gamma \begin{bmatrix} \mathcal{A} & 0 \\ 0 & \mathcal{A}^{\dagger} \end{bmatrix} \begin{bmatrix} \boldsymbol{\phi} \\ \boldsymbol{\psi} \end{bmatrix}, \tag{39}$$

where $\gamma = \pm\sigma$ are the singular values and the maximum singular value is the resolvent norm of (38), $\boldsymbol{\phi} = [\,u\;\; v\;\; w\;\; p\,]^T$ in (38), and $\boldsymbol{\psi}$ are the adjoint variables corresponding to $\boldsymbol{\phi}$.

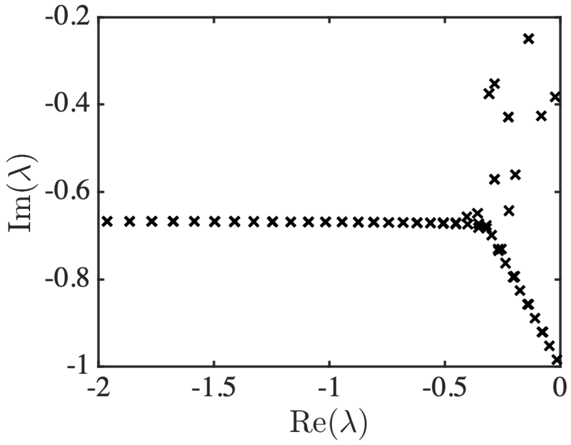
Finally, the maximum singular value over all $\omega \in \mathbb{R}$ is computed using the fast algorithm by Bruinsma and Steinbuch [2]. The Matlab code for these different problems concerning the linearized Navier-Stokes equations is given in Code 3.

Figure 2 shows plots generated from Code 3. Figure 2a shows the eigenvalues for the case of plane Poiseuille flow with $Re = 2000$, $k_x = k_z = 1$ (also see Figure 10 in the accompanying paper). Figure 2b shows the largest (σ_0) and the second largest (σ_1) singular values calculated from (39). The singular values computed in Figure 2b are in agreement with [9, Figure 4.10].

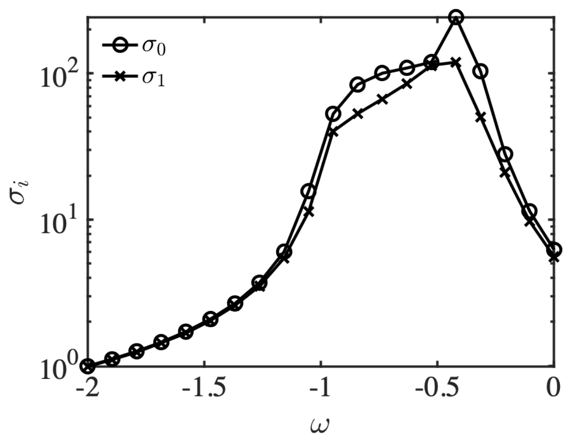
Code 3: Solving for the eigenvalues, resolvent norms, and the $\mathcal{H}_\infty$-norm of (38)

```
clear;
clc;
N = 91;
```

```
[J,K,D] = Matgen(2,N);
y = sety(N);

% Set parameters:
Re = 2000;
kx = 1;
kz = 1;
U = 1-y.^2; % Poiseuille flow.
Uy = -2*y;

% Make operators: identity and zero:
I = cell(1,1); I{1} = 1; Z = cell(1,1); Z{1} = 0;
Dy = cell(2,1); Dy{1} = 1; Dy{2} = 0;

% Make the diagonal of F:
k2 = kx*kx + kz *kz;
F11 = cell(3,1);
F11{1} = 1/Re;
F11{2} = 0;
F11{3} = (-k2/Re) - 1i*kx*U;

% Make other operators for F12, F14 etc.
F12 = cell(1,1) ; F12{1} = -Uy;
F14 = cell(1,1); F14{1} = -1i*kx;
F24 = cell(2,1); F24{1} = -1; F24{2} = 0;
F34 = cell(1,1); F34{1} = -1i*kz;
F41 = cell(1,1); F41{1} = 1i*kx;
F42 = cell(2,1); F42{1} = 1; F42{2} = 0;
F43 = cell(1,1); F43{1} = 1i*kz;

% Make the operator:

F = {F11, F12, Z, F14;...
    Z, F11, Z, F24;...
    Z, Z, F11, F34;...
    F41, F42, F43, Z};

% Make E:
E = {I, Z, Z, Z;...
   Z, I, Z, Z;...
   Z, Z, I, Z;...
   Z, Z, Z, Z};

% Make boundary conditions matrix:
bcOps = {I,Z,Z,Z;...
  Z,I,Z,Z;...
  Z,Z,I,Z;...
  Z,Dy,Z,Z;
  I,Z,Z,Z;...
  Z,I,Z,Z;...
  Z,Z,I,Z;...
  Z,Dy,Z,Z};
bcEvals = [-ones(4,4);ones(4,4)]; % as first 4 rows for bc at y = -1, next four rows for y = 1.

% Differential orders of u, v, w, and p:
n = [2,2,2,2];

% Make boundary conditions matrix
M = BcMat(n,N,bcEvals,bcOps);

% Find null space of M:
kerM = null(M);

% Find eigenvalues
Ff = Discretize(n,N,F)*kerM;
Ef = Discretize(n,N,E)*kerM;
[~,evals] = eig(Ff,Ef);

plot(real(evals),imag(evals),'xk');
ylabel('$\mathrm{Im}(\lambda)$');
xlabel('$\mathrm{Re}(\lambda)$');
ax = gca;
ax.XLim = [-2,0];
ax.YLim = [-1,-0.2];
ax.YTick = [-1 -0.8 -0.6 -0.4 -0.2];
ax.XTick = [-2 -1.5 -1 -0.5 0];
print('-painters','-dpng','../pics/LNSEigs');


% Specify the frequency:
omegas = linspace(-2,0,20);

% Store first two singular values:
eval1 = zeros(20,1);
eval2 = zeros(20,1);
for i = 1:length(omegas)
omega = omegas(i);
% Make operator A:
A = 1i*omega*Discretize(n,N,E) - Discretize(n,N,F);
% Make operator A-adjoint:
Ead = AdjointFormal(E);
Fad = AdjointFormal(F);
Aad = -1i*omega*Discretize(n,N,Ead) - Discretize(n,N,Fad);

% BBadjoint and CadjointC is the same, given by
BBad = {I,Z,Z,Z;...
     Z,I,Z,Z;...
     Z,Z,I,Z;...
     Z,Z,Z,Z};
CadC = BBad;

BBad = Discretize(n,N,BBad);
CadC = Discretize(n,N,CadC);
% Forms feedback boundary conditions:
Mf = [M, zeros(size(M));...
zeros(size(M)), M];

% Find null space:
kerMf = null(Mf);

%Form feedback operator and find eigenvalues:
Zs = zeros(size(A));
lams = eigs([Zs,BBad; CadC,Zs]*kerMf, [A,Zs; Zs, Aad]*kerMf, 2, 'LR');
eval1(i) = lams(1);
eval2(i) = lams(2);
disp(i);
end

semilogy(omegas, real(eval1),'-ok',omegas, real(eval2),'-xk');
ylabel('$\sigma_i$');
xlabel('$\omega$');
hh = legend('$\sigma_0$','$\sigma_1$','location','northwest');
hh.Interpreter = 'latex';
ax = gca;
print('-painters','-dpng','../pics/SchmidSol');

%% Directly compute the H inf norm at Re = 2000:

kz = 1;
kx = 1;
F11 = cell(3,1);
F11{1} = 1/Re;
F11{2} = 0;
F11{3} = (-k2/Re) - 1i*kx*U;

% Make other operators for F12, F14 etc.
F12 = cell(1,1) ; F12{1} = -Uy;
F14 = cell(1,1); F14{1} = -1i*kx;
F24 = cell(2,1); F24{1} = -1; F24{2} = 0;
F34 = cell(1,1); F34{1} = -1i*kz;
F41 = cell(1,1); F41{1} = 1i*kx;
F42 = cell(2,1); F42{1} = 1; F42{2} = 0;
F43 = cell(1,1); F43{1} = 1i*kz;

% Make the operator:

F = {F11, F12, Z, F14;...
    Z, F11, Z, F24;...
    Z, Z, F11, F34;...
```



(a)



(b)

Figure 2: Eigenvalues and frequency responses of the linearized Navier equations in plane Poiseuille flow with $Re = 2000$ and $k_x = k_z = 1$, generated from Code 3. The spectral integration method with $N = 91$ basis functions is used. (a) Spectrum resulting from the use of the descriptor formulation, and (b) two largest singular values of the frequency response operator computed using Code 3. A reference for this plot is found in [9, Figure 4.10].

```
    F41, F42, F43, Z};

% Make E:
E = {I, Z, Z, Z;...
    Z, I, Z, Z;...
    Z, Z, I, Z;...
    Z, Z, Z, Z};

% Make B:
B = {I,Z,Z;...
    Z,I,Z;...
    Z,Z,I;...
    Z,Z,Z};

% Make C:
C = {I,Z,Z,Z;...
     Z,I,Z,Z;...
     Z,Z,I,Z};
n = [2,2,2,2];
[omega_opt,Hinf] = HinfNorm(n,N,E,F,B,C,kerMf);
disp(['Omega_opt: ' num2str(omega_opt)]);
disp(['Hinfinity norm: ' num2str(Hinf)]);
```

Appendix A: Recurrence relations

Consider the expression for the highest derivative of a second order differential equation in a Chebyshev basis,

$$\mathrm{D}^2 u(y) \;=\; u_0^{(2)}\,\tfrac{1}{2}T_0(y) \;+\; u_1^{(2)}\,T_1(y) \;+\; u_2^{(2)}\,T_2(y) \;+\; u_3^{(2)}\,T_3(y) \;+\; \cdots\,. \tag{A1}$$

Relation of Chebyshev polynomials with derivatives is given by [5, Equation 3.25]

$$T_0(y) \;=\; T_1'(y), \tag{A2a}$$

$$T_1(y) \;=\; \tfrac{1}{4}\,T_2'(y), \tag{A2b}$$

$$T_n(y) \;=\; \tfrac{1}{2}\left(\frac{T_{n+1}'(y)}{n+1} - \frac{T_{n-1}'(y)}{n-1}\right), \qquad n > 1. \tag{A2c}$$

Substuting (A2) in (A1) and making an indefinite integration on the resultant expression yields,

$$\begin{aligned}\mathrm{D}\,v(y) \;=&\; \frac{u_0^{(2)}}{2}y + \frac{u_1^{(2)}}{2}y^2 + \frac{u_2^{(2)}}{2}\left(\frac{T_3(y)}{3} - \frac{T_1(y)}{1}\right) + \frac{u_3^{(2)}}{2}\left(\frac{T_4(y)}{4} - \frac{T_2(y)}{2}\right)\\ &+ \frac{u_4^{(2)}}{2}\left(\frac{T_5(y)}{5} - \frac{T_3(y)}{3}\right) + \cdots + c_0,\end{aligned} \tag{A3}$$

where c_0 is the effective integration constant. As $y^2 = (T_0(y) + T_2(y))/2$, (A3) takes the form:

$$\begin{aligned}\mathrm{D}\,v(y) \;=&\; \frac{u_0^{(2)}}{2}y + \frac{u_1^{(2)}}{2}\left(\frac{T_0(y) + T_1(y)}{2}\right) + \frac{u_2^{(2)}}{2}\left(\frac{T_3(y)}{3} - \frac{T_1(y)}{1}\right)\\ &+ \frac{u_3^{(2)}}{2}\left(\frac{T_4(y)}{4} - \frac{T_2(y)}{2}\right) + \frac{u_4^{(2)}}{2}\left(\frac{T_5(y)}{5} - \frac{T_3(y)}{3}\right) + \cdots + c_0,\end{aligned} \tag{A4}$$

$$= T_1(y)\underbrace{\left(\frac{u_0^{(2)}}{2} - \frac{u_2^{(2)}}{2}\right)}_{u_1^{(1)}} + T_2(y)\underbrace{\left(\frac{u_1^{(2)}}{4} - \frac{u_3^{(2)}}{4}\right)}_{u_2^{(1)}} + T_3(y)\underbrace{\left(\frac{u_2^{(2)}}{6} - \frac{u_2^{(2)}}{6}\right)}_{u_3^{(1)}} + \cdots + \underbrace{\frac{u_1^{(2)}}{4}}_{u_0^{(1)}/2} + c_0. \tag{A5}$$

Hence we have from (A5),

$$\mathrm{D}\,v(y) \;=\; u_0^{(1)}\,\tfrac{1}{2}T_0(y) \;+\; u_1^{(1)}\,T_1(y) \;+\; u_2^{(1)}\,T_2(y) \;+\; u_3^{(1)}\,T_3(y) \;+\; \cdots \;+\; c_0, \tag{A6}$$

where $u_0^{(1)} = u_1^{(2)}/2$ and the remaining coefficients for $u_i^{(1)}$ from the recursive relation in (5).

[1] D. Bourne. Hydrodynamic stability, the Chebyshev tau method and spurious eigenvalues. *Cont. Mech. Therm.*, 15:571–579, 2003.

[2] N. A. Bruinsma and M. Steinbuch. A fast algorithm to compute the $\mathcal{H}_\infty$-norm of a transfer function matrix. *Syst. Control Lett.*, 14:287–293, 1990.

[3] T. A. Driscoll, N. Hale, and L. N. Trefethen. *Chebfun guide.* Pafnuty Publications, 2014.

[4] K. Du. On well-conditioned spectral collocation and spectral methods by the integral reformulation. *SIAM J. Sci. Comp.*, 38:A3247–A3263, 2016.

[5] A. Gil, J. Segura, and N. M. Temme. Chebyshev expansions. In *Numerical methods for special functions*, chapter 3, pages 51–86. Society for Industrial and Applied Mathematics, 2007.

[6] L. Greengard. Spectral integration and two-point boundary value problems. *SIAM J. Numer. Anal.*, 28:1071–1080, 1991.

[7] S. Olver and A. Townsend. A fast and well-conditioned spectral method. *SIAM Rev.*, 55:462–489, 2013.

[8] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes 3rd Edition: The art of scientific computing.* Cambridge University Press, 2007.

[9] P. J. Schmid and D. S. Henningson. *Stability and transition in shear flows.* Springer Science and Business Media, 2012.