Spectral methods for BVPs

(steady-state equations)

A. Donev, Spring 2021

For periodic domains, Fourier based (pseudo)spectral methods work great for smooth problems with or without time dependence. However, for bounded domains, Fourier series do <u>not</u> have rapidly (exponentially) converging coefficients even for analytic functions, so we must switch instead to orthogonal polynomials as basis functions. I will focus on Chebyshev polynomials here.

Much of what we will say here applies to all series methods, meaning that we represent the solution not by a finite collection of values (FD) or averages (FV) on a grid but rather as a finite collection of coefficients in some basis (typically for L_2):

$$u(\vec{x}) = \sum_{k=1}^{N} c_k \, \varphi_k(\vec{x})$$

unknowns

For spectral methods, $\varphi_k(x)$ are orthogonal polynomials. For Finite Element Methods (FEM) they are localized polynomials.

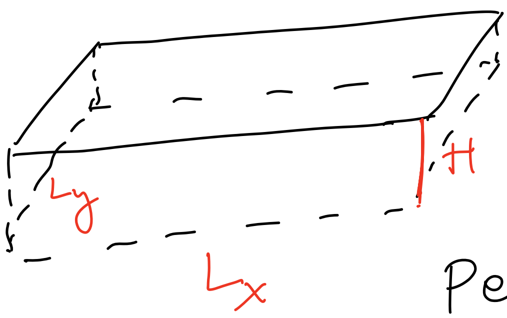
For spectral series methods, the grid is secondary, and we work with functions and/or coefficients and not with grid values per se. Two key questions are:

1. How to impose the PDE in the domain
2. How to impose the BCs

Instead of trying to give some general overview of theory (which may not even exist), I will instead show several different methods to solve one problem my research group has studied recently

(Much of this comes from PhD student Ondrej Maxian)

We will consider
3D Poisson equation in slit domain



Domain is $L_x \times L_y \times H$ box

Periodic in x & y

Electric potential satisfies:

$$\nabla^2 \varphi(x,y,z) = -f(x,y,z)$$

$\uparrow$ charge density

(and we assume f is smooth)

with some BCs in the z direction (aperiodic), such as

$$\begin{cases} \dfrac{\partial \varphi}{\partial z}(z=0) = \sigma_b(x,y) \\ \dfrac{\partial \varphi}{\partial z}(z=H) = -\sigma_t(x,y) \end{cases}$$

Step 1: Use Fourier series in x & y, i.e., take Fourier transform of PDE:

$$\vec{k}_{\parallel} = (k_x, k_y) = \left(\frac{2\pi}{L_x} n, \frac{2\pi}{L_y} m\right)$$

$n, m \in \mathbb{Z}$

$$\varphi(x,y,z) = \sum_{n,m} e^{i(xk_x + yk_y)} \hat{\varphi}_{n,m}(z)$$

we will truncate to $N_x \cdot N_y$ terms

$$\nabla^2 \to -k_{\parallel}^2 + \partial_z^2 \Rightarrow$$

$$\partial_z^2 \hat{\varphi}_{n,m}(z) - k_{\parallel}^2 \hat{\varphi}_{n,m}(z) = -\hat{f}_{n,m}(z)$$

Fourier transform of rhs

+ some BCs at $z=0$ & H

For every parallel wavenumber $k_{\parallel}$ (i.e., every n, m) there is a boundary value problem in one dimension! Once we solve this BVP we can do an inverse FFT in the x/y directions to get solution on a uniform grid in x/y.

So here let's focus on BVP:

$$(*) \quad \partial_z^2 u(z) - k^2 u(z) = -f(z)$$

$z \in [-1, 1]$

$$\text{BCs} \begin{cases} (\partial_z u + k u)_{z=1} = \alpha \\ (\partial_z u - k u)_{z=-1} = \beta \end{cases}$$

chosen for illustration

This is a linear BVP and also constant-coefficient so it is quite "easy", but it is a good example to illustrate some ideas, most of which can be generalized to other more complicated problems.

First method: Integral reformulation

This method is due to Leslie Greengard & is specialized but also the most efficient/robust spectral method to solve (*) for smooth $f(z)$ that I know of. It won't work for large k, due to boundary layers (switch to adaptive method if needed)

Let's use Chebyshev polynomials and keep only N terms:

$$\begin{cases} u(z) = \sum_{n=0}^{N-1} \hat{u}_n T_n(z) \\ f(z) = \sum_{n=0}^{N-1} \hat{f}_n T_n(z) \end{cases}$$

Let's review briefly some things about

Chebyshev polynomials

Chebyshev polynomials of the first kind:

$$T_n(\cos\theta) = \cos(n\theta)$$

$$T_n(z) = \cos(n \arccos z)$$

orthogonal w.r.t. inner product

$$\langle f, g \rangle = \int_{-1}^{1} f(z)\, g(z) \frac{dz}{\sqrt{1-z^2}}$$

since they solve the BVP:

$$(1-z^2)\, u'' - z u' + n^2 u = 0$$

Normalized so their max/min value is ± 1 on $[-1, 1]$

$$\begin{cases} T_0 = 1 \\ T_1 = z \\ T_{n+1} = 2z T_n - T_{n-1} \end{cases}$$

The roots of T_n give a non-uniform Chebyshev grid.

One can use the (i)FFT to go (stably!) from function values on

the Chebyshev grid to the Chebyshev coefficients of the unique polynomial interpolant (Chebyshev interpolant)
[see FFT notes on webpage]

This is one of the reasons I prefer Chebyshev over Legendre, but one can use Legendre also if desired.

Lots of tools implemented in the chebfun matlab library (Nick Trefethen's group)

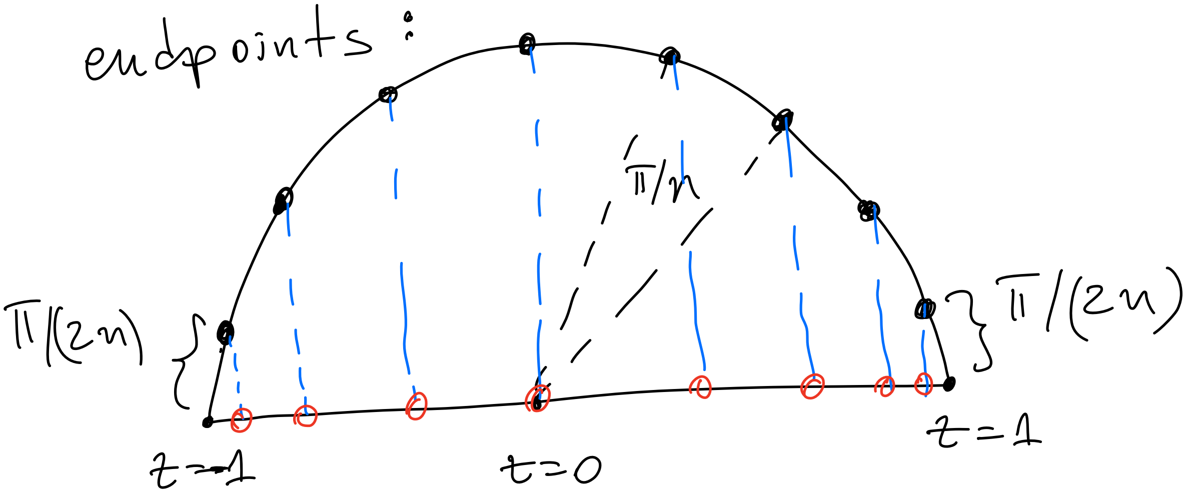
[For smooth functions the Chebyshev coefficients decay exponentially fast]

Chebyshev nodes / grid 1^{st} kind:

$$t_i = \cos\left(\frac{(2i+1)}{2n}\pi\right)$$

$i = 0, \ldots, n-1$

Does NOT include the endpoints:



Once we have the Chebyshev interpolant (interpolating polynomial), we can approximate integrals / derivatives by integrating / differentiating the polynomial.

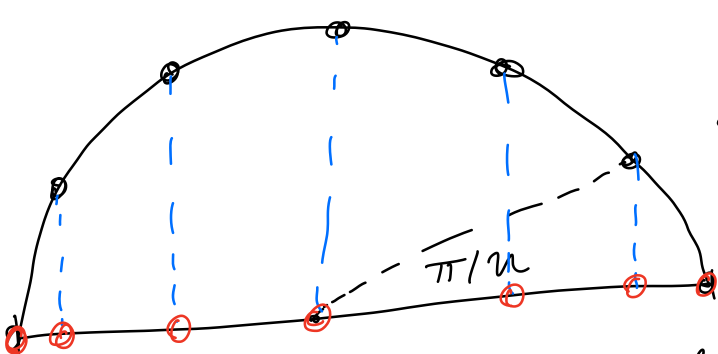
Called Chebyshev differentiation & Clenshaw-Curtis quadrature.

The extrema of T_n form the Cheb. grid of 2^{nd} kind:

$$z_i = \cos\left(\frac{i\pi}{n}\right)$$

$i = 0, \ldots, n$

Which does include endpoints



Also doable by FFT

This is the set of nodes most commonly called Chebyshev nodes (see review by Kuan Xu on webpage)

Some properties:

① $$T_n(T_m(z)) = T_{nm}(z)$$

(composition)

② $$2T_m(z)\,T_n(z) = T_{m+n}(z) + T_{|m-n|}(z)$$

Just like with Fourier, nonlinear terms grow the series & aliasing issues arise. In fact Chebyshev is as close to Fourier as one can get on a bounded domain.

But, there are ways in which Chebyshev is less convenient than Fourier.

$$T_n' = n\, U_{n-1} \quad \text{where}$$

$$U_n = \begin{cases} 2 \sum_{\text{odd } j} T_j & \text{if } n \text{ odd} \\ 2 \sum_{\text{even } j} T_j & \text{if } n \text{ even} \end{cases}$$

Cheb. poly. of 2nd kind

Differentiation is no longer a diagonal or even a nice banded matrix in the space of Chebyshev coefficients, so even for constant-coefficient linear BVPs not as simple as Fourier based methods!
Generally need to deal with dense matrices.
(But see paper by Townsend on course webpage)

But !

$$2T_n = \frac{1}{n+1} T'_{n+1} - \frac{1}{n-1} T'_{n-1}$$

$$\Rightarrow \int T_n(z)\,dz = \frac{1}{2}\left(\frac{T_{n+1}(z)}{n+1} - \frac{T_{n-1}(z)}{n-1}\right)$$

so Chebyshev integration is a banded matrix (tridiagonal)

This suggests more efficient to work with integral formulation.

Introduce $u''(x) = \sigma(x)$ as the variable $\Rightarrow$

$$u(x) = \int_{-1}^{x} \int_{-1}^{t} \sigma(s)\,ds\,dt + c_1 x + c_0$$

and plug into more general 2-point BVP:

$$(Lu)(x) = \lambda u''(x) + \mu u'(x) + \nu u(x) = f(x)$$

$$\sigma(x) = \sum_{k=0}^{N-1} \hat{\sigma}_k T_k(x)$$

or $\frac{\hat{\sigma}_0}{2}$ for zeroth mode

$$f(x) = \sum_{k=0}^{N-1} \hat{f}_k T_k(x)$$

Integral equation

$$\left(\lambda \underline{I} + \mu \overset{\leftrightarrow}{\underset{\sim}{I}}_N + \nu \overset{\leftrightarrow}{\underset{\sim}{I}}{}^2_N\right) \vec{\hat{\sigma}} = \vec{\hat{f}}$$

Integration matrix

Double integration matrix

$\boxed{\text{truncates } N+1 \text{ term}}$

Banded matrices

bandwidth = 3 + bandwidth = 5

If $\lambda \neq 0$ this is a well-conditioned matrix unlike using Chebyshev differentiation matrices!

The two unknown constants c_0 & c_1 (integration constants) need to be determined from BCs.

For example,

$$u'(1) + k\,u(1) = \alpha \Rightarrow$$

$$\underbrace{\underline{I}_N(1)}_{\text{real-space integration (dense row)}} \cdot \hat{\sigma} + c_1 +$$

$$k\left(\widetilde{\underline{I}_N^2}(1) \cdot \hat{\sigma} + c_1 + c_0\right) = \alpha$$

This approach leads to a pentadiagonal well-conditioned linear system & lets us impose the BCs simply. We imposed the PDE in coefficient / Chebyshev space not in real space, but BCs were imposed in real space.

Second method: Block spectral approach

(see review article by Aurentz & Trefethen on webpage)

A more standard approach is to stick to the differential formulation and to impose the PDE & BCs in real space, i.e, on the Chebyshev grid

This is generally referred to as collocation approach: impose PDE & BCs in the strong sense at collocation points.

Specifically, if we define a differentiation matrix

$$u'(\text{grid}) = D_N \, u(\text{grid})$$

$\uparrow$ dense matrix

$$u(\text{grid}) \longrightarrow \text{polynomial interpolant (Chebyshev grid)} \xrightarrow{\frac{d}{dx}} u'(\text{grid})$$

and $u''(\text{grid}) = D_N^2 \, u(\text{grid})$

then we can convert a BVP into a dense linear system, e.g.

$$u'' - k^2 u = -f$$

$$D_N^2 \vec{u} - k^2 \vec{u} = -\vec{f}$$

If we use (as is most common) a type 2 Chebyshev grid, then the end points are included above. The common approach is to drop the first & last equation, i.e., NOT impose the PDE at the endpoints, and instead at the endpoints impose the BCs, e.g.:

$$u'(1) + k u(1) = \alpha$$

$$D_N(1) \cdot u + k u(1) = \alpha$$

This leads to an invertible (for well-posed BVPs) but badly ill-conditioned & dense linear system for $\vec{u}$. But if we don't need many terms (i.e., if solution is quite smooth), we can just use a direct $O(N^3)$ solver (can be OK if we can re-use factorization of matrix multiple times, e.g., inside a time loop).

More recently an alternative block matrix way of imposing BCs has been used that is the analog of ghost cells for spectral methods

The idea is:

1) Use a type 1 Chebyshev grid as collocation points so the end points are <u>not</u> included

2) To impose BCs, go from type 1 to type 2 grid, and impose BCs there.

Specifically, let there be K BCs, e.g., $K=2$ for 2^{nd} order equations, and let $\tilde{N} = N + K$

Let u be function pointwise values on N-point type 1 Cheb grid, and $\tilde{u}$ be the pointwise values of function at $\tilde{N}$-point type 2 grid:

$$\underset{\substack{\uparrow\\ N \times \tilde{N} \\ \text{matrix}}}{\overset{\text{restriction matrix} \rightarrow}{R}} \; \underset{\substack{\nwarrow \\ \text{type 2}}}{\tilde{u}} = \underset{\substack{\nwarrow \\ \text{type 1}}}{u}$$

$\tilde{u} \longrightarrow$ polynomial of degree $\tilde{N}-1$

$\longrightarrow u \equiv$ values of polynomial at N nodes

[$\xrightarrow[\text{if needed}]{}$ polynomial of degree $N-1$]

$$\underset{\substack{\uparrow \\ [K \times \tilde{N}]}}{B}\,\tilde{u} = b \equiv \text{BCs at endpoints}$$

$$\begin{array}{c} N \\ \\ K \end{array} \begin{bmatrix} R \\ \hline B \end{bmatrix} \tilde{u} = \begin{bmatrix} u \\ b \end{bmatrix}$$

$$\Rightarrow \quad \tilde{u} = \begin{bmatrix} R \\ B \end{bmatrix}^{-1} \left\{ \begin{bmatrix} u \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ b \end{bmatrix} \right\}$$

$$\tilde{u} = \underbrace{E}_{\substack{\text{extension} \\ \text{matrix that} \\ \text{fills in "ghost cells" at} \\ \text{the end points using BCs}}} u + \underbrace{F b}_{\substack{\text{inhomogeneous} \\ \text{BCs}}}$$

Now, if we want to compute

$$\mathcal{L} u = u'' - k^2 u, \text{ i.e.,}$$

to discretize elliptic operator with homogeneous BCs, we compute the derivatives on the $\tilde{N}$ type 2 grid:

$$Lu = \underbrace{\left(R D_{\tilde{N}}^2 E - k^2 I\right)}_{\boxed{\text{discrete operator + homogeneous BCs}} \text{ + inhomogeneous BC terms}} u$$

One could discretize PDE as

#1 $\quad Lu = f$ on N-point type 1 grid

and, if there was time dependence, $u_t = \mathcal{L}u$, obtain "method of lines"

$$\frac{du}{dt} = Lu \quad \text{with BCs}$$

My group has used this version #1.
the original paper instead thinks of $\tilde{u}$ as the solution.
(Driscoll & Hale)

That is, we could think of this as "extending" the grid by K nodes and imposing BCs on the extended grid, or, as only imposing the PDE on an "subsampled" grid of K nodes less and imposing the BCs using the K end nodes:

$$L\tilde{u} = R(D^2_{\tilde{N}} - k^2 I)\tilde{u} = R\tilde{f}$$

$$B\tilde{u} = b$$

version #2
$$\begin{bmatrix} R(D^2_{\tilde{N}} - k^2 I) \\ \hline B \end{bmatrix} \tilde{u} = \begin{bmatrix} R\tilde{f} \\ \hline b \end{bmatrix}$$

Third method: Galerkin approach

A rather different approach to spectral solution of BVPs comes from the FEM world, and we will see it again later. This is based on (unpublished) notes by Ondrej Maxian.

Go back to:

$$\begin{cases} u''(x) - k^2 u(x) = f(x) \\ u'(-1) - k\,u(-1) = 0 \\ u'(1) + k\,u(1) = 0 \end{cases}$$

homogeneous natural BCs

(Distinction between "natural" & "essential" BCs comes from variational formulations in FEM)

Option #1 : Use as basis functions polynomials that satisfy the BCs. Turns out that

$$\varphi_n(x) = T_n(x) - \frac{k+n^2}{4+k+4n+n^2} T_{n+2}(x)$$

satisfies the BCs. This is a reasonable polynomial basis but it is NOT orthogonal.

Option #2 : Use the basis

$$\varphi_n(x) = T_n(x)$$

and impose BCs differently.

With either choice, first we need to impose PDE!

Weakly imposing PDE (Galerkin):

$$-u'' + k^2 u = -f$$

Take dot products of PDE with all the basis functions, then use integration by parts using the BCs (if it works):

$$-\langle u'' + k^2 u, \varphi_n \rangle_{L_2} = -\langle f, \varphi_n \rangle_{L_2}$$

weak form

for all n

Plug in $u(x) = \sum_{k=0}^{N-1} \hat{u}_k \varphi_k(x)$

$$\sum_j \left(-\langle \varphi_j'', \varphi_n \rangle + k^2 \langle \varphi_j, \varphi_n \rangle \right) \hat{u}_j = -\langle f, \varphi_n \rangle := -\hat{f}_n \quad (**)$$

Now note that by integration by parts

$$\langle \varphi_j'', \varphi_n \rangle = -\langle \varphi_j', \varphi_n' \rangle + \left(\varphi_j' \varphi_n \right)_{-1}^{1}$$

If φ's satisfy BCs (option #1) then:

$$\varphi_j'(\pm 1) = \mp k \varphi_j(\pm 1)$$

So

$$\left(\varphi_j' \varphi_n \right)_{-1}^{1} = -k \left(\varphi_j \varphi_n \big|_{1} + \varphi_j \varphi_n \big|_{-1} \right)$$

Putting this together in the Galerkin discretization ($\cancel{*}$ $*$) gives the linear system

$$(A + k^2 B)\hat{u} = -\hat{f}$$

$$A_{nj} = \langle \varphi_n', \varphi_j' \rangle$$
$$+ k\left(\varphi_j(1)\varphi_n(1) + \varphi_j(-1)\varphi_j(1)\right)$$

which is a symmetric positive (semi) definite (SPD) matrix

<u>Why</u>? $\sum_{n,j} \hat{u}_n \langle \varphi_n', \varphi_j' \rangle \hat{u}_j =$

$$= \langle \sum_n \hat{u}_n \varphi_n', \sum_j \hat{u}_j \hat{\varphi}_j' \rangle$$

$$= \langle u', u' \rangle \geq 0$$

and the boundary term is of the rank-1 SPD form:

$$k\left(\vec{\varphi}(-1)\vec{\varphi}(-1)^T + \vec{\varphi}(1)\vec{\varphi}(1)^T\right)$$

with $k > 0$

Similarly

$B_{jn} = \langle \varphi_j, \varphi_n \rangle$ is SPD

so $A + k^2 B$ is SPD if $k \neq 0$

$$\Rightarrow \hat{u} = -\underbrace{(A + k^2 B)}_{\text{dense SPD matrix}}^{-1} \hat{f}$$

The only missing piece is how to estimate $\hat{f}$

$$\hat{f}_n = \langle f, \varphi_n \rangle_{L_2}$$

If we are given f on a Chebyshev grid of N points, then we can approximate f with a polynomial of degree $N-1$, and so $\langle f, \varphi_n \rangle_{L_2}$ is an integral of a product of two polynomials, which we can <u>compute</u> <u>exactly</u> on an upsampled grid of $\tilde{N} \geq 2N-1$ Chebyshev nodes if both polynomials are of degree no bigger than $N-1$.

Denote matrix

$$\tilde{\Phi}_{ij} = \varphi_i(x_j)$$

φ_i: N basis functions

x_j: $\tilde{N}$ nodes

"Vandermonde" matrix

and put Clenshaw-Curtis quadrature weights on the diagonal of matrix W.

Then $B = \Phi^T W \Phi$

and $A = \Phi^T \tilde{D}^T W \tilde{D} \Phi +$ $\leftarrow$ differentiation matrix

$k \left(\vec{\varphi}(1) \vec{\varphi}^T(1) + \vec{\varphi}(-1) \vec{\varphi}^T(-1) \right)$

<u>Note</u>: It is possible to apply these matrices in $O(N \log N)$ time using FFTs and recursions

Things work out similarly if we use option #2 ($\varphi_n \equiv T_n$) but now we need to weakly impose the BCs since φ_n do not satisfy the BCs. Let's illustrate this on our example BVP:

$$\begin{cases} u'(-1) - k u(-1) = \alpha \\ u'(1) + k u(1) = \beta \end{cases}$$

PDE in weak form:

$$-\langle u'', \varphi_n \rangle + k^2 \langle u, \varphi_n \rangle = -\hat{f}_n$$

Use integration by parts and the BCs to get

$$\langle u', \varphi_n' \rangle + k^2 \langle u, \varphi_n \rangle$$

$$+ \left(\alpha + k u(-1) \right) \varphi_n(-1)$$

$$+ \left(-\beta + k u(1) \right) \varphi_n(1) = -\hat{f}$$

inhomogeneous BCs

Now again use $u(x) = \sum \hat{u}_n \varphi_n(x)$ as we did for option #1, and get the linear system

$$\left(A + k^2 B\right)\hat{u} = -\hat{f} + \underbrace{\beta \vec{\varphi}(1) - \alpha \vec{\varphi}(-1)}_{\text{inhomogeneous BCs on r.h.s.}}$$

same matrices as before

$\boxed{\text{SPD operator} \Leftrightarrow \text{SPD matrix}}$

Interestingly, this is the same linear system as for option #1 if the BCs are homogeneous, even though our basis do not satisfy the BCs.

This is because this specific BC is a <u>natural</u> BC and does not need to be enforced explicitly (it follows from the PDE) by extra equations or by the basis functions.