Lesson 13
Spectral methods
• In this lecture we consider the problem of computing solutions to linear ODEs:

\[ a^{(N)}(\theta)u^{(N)} + \cdots + a^{(0)}(\theta)u = f(\theta) \quad \text{on the periodic interval} \]

\[ a^{(N)}(z)u^{(N)} + \cdots + a^{(0)}(z)u = f(z) \quad \text{inside the unit disk} \]

\[ a^{(N)}(x)u^{(N)} + \cdots + a^{(0)}(x)u = f(x) \quad \text{on the unit interval} \]

• We will impose boundary conditions so that the solution exists and is unique

• For example, on the periodic interval we might solve:

\[ u'' + u \cos \theta = 0, \]

\[ u(-\pi) = u(\pi) = 1 \]

• The basic idea: represent the infinite-dimensional differential operator with a finite dimensional matrix
Fourier series spectral methods
• We start with a simple equation:

\[ u' + u = f(\theta), \quad u(-\pi) = u(\pi) \]

• Idea: represent \( u \) and \( f \) by their Fourier series:

\[
\sum_{k=0}^{\infty} \hat{u}_k (ik + 1)e^{ik\theta} = \sum_{k=0}^{\infty} \hat{f}_k e^{ik\theta}
\]
We start with a simple equation:

\[ u' + u = f(\theta), \quad u(-\pi) = u(\pi) \]

Idea: represent \( u \) and \( f \) by their Fourier series:

\[
\sum_{k=0}^{\infty} \hat{u}_k(i k + 1)e^{ik\theta} = \sum_{k=0}^{\infty} \hat{f}_k e^{ik\theta}
\]

The solution is obvious. But let’s anyways represent this in matrix form:

\[
\begin{pmatrix}
\vdots \\
-2i + 1 \\
-1 + 1 \\
1 \\
i + 1 \\
2i + 1 \\
\vdots 
\end{pmatrix}
\begin{pmatrix}
\hat{u}_{-2} \\
\hat{u}_{-1} \\
\hat{u}_0 \\
\hat{u}_1 \\
\hat{u}_2 \\
\vdots 
\end{pmatrix}
= 
\begin{pmatrix}
\vdots \\
\hat{f}_{-2} \\
\hat{f}_{-1} \\
\hat{f}_0 \\
\hat{f}_1 \\
\hat{f}_2 \\
\vdots 
\end{pmatrix}
• Numerically, we truncate the operator and use the DFT to compute the coefficients:

\[
\begin{pmatrix}
(\alpha i + 1) & \cdots \\
\vdots & \ddots \\
\beta i + 1
\end{pmatrix}
\hat{u}_m = \mathcal{F} f_m
\]

where

\[
f_m = f(\theta_m) = \begin{pmatrix}
f(\theta_1) \\
\vdots \\
f(\theta_m)
\end{pmatrix}, \text{ and}
\]

\[
\hat{u}_m \approx \begin{pmatrix}
\hat{u}_\alpha \\
\vdots \\
\hat{u}_\beta
\end{pmatrix}
\]

• In this simple case, convergence follows from convergence of \( f_m \)
• An alternative is to represent the operator as acting on function values:

\[
\mathcal{F}^{-1} \begin{pmatrix} \alpha i + 1 \\ \cdots \\ \beta i + 1 \end{pmatrix} \mathcal{F}u_m = f_m
\]

where

\[u_m \approx u(\theta_m)\]

• Which method is better? Let's compare
Matrix density

coefficient space
$O(m)$ operations

value space
$O(m^3)$ operations
• We introduce some notation

• The *differentiation operator* is the bi-infinite operator

\[
\mathcal{D} = \begin{pmatrix}
\ddots & -i \\
& 0 \\
& i & \ddots
\end{pmatrix}
\]
• We introduce some notation

• The *differentiation operator* is the bi-infinite operator

\[
\mathcal{D} = \begin{pmatrix}
\cdots & -i & 0 \\
0 & i & \cdots \\
\end{pmatrix}
\]

• The *truncation operator* maps bi-infinite vectors to finite vectors:

\[
P_m \hat{\mathbf{u}} = \begin{pmatrix}
\hat{u}_\alpha \\
\vdots \\
\hat{u}_\beta
\end{pmatrix}
\]
• We introduce some notation

• The **differentiation operator** is the bi-infinite operator

\[
\mathcal{D} = \begin{pmatrix}
  \ddots & -i \\
  & 0 \\
  & i \\
  \ddots & \ddots
\end{pmatrix}
\]

• The **truncation operator** maps bi-infinite vectors to finite vectors:

\[
\mathcal{P}_m \hat{\mathbf{u}} = \begin{pmatrix}
  u_\alpha \\
  \vdots \\
  u_\beta
\end{pmatrix}
\]

• The truncated **differentiation matrix** is:

\[
\mathcal{D}_m = \mathcal{P}_m \mathcal{D} \mathcal{P}_m^\top = \begin{pmatrix}
  i\alpha \\
  \ddots \\
  \ddots \\
  i\beta
\end{pmatrix}
\]
• We want to now incorporate variable coefficients, i.e., equations of the form

\[ u' + a(\theta)u = f(\theta) \]

• In value space this is:

\[
\left[ F^{-1}D_m F + \begin{pmatrix} a(\theta_1) \\ \vdots \\ a(\theta_m) \end{pmatrix} \right] u_m = f_m
\]

where \( D_m \) is the discretization of the differentiation operator:

\[
D_m = \begin{pmatrix} i\alpha \\ \vdots \\ i\beta \end{pmatrix}.
\]
• We want to now incorporate variable coefficients, i.e., equations of the form

\[ u' + a(\theta)u = f(\theta) \]

• In value space this is:

\[
\left[ \mathcal{F}^{-1} \mathcal{D}_m \mathcal{F} + \begin{pmatrix} a(\theta_1) \\ \vdots \\ a(\theta_m) \end{pmatrix} \right] u_m = f_m
\]

where \( \mathcal{D}_m \) is the discretization of the differentiation operator:

\[
\mathcal{D}_m = \begin{pmatrix} i\alpha \\ \vdots \\ i\beta \end{pmatrix}
\]

• In coefficient space we have:

\[
\left[ \mathcal{D}_m + \mathcal{F} \begin{pmatrix} a(\theta_1) \\ \vdots \\ a(\theta_m) \end{pmatrix} \mathcal{F}^{-1} \right] \hat{u}_m = \mathcal{F} f_m
\]
Matrix density

Coefficient space
\[ O(C \cdot m) \] operations

Value space
\[ O(m^3) \] operations
Multiplication of Fourier series
• We want to simplify

\[ \mathcal{F} \begin{pmatrix} a(\theta_1) \\ \vdots \\ a(\theta_m) \end{pmatrix} \mathcal{F}^{-1}, \]

i.e., we want to investigate multiplication in coefficient space.
• We want to simplify
\[
\mathcal{F} \left( \begin{array}{c}
  a(\theta_1) \\
  \vdots \\
  a(\theta_m)
\end{array} \right) \mathcal{F}^{-1},
\]
i.e., we want to investigate multiplication in coefficient space

• Let’s consider \( a(\theta) = e^{-i\theta} \): we have
\[
e^{-i\theta} \sum_{k=-\infty}^{\infty} \hat{f}_k e^{ik\theta} = \sum_{k=-\infty}^{\infty} \hat{f}_{k+1} e^{ik\theta}
\]
• We want to simplify

\[ \mathcal{F} \left( \begin{pmatrix} a(\theta_1) \\ \vdots \\ a(\theta_m) \end{pmatrix} \right) \mathcal{F}^{-1}, \]

i.e., we want to investigate multiplication in coefficient space.

• Let's consider \( a(\theta) = e^{-i\theta} \): we have

\[ e^{-i\theta} \sum_{k=-\infty}^{\infty} \hat{f}_k e^{ik\theta} = \sum_{k=-\infty}^{\infty} \hat{f}_{k+1} e^{ik\theta} \]

• Thus in operator space we have the shift operator:

\[ S = \begin{pmatrix} \vdots & & & \vdots \\ & 1 & 0 & 1 \\ & 0 & 1 & 1 \end{pmatrix}, \text{ so that } S \begin{pmatrix} \hat{f}_{-1} \\ \hat{f}_0 \\ \hat{f}_1 \end{pmatrix} = \begin{pmatrix} \hat{f}_{-1} \\ \hat{f}_0 \\ \hat{f}_1 \end{pmatrix} \]
• More generally, if $a(\theta) = e^{ij\theta}$: we have

$$e^{-ij\theta} \sum_{k=-\infty}^{\infty} \hat{f}_k e^{ik\theta} = \sum_{k=-\infty}^{\infty} \hat{f}_{k+j} e^{ik\theta}$$
More generally, if \( a(\theta) = e^{ij\theta} \): we have

\[
e^{-ij\theta} \sum_{k=-\infty}^{\infty} \hat{f}_k e^{ik\theta} = \sum_{k=-\infty}^{\infty} \hat{f}_{k+j} e^{ik\theta}
\]

\( j \) times

This is equivalent to \( e^{-i\theta} \cdots e^{-i\theta} \), i.e.,

\[
S^j = \begin{pmatrix}
0 & 0 & \cdots & 0 & 1 \\
0 & 0 & \cdots & 0 & j \text{ times}
\end{pmatrix}
\]
• For general, smooth and periodic \( a \) we can expand in Fourier series:

\[
a(\theta) = \sum_{k=-\infty}^{\infty} \hat{a}_k e^{ik\theta}
\]

• Each term is multiplication by \( e^{ij\theta} \)

• Thus, in operator space we have the multiplication operator

\[
L[a] = \sum_{k=-\infty}^{\infty} \hat{a}_k S^{-k} = \begin{pmatrix}
\hat{a}_0 & \hat{a}_{-1} & \hat{a}_{-2} \\
\hat{a}_{1} & \hat{a}_0 & \hat{a}_{-1} \\
\hat{a}_{2} & \hat{a}_{1} & \hat{a}_0 \\
\end{pmatrix}
\]

• A bi-infinite matrix with constant diagonals is called a \textit{Laurent operator}
• In practice, we can truncate $a$:

$$a(\theta) \approx \sum_{k=\alpha}^{\beta} \hat{a}_k e^{ik\theta}$$

• Thus, in operator space the multiplication operator is **banded**

• For example, if $\alpha = \beta = 1$ we have

\[
L[a] = \begin{pmatrix}
\hat{a}_0 & \hat{a}_{-1} & \hat{a}_{-2} \\
\hat{a}_1 & \hat{a}_0 & \hat{a}_{-1} \\
\hat{a}_2 & \hat{a}_1 & \hat{a}_0 \\
\end{pmatrix} = \begin{pmatrix}
\hat{a}_0 & \hat{a}_{-1} & \hat{a}_{-2} \\
\hat{a}_1 & \hat{a}_0 & \hat{a}_{-1} \\
\hat{a}_2 & \hat{a}_1 & \hat{a}_0 \\
\end{pmatrix}
\]
Finally, we truncate the multiplication operator. Thus the operator

\[ u' + a(\theta)u = f \]

becomes

\[ (\mathcal{D}_m + L_m[a]) \hat{u}_m = \mathcal{F} f_m \]

where

\[ L_m[a] = \mathcal{P}_m L[a] \mathcal{P}_m^\top = \begin{pmatrix}
\hat{a}_0 & \hat{a}_{-1} \\
\hat{a}_1 & \hat{a}_0 & \hat{a}_{-1} \\
& \ddots & \ddots & \ddots \\
& & \hat{a}_1 & \hat{a}_0 & \hat{a}_{-1} \\
& & \hat{a}_1 & \hat{a}_0
\end{pmatrix} \]
Higher order equations
• The operator

\[ a^{(N)}(\theta)u^{(N)} + \cdots + a^{(0)}u \]

in coefficient space is

\[ L[a^{(N)}]D^N + \cdots L[a^{(0)}] \]

• We can truncate it by

\[ L_m[a^{(N)}]D^N_m + \cdots L_m[a^{(0)}] \]

• However, we need to impose **boundary conditions** so that the operator is uniquely invertible

• In finite dimensions, non-unique inverse causes **bad conditioning**
• We denote the **boundary conditions** as an $d \times \infty$ operator:

$$\mathcal{B}\hat{u}$$

• An example: suppose we want to impose Neumann conditions at the point $\chi$:

$$u(\chi) = u'(\chi) = 0$$

  – Then we have the boundary condition operator

$$\mathcal{B} = \begin{pmatrix} \cdots & e^{-i2\chi} & e^{-ix} & 1 & e^{ix} & e^{2ix} & \cdots \\ \cdots & -2ie^{-i2\chi} & -ie^{-ix} & 0 & ie^{ix} & 2ie^{2ix} & \cdots \end{pmatrix}$$
• The differential equation

\[ \mathcal{B}u = c \quad \text{and} \quad a^{(N)}(\theta)u^{(N)} + \cdots + a^{(0)}(\theta)u = f(\theta) \]

in coefficient space is a pair of equations:

\[ \mathcal{B}\hat{u} = c \quad \text{and} \]

\[ \left( L[a^{(N)}]\mathcal{D}^N + \cdots + L[a^{(0)}] \right) \hat{u} = \hat{f} \]
• The differential equation

\[ \mathcal{B}u = c \quad \text{and} \quad a^{(N)}(\theta)u^{(N)} + \cdots + a^{(0)}(\theta)u = f(\theta) \]

in coefficient space is a pair of equations:

\[ \mathcal{B}\hat{u} = c \quad \text{and} \quad \left( L[a^{(N)}]D^N + \cdots + L[a^{(0)}] \right)\hat{u} = \hat{f} \]

• We need a square matrix, so we now truncate it as

\[ \begin{pmatrix} \mathcal{B}\mathcal{P}_m^\top \\ \mathcal{P}_{m-d} (L[a^{(N)}]D^N + \cdots + L[a^{(0)}]) \mathcal{P}_m^\top \end{pmatrix} \hat{u}_m = \begin{pmatrix} c \\ \mathcal{F}f_{m-d} \end{pmatrix} \]
Taylor series spectral methods
• For Taylor series, the vector of coefficients becomes only infinite in one direction

\[
\hat{\mathbf{u}} = \begin{pmatrix}
\hat{u}_0 \\
\hat{u}_1 \\
\vdots
\end{pmatrix}
\]

• Differentiation becomes

\[
D = \begin{pmatrix}
0 & 1 \\
2 & 3 \\
\vdots & \vdots
\end{pmatrix}
\]

• Truncation is now defined by

\[
P_m\hat{\mathbf{u}} = \begin{pmatrix}
\hat{u}_0 \\
\vdots \\
\hat{u}_{m-1}
\end{pmatrix}
\]
• Multiplication by $z^j$ is again a shift operator:

$$T[z^j] = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 1 & \ddots \end{pmatrix}$$
• Multiplication by $z^j$ is again a shift operator:

$$T[z^j] = \begin{pmatrix}
0 \\
\vdots \\
0 \\
1 \\
\end{pmatrix}^j$$

• Thus multiplication by $a(z)$ becomes a Toeplitz operator

$$T[a] = \begin{pmatrix}
\hat{a}_0 \\
\hat{a}_1 \\
\hat{a}_2 \\
\vdots \\
\end{pmatrix}$$
The differential equation

\[ a^{(N)}(z) u^{(N)} + \cdots + a^{(0)}(z) u = f(z) \quad \text{and} \quad B u = c \]

becomes

\[
\left( \begin{array}{c} \mathcal{B} \mathcal{P}_{m}^{\top} \\ \mathcal{P}_{m-d} \left( T[a^{(N)}] \mathcal{D}^{N} + \cdots + T[a^{(0)}] \right) \mathcal{P}_{m}^{\top} \end{array} \right) \hat{u}_m = \left( \begin{array}{c} c \\ T f_{m-d} \end{array} \right)
\]
• Example: the Airy function \( \text{Ai}(z) \) satisfies

\[
\text{Ai}''(z) = z\text{Ai}(z) \quad \text{and} \quad \text{Ai}(0) = \frac{1}{3^{2/3} \Gamma^{2/3}}, \quad \text{Ai}'(0) = \frac{1}{3^{1/3} \Gamma^{1/3}}
\]
• Example: the Airy function \( \text{Ai}(z) \) satisfies

\[
\text{Ai}''(z) = z\text{Ai}(z) \quad \text{and} \quad \text{Ai}(0) = \frac{1}{3^{2/3}\Gamma^{2/3}}, \text{Ai}'(0) = \frac{1}{3^{1/3}\Gamma^{1/3}}
\]

• The operator becomes

\[
\begin{pmatrix}
B \\
D^2 + T[z]
\end{pmatrix} = 
\begin{pmatrix}
1 & 1 & 2 \\
1 & 6 & 12 \\
\vdots & \vdots & \vdots
\end{pmatrix}
\]
Example: the *Airy function* \( \text{Ai}(z) \) satisfies

\[
\text{Ai}''(z) = z\text{Ai}(z) \quad \text{and} \quad \text{Ai}(0) = \frac{1}{3^{2/3}\Gamma\left(\frac{2}{3}\right)}, \quad \text{Ai}'(0) = \frac{1}{3^{1/3}\Gamma\left(\frac{1}{3}\right)}
\]

The operator becomes

\[
\begin{pmatrix}
B \\
D^2 + T[z]
\end{pmatrix} = \begin{pmatrix}
1 & 1 & 1 & 1 & \ldots \\
1 & 2 & 6 & \ldots & \ldots \\
1 & 6 & 12 & \ldots & \ldots \\
\vdots & \vdots & \vdots & \ddots & \ddots \\
1 & (m-1)! & \ldots & \ldots & \ldots
\end{pmatrix}
\]

We truncate the operator and solve the associated equation:

\[
\begin{pmatrix}
1 & 1 & 1 & 1 & \ldots \\
1 & 2 & 6 & \ldots & \ldots \\
1 & 6 & 12 & \ldots & \ldots \\
\vdots & \vdots & \vdots & \ddots & \ddots \\
1 & (m-1)! & \ldots & \ldots & \ldots
\end{pmatrix} \hat{u}_m = \begin{pmatrix}
\text{Ai}(0) \\
\text{Ai}'(0) \\
0 \\
\vdots \\
0
\end{pmatrix}
\]
Convergence at 1
Convergence at 1

Computed Taylor coefficients
Convergence at 1

Computed Taylor coefficients

Convergence at 2
Chebyshev Spectral Methods
• Recall from last lecture we found a fast way of computing a derivative of function

• In other words, we can apply the differentiation operator $\mathcal{D} \hat{u}$ efficiently

• On the other hand, the differentiation operator itself is dense:

$$
\mathcal{D} = \begin{pmatrix}
0 & 1 & 3 & 5 & \cdots \\
4 & 8 & 12 & \cdots \\
6 & 10 & 12 & \cdots \\
8 & 12 & \cdots & \cdots \\
10 & 12 & \cdots & \cdots & \cdots \\
& & & & & \ddots
\end{pmatrix}
$$

• Thus imposing the equation in coefficient space loses efficiency, and hence we will focus on constructing the equation in value space

  – Next lecture we will adapt the approach to regain banded matrices
• Recall the discrete cosine transform $\mathcal{C}$ which maps function values at Chebyshev points $f_n = f(x_n)$ to (approximate) Chebyshev coefficients

$$\mathcal{C} f_n \approx \begin{pmatrix} \hat{f}_0 \\ \vdots \\ \hat{f}_{n-1} \end{pmatrix}$$
• Recall the **discrete cosine transform** $\mathcal{C}$ which maps function values at Chebyshev points $f_n = f(x_n)$ to (approximate) Chebyshev coefficients

$$
\mathcal{C} f_n \approx \begin{pmatrix} \tilde{f}_0 \\ \vdots \\ \tilde{f}_{n-1} \end{pmatrix}
$$

• Using this, we can represent differentiation in value space by

$$
D_n = \mathcal{C}^{-1} D_n \mathcal{C}
$$

• While this can be slow to generate for each $n$, the computational can be **reused** for multiple ODEs
• The second-order differential operator

\[
a^{(2)}(x)u'' + a^{(1)}(x)u' + a^{(0)}(x)u
\]

becomes

\[
\left[ \text{diag}(a^{(2)}(x_n))D_n^2 + \text{diag}(a^{(1)}(x_n))D_n + \text{diag}(a^{(0)}(x_n)) \right] u_n
\]
• The second-order differential operator

\[ a^{(2)}(x)u'' + a^{(1)}(x)u' + a^{(0)}(x)u \]

becomes

\[
\left[ \text{diag}(a^{(2)}(x_n))D_n^2 + \text{diag}(a^{(1)}(x_n))D_n + \text{diag}(a^{(0)}(x_n)) \right] u_n
\]

• There's a problem: which rows do we drop to incorporate the boundary conditions?
• The second-order differential operator

\[ a^{(2)}(x)u'' + a^{(1)}(x)u' + a^{(0)}(x)u \]

becomes

\[ \left[ \text{diag}(a^{(2)}(x_n))D_n^2 + \text{diag}(a^{(1)}(x_n))D_n + \text{diag}(a^{(0)}(x_n)) \right] u_n \]

• There's a problem: which rows do we drop to incorporate the boundary conditions?

• The typical boundary conditions are at the endpoints \( \pm 1 \)
  
  – For example, Dirichlet conditions are

\[ B u_n = \begin{pmatrix} e_1^T \\ e_{-1}^T \end{pmatrix} u_n = \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix} u_n = \begin{pmatrix} u(x_1) \\ u(x_n) \end{pmatrix} \]

• Thus the standard approach is to drop the boundary rows
The second-order differential equation
\[ a^{(2)}(x)u'' + a^{(1)}(x)u' + a^{(0)}(x)u = f(x) \]
and \( B u = c \)
becomes
\[
\left( I_{2:n-1} \left[ \text{diag}(a^{(2)}(x_n))D_n^2 + \text{diag}(a^{(1)}(x_n))D_n + \text{diag}(a^{(0)}(x_n)) \right] \right) u_n = \begin{pmatrix} c \\ f_n \end{pmatrix}
\]
for the \( n - 2 \times n \) boundary row dropping matrix
\[
I_{2:n-1} = \begin{pmatrix} 0 & 1 & & \\ & & \ddots & \\ & & & 1 & 0 \end{pmatrix}
\]