

A fast and well-conditioned spectral method: The US method

Alex Townsend
University of Oxford

Leslie Fox Prize, 24th of June 2013



Partially based on: S. Olver & T., A fast and well-conditioned spectral method, to appear in *SIAM Review*, (2013).

Problem: Solve linear ordinary differential equations (ODEs)

$$a^N(x) \frac{d^N u}{dx^N} + \cdots + a^1(x) \frac{du}{dx} + a^0(x) u = f(x), \quad x \in [-1, 1]$$

with K linear boundary conditions (Dirichlet, Neumann, $\int_{-1}^1 u = c$, etc.)

Assumptions:

- a^0, \dots, a^N, f are continuous with bounded variation.
- The leading term $a^N(x)$ is non-zero on the interval.
- The solution exists and is unique.

Main philosophy: Replace a^0, \dots, a^N, f by polynomial approximations, and solve for a polynomial approximation for u . The spectral method finds the coefficients in a **Chebyshev series** in $\mathcal{O}(m^2 n)$ operations:

$$u(x) \approx \sum_{k=0}^{n-1} \alpha_k T_k(x), \quad T_k(x) = \cos(k \cos^{-1} x).$$

Conventional wisdom

A comparison

First-order ODEs

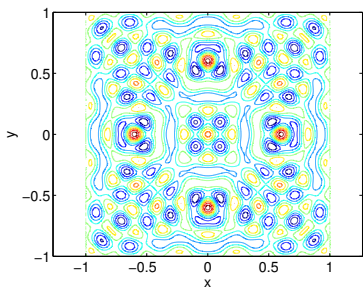
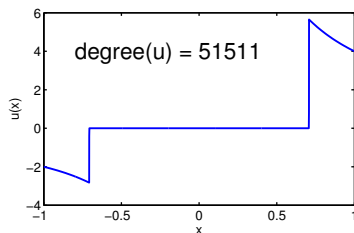
Higher-order ODEs

Stability and convergence

Fast linear algebra

PDEs

Conclusion



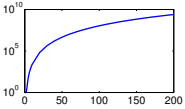
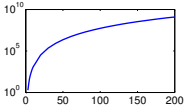
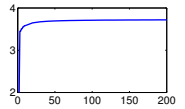
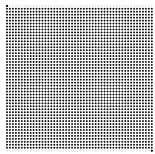
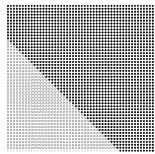
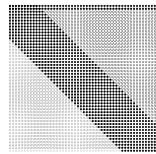
"It is known that matrices generated by spectral methods can be dense and ill-conditioned." [Chen 2005]

"The idea behind spectral methods is to take this process to the limit, at least in principle, and work with a differentiation formula of infinite order and infinite bandwidth, i.e., a dense matrix." [Trefethen 2000]

"The best advice is still this: use pseudospectral (collocation) methods instead of spectral (coefficient), and use Fourier series and Chebyshev polynomials in preference to more exotic functions." [Boyd 2003]

The Ultraspherical spectral method (US method) can result in almost banded, well-conditioned linear systems.

$$u''(x) + \cos(x)u(x) = 0, \quad u(\pm 1) = 0.$$

	Collocation	Coefficients	US method
Condition number			
Density			

	Collocation	Coefficient	US method
Matrix structure	Dense	Banded from below	Banded + rank-K
Condition number	$\mathcal{O}(n^{2N})$	$\mathcal{O}(n^{2N})$	$\mathcal{O}(n^{2(D-1)})$
Problem sizes	$n \leq 5,000$	$n \leq 5,000$	$n \leq 70,000$
Operation count	$\mathcal{O}(n^3)$	$\mathcal{O}(mn^2)$	$\mathcal{O}(m^2n)$

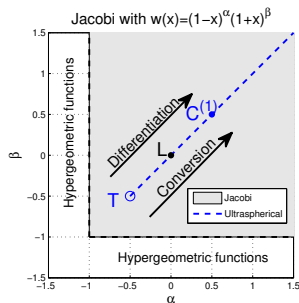
Differentiation
Operator

Multiplication
Operator

$$u'(x) + a(x)u(x) = f(x)$$

$$(\mathcal{D} + \mathcal{SM}[a]) \mathbf{u} = \mathcal{S}f$$

- $\mathcal{D} : \mathbf{T} \rightarrow \mathbf{C}^{(1)}$
- $\mathcal{M}[a] : \mathbf{T} \rightarrow \mathbf{T}$
- $\mathcal{S} : \mathbf{T} \rightarrow \mathbf{C}^{(1)}$



Differentiation operator: $\mathcal{D} : \mathbf{T} \rightarrow \mathbf{C}^{(1)}$, [DLMF]

$$\frac{dT_k}{dx} = \begin{cases} kC_{k-1}^{(1)}, & k \geq 1, \\ 0, & k = 0, \end{cases} \quad \mathcal{D} = \begin{pmatrix} 0 & 1 & & & \\ & & 2 & & \\ & & & 3 & \\ & & & & \ddots \end{pmatrix}.$$

Conversion operator: $\mathcal{S} : \mathbf{T} \rightarrow \mathbf{C}^{(1)}$, [DLMF]

$$T_k = \begin{cases} \frac{1}{2} (C_k^{(1)} - C_{k-2}^{(1)}), & k \geq 2, \\ \frac{1}{2} C_1^{(1)}, & k = 1, \\ C_0^{(1)}, & k = 0, \end{cases} \quad \mathcal{S} = \begin{pmatrix} 1 & 0 & -\frac{1}{2} & & \\ & \frac{1}{2} & 0 & -\frac{1}{2} & \\ & & \frac{1}{2} & 0 & \ddots \\ & & & \ddots & \ddots \end{pmatrix}.$$

Multiplication operator: $\mathcal{M}[a] : \mathbf{T} \rightarrow \mathbf{T}$, [DLMF]

$$T_j T_k = \frac{1}{2} (T_{|j-k|} + T_{j+k}), \quad j, k \geq 0.$$

$$T_j T_k = \frac{1}{2} T_{|j-k|} + \frac{1}{2} T_{j+k}$$

$$\mathcal{M}[\mathbf{a}] = \underbrace{\frac{1}{2} \begin{pmatrix} 2a_0 & a_1 & a_2 & a_3 & \dots \\ a_1 & 2a_0 & a_1 & a_2 & \ddots \\ a_2 & a_1 & 2a_0 & a_1 & \ddots \\ a_3 & a_2 & a_1 & 2a_0 & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots \end{pmatrix}}_{\text{Toeplitz}} + \underbrace{\frac{1}{2} \begin{pmatrix} 0 & 0 & 0 & 0 & \dots \\ a_1 & a_2 & a_3 & a_4 & \ddots \\ a_2 & a_3 & a_4 & a_5 & \ddots \\ a_3 & a_4 & a_5 & a_6 & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots \end{pmatrix}}_{\text{Hankel} + \text{rank-1}}$$

Multiplication is not a dense operator in finite precision. It is **m-banded**:

$$a(x) = \sum_{k=0}^{\infty} a_k T_k(x) = \sum_{k=0}^m \tilde{a}_k T_k(x) + \mathcal{O}(\epsilon),$$

where \tilde{a}_k are aliased Chebyshev coefficients.

- **Dirichlet:** $u(-1) = c$:

$$\mathcal{B} = (1, -1, 1, -1, \dots), \quad T_k(-1) = (-1)^k.$$

- **Neumann:** $u'(1) = c$:

$$\mathcal{B} = (0, 1, 4, 9, \dots), \quad T_k(1) = k^2.$$

- **Other conditions:** $\int_{-1}^1 u = c$, $u(0) = c$, or $u(0) + u'(\pi/6) = c$.

Imposing boundary conditions by boundary bordering:

$$\begin{aligned} u'(x) + a(x)u(x) &= f, \\ \mathcal{B}u &= c, \end{aligned} \quad \begin{pmatrix} \mathcal{B} \\ \mathcal{D} + \mathcal{SM}[\mathbf{a}] \end{pmatrix} = \begin{pmatrix} c \\ \mathbf{f} \end{pmatrix}.$$

The finite section can be done exactly by projection $\mathcal{P}_n = (I_n, \mathbf{0})$:

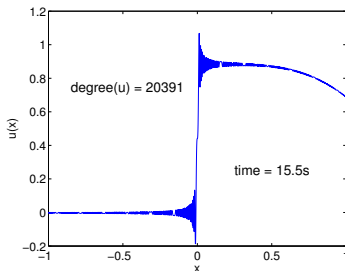
$$\begin{pmatrix} \mathcal{BP}_n^T \\ \mathcal{P}_{n-1}\mathcal{D}\mathcal{P}_n^T + \mathcal{P}_{n-1}\mathcal{SP}_{n+m}^T\mathcal{P}_{n+m}\mathcal{M}[\mathbf{a}]\mathcal{P}_n^T \end{pmatrix} = \begin{pmatrix} c \\ \mathcal{P}_{n-1}\mathbf{f} \end{pmatrix}.$$

Example 1

$$u'(x) + x^3 u(x) = 100 \sin(20,000x^2), \quad u(-1) = 0.$$

The exact solution is

$$u(x) = e^{-\frac{x^4}{4}} \left(\int_{-1}^x 100 e^{\frac{t^4}{4}} \sin(20,000t^2) dt \right).$$



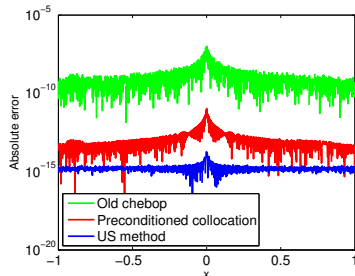
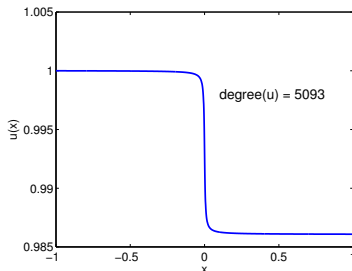
- `N = ultraop(@(x,u) ...);`
`N.lbc = 0; u = N \ f;`
- Adaptively selects the discretisation size.
- Forms a chebfun object [Chebfun V4.2].
- $\|\tilde{u} - u\|_{\infty} = 1.5 \times 10^{-15}.$

Example 2

$$u'(x) + \frac{1}{1 + 50,000x^2} u(x) = 0, \quad u(-1) = 1.$$

The exact solution with $a = 50,000$ is

$$u(x) = \exp\left(-\frac{\tan^{-1}(\sqrt{a}x) + \tan^{-1}(\sqrt{a})}{\sqrt{a}}\right).$$



Higher-order diff operators: $\mathcal{D}_\lambda : \mathbf{T} \rightarrow \mathbf{C}^{(\lambda)}$, [DLMF]

$$\frac{dC_k^{(\lambda)}}{dx} = \begin{cases} 2\lambda C_{k-1}^{(\lambda+1)}, & k \geq 1, \\ 0, & k = 0, \end{cases} \quad \mathcal{D}_\lambda = 2^{\lambda-1}(\lambda-1)! \begin{pmatrix} \overbrace{0 \ \dots \ 0}^{\lambda \text{ times}} & \lambda & & \\ & \lambda+1 & & \\ & & \ddots & \\ & & & \ddots \end{pmatrix}.$$

Higher-order conversion operators: $\mathcal{S}_\lambda : \mathbf{C}^{(\lambda)} \rightarrow \mathbf{C}^{(\lambda+1)}$, [DLMF]

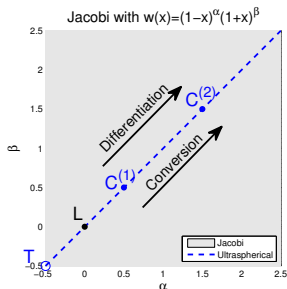
$$C_k^{(\lambda)} = \begin{cases} \frac{\lambda}{\lambda+k} (C_k^{(\lambda+1)} - C_{k-2}^{(\lambda+1)}), & k \geq 2, \\ \frac{\lambda}{\lambda+1} C_1^{(\lambda+1)}, & k = 1, \\ C_0^{(\lambda+1)}, & k = 0, \end{cases} \quad \mathcal{S}_\lambda = \begin{pmatrix} 1 & & -\frac{\lambda}{\lambda+2} & & \\ & \frac{\lambda}{\lambda+1} & & -\frac{\lambda}{\lambda+3} & \\ & & \ddots & & \ddots \end{pmatrix}.$$

Multiplication operators:

$$\mathcal{M}_\lambda[\mathbf{a}] : \mathbf{C}^\lambda \rightarrow \mathbf{C}^\lambda,$$

$$\mathcal{M}_1[\mathbf{a}] : \mathbf{C}^{(1)} \rightarrow \mathbf{C}^{(1)},$$

$$\mathcal{M}_1[\mathbf{a}] = \text{Toeplitz} + \text{Hankel}.$$



$\mathcal{M}_\lambda[\mathbf{a}] : \mathbf{C}^{(\lambda)} \rightarrow \mathbf{C}^{(\lambda)}$ represents multiplication with $a(x)$ in $C^{(\lambda)}$. If

$$a(x) = \sum_{j=0}^m a_j C_j^{(\lambda)}(x),$$

then it can be shown that $\mathcal{M}_\lambda[a]$ is **m -banded** with

$$(\mathcal{M}_\lambda[a])_{j,k} = \sum_{s=\max(0,k-j)}^k a_{2s+j-k} c_s^\lambda(k, 2s+j-k), \quad j, k \geq 0,$$

where (to prevent overflow issues) we have

$$\begin{aligned} c_s^\lambda(j, k) &= \frac{j+k+\lambda-2s}{j+k+\lambda-s} \times \prod_{t=0}^{s-1} \frac{\lambda+t}{1+t} \times \prod_{t=0}^{j-s-1} \frac{\lambda+t}{1+t} \\ &\quad \times \prod_{t=0}^{s-1} \frac{2\lambda+j+k-2s+t}{\lambda+j+k-2s+t} \times \prod_{t=0}^{j-s-1} \frac{k-s+1+t}{k-s+\lambda+t}. \end{aligned}$$

For efficiency, we use a recurrence relation to generate $c_s^\lambda(j, k)$.

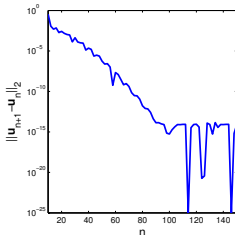
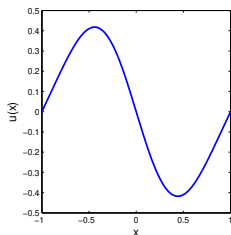
$$a^N(x) \frac{d^N u}{dx^N} + a^{N-1}(x) \frac{d^{N-1} u}{dx^{N-1}} + \cdots + a^0(x) u = f(x), \quad \mathcal{B}u = \mathbf{c},$$

$$\left(\mathcal{M}_N[\mathbf{a}^N] \mathcal{D}_N + \mathcal{S}_{N-1} \mathcal{M}_{N-1}[\mathbf{a}^{N-1}] \mathcal{D}_{N-1} + \cdots + \mathcal{S}_{N-1} \cdots \mathcal{S}_0 \mathcal{M}_0[\mathbf{a}^0] \right) = \left(\mathcal{S}_{N-1} \cdots \mathcal{S}_0 \mathbf{c} \right)$$

Example 3: High-order ODE

$$u^{(10)}(x) + \cosh(x) u^{(8)}(x) + \cos(x) u^{(2)}(x) + x^2 u(x) = 0$$

$$u(\pm 1) = 0, \quad u'(\pm 1) = 1, \quad u^{(k)}(\pm 1) = 0, \quad k = 2, 3, 4.$$

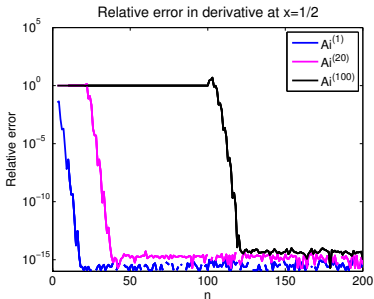
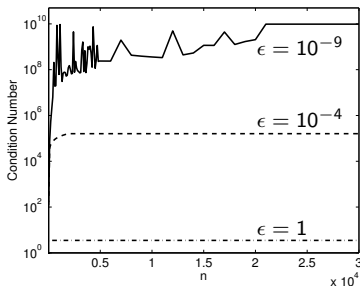


$$\left(\int_{-1}^1 (\tilde{u}(x) + \tilde{u}(-x))^2 dx \right)^{\frac{1}{2}} = 1.3 \times 10^{-14}.$$

Example 4

Consider Airy's equation for $\epsilon > 0$,

$$\epsilon u''(x) - xu(x) = 0, \quad u(-1) = \text{Ai}\left(-\sqrt[3]{1/\epsilon}\right), \quad u(1) = \text{Ai}\left(\sqrt[3]{1/\epsilon}\right).$$



The US method can accurately compute high derivatives of functions. For a different approach, see [Bornemann 2011].

Assumption: Number of boundary conditions = Differential order.

Definition: Higher-order ℓ_λ^2 -norms

For $\lambda \in \mathbb{N}$, $\ell_\lambda^2 \subset \mathbb{C}^\infty$ is the Banach space with norm

$$\|\mathbf{u}\|_{\ell_\lambda^2}^2 = \sum_{k=0}^{\infty} |u_k|^2 (1+k)^{2\lambda} < \infty.$$

Right diagonal preconditioner:

$$\mathcal{R} = \frac{1}{2^{N-1}(N-1)!} \begin{pmatrix} \mathbf{I}_N & & & \\ & \frac{1}{N} & & \\ & & \frac{1}{N+1} & \\ & & & \ddots \end{pmatrix}.$$

Recall the notation:

$$\begin{aligned} \text{Solve } \mathcal{L}u &= f, \\ \text{with } \mathcal{B}u &= \mathbf{c}, \quad \mathcal{B} : \ell_D^2 \rightarrow \mathbb{C}^K, \text{ bounded, } D \geq 1. \end{aligned}$$

Theorem: Condition number is bounded with n

Let $\begin{pmatrix} \mathcal{B} \\ \mathcal{L} \end{pmatrix} : \ell_{\lambda+1}^2 \rightarrow \ell_{\lambda}^2$ be an invertible operator for some $\lambda \geq D - 1$.

Then, as $n \rightarrow \infty$,

$$\|A_n R_n\|_{\ell_{\lambda}^2} = \mathcal{O}(1), \quad \|(A_n R_n)^{-1}\|_{\ell_{\lambda}^2} = \mathcal{O}(1),$$

where

$$R_n = \mathcal{P}_n^T \mathcal{R} \mathcal{P}_n, \quad A_n = \mathcal{P}_n^T \begin{pmatrix} \mathcal{B} \\ \mathcal{L} \end{pmatrix} \mathcal{P}_n.$$

- Proof uses integral reformulation ideas: “ \mathcal{R} integrates the ODE to form a Fredholm operator”, i.e.,

$$\begin{pmatrix} \mathcal{B} \\ \mathcal{L} \end{pmatrix} \mathcal{R} = \mathcal{I} + \mathcal{K}, \quad \mathcal{K} = \text{compact}.$$

Theorem: Computed solution converges in high-order norms

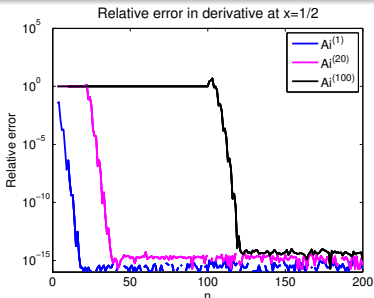
Suppose $\mathbf{f} \in \ell_{\lambda-N+1}^2$ for some $\lambda \geq D - 1$, and that $\begin{pmatrix} \mathcal{B} \\ \mathcal{L} \end{pmatrix} : \ell_{\lambda+1}^2 \rightarrow \ell_{\lambda}^2$ is an invertible operator. Define

$$\mathbf{u}_n = A_n^{-1} \mathcal{P}_n \begin{pmatrix} \mathbf{c} \\ \mathcal{S}_{N-1} \cdots \mathcal{S}_0 \mathbf{f} \end{pmatrix},$$

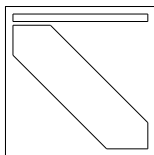
then

$$\|\mathbf{u} - \mathcal{P}_n^\top \mathbf{u}_n\|_{\ell_{\lambda+1}^2} \leq C \|\mathbf{u} - \mathcal{P}_n^\top \mathcal{P}_n \mathbf{u}\|_{\ell_{\lambda+1}^2}.$$

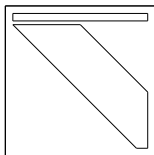
- The constant, C , does not depend on n .
- You don't always need the complex plane to compute high derivatives.



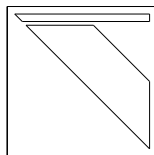
Original matrix:



After left Givens:



After right Givens:



- Apply a partial factorisation on the left, followed by a partial factorisation on the right.
- Factorisation: $A_n = QTP^*$ in $\mathcal{O}(m^2n)$ operations.
- No fill-in: The matrix T contains no more non-zeros than A_n .
- For solving $A_n x = b$: The matrix Q can be immediately applied to b . The matrix P^* can be stored using Demmel's trick [Demmel 1997].
- UMFPACK by Davis is faster for $n \leq 50000$, but has observed complexity of, roughly, $\mathcal{O}(n^{1.25})$ with a very small constant.

Consider the constant coefficient PDE

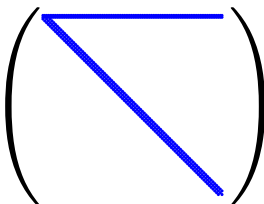
$$\sum_{j=0}^N \sum_{i=0}^{N-j} a_{ij} \frac{\partial^{i+j} u}{\partial^i x \partial^j y} = f(x, y), \quad a_{ij} \in \mathbb{R}$$

defined on $[a, b] \times [c, d]$ with linear boundary conditions satisfying **continuity conditions**.

We discretise as a generalised Sylvester matrix equation

$$\sum_{j=1}^k \sigma_j A_j X B_j^T = F, \quad A_j \in \mathbb{R}^{n_1 \times n_1}, \quad B_j \in \mathbb{R}^{n_2 \times n_2},$$

where A_j and B_j are US discretisations, for example,

$$A_j = \begin{pmatrix} \text{---} \\ \text{---} \end{pmatrix}$$


Given a PDE with constant coefficients of the form,

$$\mathcal{L} = \sum_{j=0}^N \sum_{i=0}^{N-j} a_{ij} \frac{\partial^{i+j}}{\partial x^i \partial y^j}, \quad A = (a_{ij}).$$

We can rewrite this as

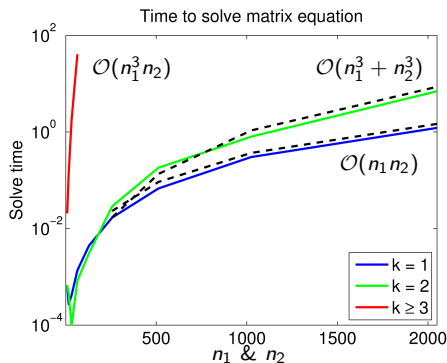
$$\mathcal{L} = \mathfrak{D}(y)^T A \mathfrak{D}(x), \quad \mathfrak{D}(x) = \begin{pmatrix} \partial^0 / \partial x^0 \\ \vdots \\ \partial^N / \partial x^N \end{pmatrix}.$$

Hence, if $A = U \Sigma V^T$ is the truncated SVD of A with $\Sigma \in \mathbb{R}^{k \times k}$ then

$$\mathcal{L} = \mathfrak{D}(y)^T (U \Sigma V^T) \mathfrak{D}(x) = (\mathfrak{D}(y)^T U) \Sigma (V^T \mathfrak{D}(x)).$$

This low rank representation for \mathcal{L} tells us how to discretize and solve the PDE.

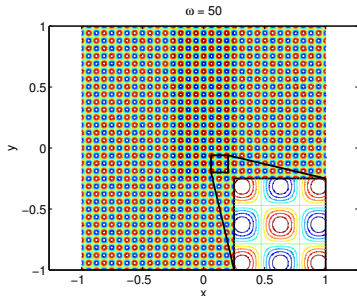
- **Rank-1:** $A_1XB_1^T = F$. Solve $A_1Y = F$, then $B_1X^T = Y^T$.
- **Rank-2:** $A_1XB_1^T + A_2XB_2^T = F$. Generalised Sylvester solver [Jonsson & Kågström, 1992].
- **Rank-k, $k \geq 3$:** Solve $n_1n_2 \times n_1n_2$ system with QTP^* factorization.



- Boundary conditions are imposed by carefully removing degrees of freedom in the matrix equation.
- Automatically forms and solves subproblems, if possible.
- Corner singularities can be partly resolved by domain subdivision.

Example 5: Helmholtz equation

$u_{xx} + u_{yy} + 2\omega^2 u = 0, \quad u(\pm 1, y) = f(\pm 1, y), \quad u(x, \pm 1) = f(x, \pm 1),$
 where $f(x, y) = \cos(\omega x) \cos(\omega y)$.



- `N = chebop2(@(u) ...);`
`N.lbc=f(-1,:); u=N\0;`
- Adaptively selects the discretisation size.
- Forms a `chebfun2` object [T. & Trefethen 2013].
- For $\omega = 2000$,
 $\|\tilde{u} - u\|_2 = 2.43 \times 10^{-9}$.

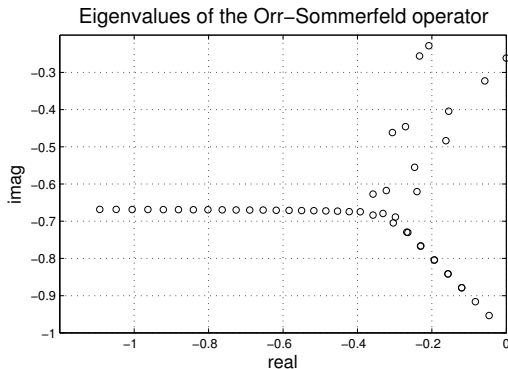
For $\omega = 50$ solve takes 0.27s for $n_1 = n_2 = 126$,
 for $\omega = 2000$ solve takes 32.1s for $n_1 = n_2 = 2124$.

- The US method results in almost banded well-conditioned matrices.
- It requires $\mathcal{O}(m^2n)$ operations to solve an ODE.
- It can be used as a preconditioner for the collocation method.
- The US method converges and is numerically stable.
- It can be extended to a spectral method for solving PDEs on rectangular domains.

Orr-Sommerfeld equation

$$\frac{1}{5772.2}(u'''' - 2u'' + u) - 2iu - i(1 - x^2)(u'' - u) = \lambda(u'' - u),$$







$$u(\pm 1) = u'(\pm 1) = 0.$$



Thank you

More information:

S. Olver & T., A fast and well-conditioned spectral method, to appear in SIAM Review, (2013).

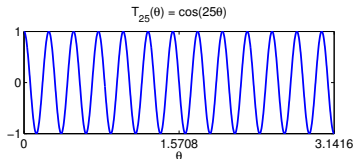
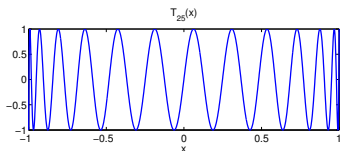
-  F. BORNEMANN, Accuracy and stability of computing high-order derivatives of analytic functions by Cauchy integrals, *Found. Comput. Math.*, 11 (2011), pp. 1–63.
-  I. JONSSON & KÅGSTRÖM, Recursive blocked algorithms for solving triangular matrix equations — Part II: Two-sided and generalized Sylvester and Lyapunov equations, *ACM Trans. Math. Softw.*, 28 (2002), pp. 416–435.
-  S. OLVER & T., A fast and well-conditioned spectral method, *to appear in SIAM Review*, (2013).
-  S. ORSZAG, Accurate solution of the Orr-Sommerfeld stability equation, *J. Fluid Mech.*, 50 (1971), pp. 689–703.
-  T. & L. N. TREFETHEN, An extension of Chebfun to two dimensions, likely to appear in *SISC*, (2013).
-  L. N. TREFETHEN ET AL., *The Chebfun software system, version 4.2*, 2011.

An underappreciated fact: Every Chebyshev series can be rewritten as a palindromic Laurent series.

The degree k Chebyshev polynomial (of the first kind) is defined by

$$T_k(x) = \cos(k \cos^{-1} x) = \frac{1}{2} (z^k + z^{-k}), \quad x \in [-1, 1],$$

where $z = e^{ik\theta}$ and $\cos \theta = x$.



Every Chebyshev series can be written as a palindromic Laurent series:

$$\sum_{k=0}^N \alpha_k T_k(x) = \frac{1}{2} \sum_{k=1}^N \alpha_k z^{-k} + \alpha_0 + \frac{1}{2} \sum_{k=1}^N \alpha_k z^k, \quad z = e^{ik\theta}.$$

A Givens rotation zeros out one entry of a matrix and we can store the rotation information there.

$$\begin{pmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \boxed{\times} & \times & \times \\ \times & \times & \times & \times \end{pmatrix} \xrightarrow{\text{Givens}} \begin{pmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & p & \times & \times \\ \times & \times & \times & \times \end{pmatrix}$$

Let $s = \sin \theta$ and $c = \cos \theta$, where θ is rotation angle, then

$$p = \begin{cases} s \cdot \text{sign}(c), & |s| < |c|, \\ \frac{\text{sign}(s)}{c}, & |s| \geq |c|. \end{cases}$$

To restore (up to a 180 degrees): If $|p| < 1$, then $s = p$ and $c = \sqrt{1 - s^2}$, otherwise $c = 1/p$ and $s = \sqrt{1 - c^2}$.