Met. Mat. Computat. Abril 11, 2024

Práctica IV

1.- This problem demonstrates that slow convergence and the Gibbs phenomemon may occur in orthogonal function expansions other than Chebyshev expansions—and, specifically, for Fourier-Bessel series (see pp. 97 ff. in the class notes). [The calculations indicated in point 1(a) are not required as part of this HW set.]

(a) It is easy to check, on the basis of the series mentioned in class for the Bessel function J_m

$$J_m(t) = \sum_{k=0}^{\infty} \frac{(-1)^k \left(\frac{1}{2}t\right)^{m+2k}}{k!(m+k)!},$$

that

$$\frac{d}{dt}\left[t^{-m}J_m(t)\right] = -t^{-m}J_{m+1}(t) \tag{1}$$

and

$$\frac{d}{dt}\left[t^{m}J_m(t)\right] = t^{m}J_{m-1}(t). \tag{2}$$

(b) Using (1) and (2) verify that

$$\frac{d}{dt}\left[t^2\left\{J_m(t)^2 - J_{m+1}(t)J_{m-1}(t)\right\}\right] = 2tJ_m^2(t). \tag{3}$$

[Hint: Write $\left[t^2\left\{J_m(t)^2 - J_{m+1}(t)J_{m-1}(t)\right\}\right] = t^2(t^m J_m)(t^{-m}J_m) - (t^{m+1}J_{m+1})(t^{-(m-1)}J_{m-1})$.]

(c) Here we consider, for a *fixed* non-negative integer m, the expansion of functions $y = f(x)$ $(0 \leq x \leq 1)$ in Fourier-Bessel series in terms of the m-th Bessel function J_m:

$$f(x) = \sum_{n=1}^{\infty} a_n J_m(j_{m,n}\, x),$$

where $j_{m,n} \in \mathbb{R}$ $(n \in \mathbb{N})$ denote the positive zeroes of the function $J_m(x)$. The evaluation of the expansion coefficients a_n requires, in particular, knowledge of the quantities $\int_0^1 xJ_m^2\left(j_{m,k}x\right)dx$. Using (3) show that

$$\int_0^1 xJ_m^2\left(j_{m,k}x\right)dx = \frac{1}{2}J_{m+1}^2(j_{m,k}). \tag{4}$$

[Hint: use (3) to reduce the integral to boundary terms. Then, use (1) and (2) to see that $J_{m-1}(j_{m,k}) = J_m'(j_{m,k}) = -J_{m+1}(j_{m,k})$.]

(d) Let f_γ $(0 \leq \gamma \leq 1)$ be the function defined in the interval $[0, 1]$ by

$$f_\gamma(x) = \begin{cases} 1 & \text{for } 0 < x < \gamma \\ 0 & \text{for } \gamma < x \leq 1. \end{cases}$$

Using (2) and (4) obtain a closed form expression for the J_m Fourier-Bessel expansion of f_γ with $m = 0$. Plot the approximations obtained from truncated expansions containing N terms, for $\gamma < 1$ and for $\gamma = 1$, and for $N = 50$ and other values of N you deem useful. [Suggestion: Use a matlab function for evaluation of zeroes of Bessel functions, such as, e.g., either `https://www.mathworks.com/matlabcentral/fileexchange/48403-bessel-zero-solver` or `https://www.mathworks.com/matlabcentral/fileexchange/6794-bessel-function-zeros`.] Do you observe a Gibbs phenomenon at $x = \gamma$ (in either or both of the cases $\gamma < 1$ pr $\gamma = 1$), with overshoots of approximately 8.95% of the jump (as shown in class to be the for the trigonometric Fourier series)? Do you observe a Gibbs phenomenon at $x = 0$? Perform a (numerical) convergence analysis to see whether actual Gibbs-like oscillations take place around that point. Note that, in particular, uniform convergence does not take place for functions not vanishing at $x = 1$.

[Note: The Bessel Gibbs phenomenon can be related to the trigonometric Gibbs phenomenon via consideration of the asymptotic relation

$$J_0(x) \sim \left(\frac{2}{\pi x}\right)^{\frac{1}{2}} \cos(x - \pi/4)$$

which is valid as $x \to \infty$. This relationship can be used to explain the similarities between the Bessel-Gibbs and trigonometric-Gibbs phenomena, including the aforementioned 8.95% overshoot.]

2.- We consider once again Burgers' equation

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \varepsilon \frac{\partial^2 u}{\partial x^2},$$

(see STW p. 179).

(i) Verify that Burgers' equation admits the soliton solution

$$u(x,t) = \kappa \left[1 - \tanh\left(\frac{\kappa(x - \kappa t - x_c)}{2\varepsilon}\right)\right], \tag{5}$$

where $\kappa > 0$, and where the "center" x_c is an arbitrary real number $x_c \in \mathbb{R}$. [The details of this calculation need not be included in the solution set.]

(ii) Consider the parameter values $\varepsilon = 0.1$, $\kappa = 0.5$, $x_c = -3$, $x \in [-5,5]$, and impose initial values $u(x,0)$ and boundary conditions $u(\pm 5,t)$ which coincide with those associated with the exact solution (5). Use the Crank-Nicolson leap-frog scheme described in pbm. 2 in Práctica 3, and the Chebyshev collocation method in space to solve the equation. [HINT: Since boundary conditions need to be specified at both boundary points $x = \pm 5$, it is necessary to use the Gauss-Lobatto Chebyshev rule; the associated differentiation matrix is given e.g. in equation (3.228), p. 110, in STW]. Note that rather significant reductions in time-step are necessary to maintain stability as the spatial discretizations are refined—a difficulty that reflects the spatial mesh-refinement near the boundary points that is induced by the Chebyshev method. Evaluate the discrete maximum errors for temporal step-sizes $\Delta t = 10^{-k}$ with $k = 2, 3, 4$ and with a number $N = 32,\ 64,\ 128$ of spatial discretization points at $t = 12$.

(iii) Consider the Burgers' equation in the interval $(-1,1)$ with data $u(\pm 1,t) = 0$, $u(x,0) = -\sin(\pi x)$, $x \in [-1,1]$. Solve this problem by the methods above with $\varepsilon = 0.02$, $\Delta t = 10^{-4}$ and $N = 128$, and plot the numerical solution at $t = 1$. Refer to[1] for some details concerning the numerical solution (obtained by other means).

[1] Shen J, Wang L (2007b) Fourierization of the Legendre-Galerkin method and a new space-time spectral method. Appl Numer Math 57(5-7):710-720

3.-

(a) Consider the text and code (Program 27) concerning the KdV equation in pp. 108–111 of the book "Spectral Methods in Matlab", by N. Trefethen. (The code can be downloaded from the author's website.) As indicated in the text, use of the Runge-Kutta method would give rise to convergent numerical solutions even if the "integrating-factor" trick described in the text were not used. Modify Program 27 in such a way that use of the integrating factor method is eliminated. By experimenting with the resulting code show that the algorithm is still stable and convergent provided sufficiently smaller time-steps are used. Explain why the modified code requires smaller time-steps for stability. (For an alternative related reference see pp. 38–40 of the "Spectral Methods" text by Shen, Tang and Wang.)

(b) Provide a qualitative explanation of the enlarged stability domain that results from use of an integrating factor.

4.- Fourier Continuation: FC(SVD).

(i) Produce a Matlab-based FC(SVD) code, as described in class[2], for accurate FC expansion of a given (generally non-periodic) smooth function f defined in the interval $[0,1]$. [Notes: As discussed in class, FC(SVD) is an expensive and ill-conditioned algorithm (although it works well for certain applications which require up to several thousand representation points). Additionally, the FC(SVD) approach should lend useful intuition concerning the character of the continuation process. And, finally, the FC(SVD) strategy does provide one essential element in the fast and well-conditioned FC(Gram) algorithm described in class.]

As an addition to the class description and the aforementioned article, the following "quick-access" notes may prove useful. Given a smooth function f defined in the interval $[0,1]$, and using the column vectors $x = (x_1, \ldots, x_N)^T$ and $y = (y_1, \ldots, y_N)^T$ ($x_j = (j-1)/(N-1)$ and $y_j = f(x_j)$ for $j = 1, \ldots, N$), the algorithm should produce a b-periodic Fourier expansion ($b > 1$; think $b = 2$) of the form[3]

$$S_M(x) = \sum_{k \in t(M)} a_k e^{\frac{2\pi i}{b} kx} \tag{6}$$

which approximates $f(x)$ closely throughout the interval $[0,1]$. (Note that in the overdetermined case $M < N$, the Fourier series S may not be (generally will not be) interpolatory. This means that we may have $S_M(x_j) \neq y_j$ for some or all j.)

The coefficients a_k are obtained as the *least-squares* solution, based on use of an SVD decomposition[4], of the matrix of the (possibly over-determined) system of linear equations

$$y_j = \sum_{k \in t(M)} a_k e^{\frac{2\pi i}{b} kx_j}, \quad j = 1, \ldots, N. \tag{7}$$

In practice it is useful to arrange the Fourier coefficients in the column vectors

[2] "Accurate, high-order representation of complex three-dimensional surfaces via Fourier-Continuation analysis", O. P. Bruno, Y. Han and M. Pohlman; Journal of Computational Physics **227**, 1094–1125 (2007).

[3] In what follows we let $t(M) = \{j \in \mathbb{N} : -(M-1)/2 \leq j \leq (M-1)/2\}$ for M odd and $t(M) = \{j \in \mathbb{N} : -M/2 \leq j \leq M/2 - 1\}$ for M even.

[4] It is also possible to solve the least squares problem using a QR factorization. The SVD method has been found more accurate and robust in this context, although somewhat more expensive. For an introduction to least squares via the SVD see e.g. Golub and van Loan's "Matrix Computations" monograph, Section 5.5.3.

$a = (a_{-M/2}, \ldots, a_0, \ldots, a_{M/2-1})^T$ for M even and $a = (a_{-(M-1)/2}, \ldots, a_0, \ldots, a_{(M-1)/2})^T$ for M odd.

(ii) Use the code produced per point (i) to approximate the functions $f(x) = e^{\cos^2(x)}$ in the interval $[0, \pi/4]$, $f(x) = \sin(x)$ in the interval $[0, 5\pi/2]$, and any other function which you think may provide an interesting test case.

5.- Fourier Continuation: FC(Gram) code and tests.

(i) Using Matlab open and run the provided file "advection_eqn_explicit.m". This will produce a numerical solution to the equation $u_t + u_x = 0$ with initial and boundary conditions such that the exact solution is $u(x,t) = \sin(\kappa(x - t))$. Then multiply (resp. divide) n and κ (resp. Δt) by the amount f (with e.g. $f = 2,\ 4,\ 8$). Note that the error is essentially unchanged in spite of the increasing size, in terms of wavelengths, of the domain of the PDE. Additionally, examine the error decreases that take place as the spatial and temporal discretizations are refined. Substituting the FC code by a finite-difference method repeat all of these experiments. Compare the resulting errors for large values of κ when using the same values of n and Δt in the FC and finite-difference codes.

(ii) Using the provided code *fcont_test.m* explore the convergence properties of the FC algorithm for the functions provided and/or other functions of your choice.