Solução aproximada de equações diferenciais II

Praciano-Pereira, Tarcisio [*] Neves, Antônio Jorge[†]

VIII Encontro de PósGraduação e Pesquisa - Outubro/2013

Resumo

Este artigo é a continuação dum anterior apresentado ao VII Encontro e também no DDays-2012, Cartagena, Colombia. Estamos construindo aqui uma solução aproximada duma equação diferencial. A metodologia será melhor discutida na primeira seção, entretanto, resumidamente ela consiste em construir uma solução aproximada num espaço de dimensão finita de splines polinomiais que é um espaço vetorial normado em que a norma é do tipo Sobolev, portanto um espaço vetorial finito de Sobolev. Estamos apresentando um exemplo uma vez que a equação diferencial que é o objeto deste artigo é fácil de ser resolvida exatamente e a vantagem desta apresentação é que ela representa um teste para o método.

palavras chave: base de splines, núcleos splines, operador diferencial

Classificação de assuntos da AMS-MSC: 47E05,41A05,42C40,46E35

This paper is a follow up of a previous one submitted to VII Encontro de Posgraduação e Pesquisa and presented in full at DDays-2012, Cartagena, Colombia. Here we are solving an approximate solution of an ordinary differential equation. The method will be better explained at the first section but we can shortly describe the work saying that we shall pick the approximate solution from a finite dimensional Sobolev space whose base is a family of highly differentiable splines. In fact this an example of the method as the equation involved is a simple one whose exact solution can be easily obtained by conventional method, the point is to present a test for the method.

keywords: differential operator, splines base, splines kernel.

AMS-MSC subject classification 47E05,41A05,42C40,46E35

[*] tarcisio@member.ams.org

[†] U Aveiro - Portugal jorgeneves@ua.pt

1 O problema e a metodologia

Suponha que tenhamos uma grande família de splines polinomiais a suporte compacto e de classe de diferenciabilidade muito alta, ver [2] onde se encontra demonstrado e mais do que isto, construída explicitamente uma tal família. Aqui não vamos usar nem as expressões dos elementos desta família e nem mesmo os detalhes de como eles foram construídos para assim produzir uma teoria livre dos detalhes da construção. Foi durante a redação deste artigo que chegamos à conclusão que esta abstração ajuda de forma geral até mesmo na construção de um dispositivo tão concreto como é o algoritmo computacional desenvolvido na última seção.

Por outro lado é possível que a leitora sinta que há uma excessiva abstração e para amenizar este aspecto sugerimos uma leitura preliminar de fonte onde a base de vetores pode ser encontrada, em [2] assim como nos itens mencionados na bibliografia do artigo citado.

Uma equação diferencial, ordinária, é uma expressão

$$F(t, x, y, \ldots, y^{(n)}) = 0; y = g(t, x); g \text{ diferenciável em } x; x \in [a, b]; \quad (1)$$

em que t representa o tempo e neste caso dizemos se tratar de um *sistema dinâmico*, ou quando for independente do tempo um *sistema estático*, entretanto neste artigo estaremos considerando um ponto fixo do tempo sendo possivel que posteriormente voltemos a estudar as consequências da dinâmica de um tal processo. É uma redução óbvia do problema.

Nós vamos supor que as condições do *teorema da função implícita* sejam preenchidas e que portanto em algum momento possamos escrever a equação (1.1) na forma

$$y^{(n)} = f(t, x, y, \ldots, y^{(n-1)};\ y = g(t, x);\ g \text{ diferenciável em } x;\ x \in [a, b]; \quad (2)$$

Esta suposição no fundo signfica que há uma adequada continuidade na dependência dos parâmetros deixando para um momento posterior o estudo de "pontos críticos".

Finalmente vamos descrever a metodologia e as ferramentas com que vamos trabalhar para resolver um problema como (1.2), deixando de lado os detalhes sobre as ferramentas como dissemos acima.

Seja

$$[a, b], I_i \in \Pi_n([a, b]);\ (\eta_i)_{i\in\{0,\ldots n-1\}};\ n \in \mathbf{Z}; \quad (3)$$

uma partição uniforme $\Pi_n([a, b])$ de um intervalo da reta $[a, b]$ da reta com n subintervalos, t um valor fixo do tempo, I_i o subintervalo de ordem i da partição e $supp(\eta_i) \subset I_i$ uma família de funções de classe $C^m([a, b])$ em que m é um inteiro arbitrário.

Nestas condições, e há uma construção desta ferramenta em [2], existe uma construção da família $(\eta_i)_{i\in\{0,\ldots n-1\}}$ temos um projetor de interpolação

$$\Pi_n([a, b]) \text{ a partição uniforme de } [a, b]; \quad (4)$$

$$S\,pl_n([a,b]) \text{ o espaço vetorial} \tag{5}$$
$$\eta_k; k = 0, \ldots, n \text{ os elementos da base} \tag{6}$$
$$\Phi \text{ um projetor sobre } S\,pl_n([a,b]) \tag{7}$$
$$\Phi(f) = \sum_{k=0}^{n} f(x_k)\eta_k \tag{8}$$

em que $f \in C^n([a,b])$ e os pontos $(x_k)_{k=0}^{n-1}$ são selecionados de tal forma que $\eta_k(x_k) = 1$. Observe que a base do espaço vetorial tem $n+1$ elementos uma vez os pontos de indexação são os $n+1$ nós da partição $\Pi_n([a,b])$. Isto faz com que

$$\Phi(f) = \sum_{k=0}^{n} f(x_k)\eta_k; \Phi : C^m \to S\,pl_n([a,b]); \tag{9}$$

seja um projetor de interpolação, ver [1], dum espaço de funções de classe C^k no espaço vetorial de dimensão $n+1$ de splines de classe C^m que é a classe de diferenciabilidade dos elementos da base de $S\,pl_n([a,b])$. Os pontos x_k são os *pontos de precisão* deste projetor.

Em tentativas anteriores tentamos expandir o espaço incluindo na base as derivadas dos núcleos e todas as verificações computacionais se mostraram ineficientes quando se nos veio a ideia de simplesmente *aproximar η_i^k, as derivadas, com o mesmo projetor e então calcular a norma usando estas aproximações*. O erro era evidente mas estamos procurando uma forma simples de simular operadores diferenciais e este não seria o caminho adequado.

Não há nenhuma dificuldade na obtenção dos pontos $x_k; \eta_k(x_k) = 1$, na contrução feita em [2], porque os núcleos são simétricos em relação aos nós da partição $\Pi_n([a,b])$, por contrução, uma vez que são potências de convolução duma função característica, então x_k são exatamente os nós da partição. No caso genérico, como temos uma família finita, com n elementos, basta varrer $[a,b]$ para encontrar os *pontos de precisão*, quando $\eta_k(x_k) = 1$, inclusive, computacionalmente, isto pode ser feito aproximadamente, para o caso que temos em mente que é a *fuga das partições uniformes*. Possivelmente iremos descobrir um algoritmo adequado para determinação dos *pontos de precisão* no caso geral em que a partição não seja uniforme.

Agora vem a definição da norma no espaço $S\,pl_n([a,b])$:

$$\|\phi\|_{S\,pl_n} = \sum_{j=0}^{m} \|\phi^{(j)}\|_\infty \tag{10}$$

em que m é a classe de diferenciabilidade dos elementos da família $(\eta_i)_{i=0\ldots n}$, n é o número de elementos de $\Pi_n([a,b])$, ou a dimensão do espaço de splines polinomiais $S\,pl_n$ o que no mesmo.

Estamos em condições de expor o método que nos vai levar a uma solução aproximada de uma equação diferencial, num caso particular de equação para a qual conseguimos desenvolver a metodologia, como já observamos no início, aplicar a metodologia num caso particular e inclusive bem conhecido, tem sentido que é de testar a metodologia. Faremos isto na próxima seção.

2 Uma solução aproximada

Considere a equação diferencial linear

$$y' = y; y = Kexp(t); \tag{11}$$

que é uma equação bem simples e bem conhecida a ponto de aparecer na disciplina introdutória *Cálculo Diferencial e Integral*, no primeiro ano dos curso universitários, apesar de que não apareça como *equação diferencial*. Este exemplo é muito bom pela taxa de crescimento da solução e consequemente é um bom teste da metodologia e pela absoluta simplicidade com que vamos poder apresentar o método.

Queremos resolver esta equação no intervalo

$$[0, 10]; exp(10) \approx 22026.465;$$

e vamos usar uma partição uniforme com $n = 10$ e seguindo os passo da metodologia desenvolvida em [2], vamos imergir o intervalo $[0, 10]$ num aberto, $[0, 10] \subset (-0.1, 10.1)$ e considerar um núcleo η de suporte medindo 0.2 centrado no ponto zero. As translações de η produzem uma partição da unidade e vamos considerar que $\eta \in C^m((-0.1, 10.1))$ com m suficientemente grande. Não importa neste momento qual é o valor de m uma vez que não temos nenhuma aplicação em vista que indicaria qual seria a necessidade de uma certa classe de diferenciabilidade. A leitora pode considerar qualquer seja $m > 0$. Temos assim uma partição da unidade de classe C^m subordinada a ao aberto $((-0.1, 10.1))$ que contém $[0, 10]$.

A equação $y' = y$ aplicada a esta partição da unidade nos dá a projeção, usando a equação (2.9),

$$\Phi(f)(x_k) = f'(x_k)\eta_k = f(x_k)\eta_k \tag{12}$$

aplicando a equação (2.9). É esta a função que vou integrar para obter o resultado aproximado.

A evidência computacional de que esta aproximação funciona foi obtida comparando a solução exata e a aproximada usando a norma de $||f|| = ||f||_\infty + ||f'||_\infty$ para calcular a distância entre a solução aproximada e solução exata. Esta é uma verificação computacional que pode ser encontrada em [3] e da qual vamos apresentar, na próxima seção, uma breve discussão.

3 Comprovação computacional

Para apresentar uma evidência razoável de que a metodologia funciona, construimos alguns programas em `python` que serão objetos de um trabalho em separado uma vez que neste momento ainda se encontram numa versão muito incipiente, mas já está funcional como bem o mostra [3], apenas a leitora precisa de alguma paciência e conhecimento de `python`.

Resumidamente, para não repetir o conteúdo de [3], construímos o projeto de interpolação ao qual aplicamos a derivada da exponencial, que é uma solução da equação diferencial que selecionamos para esta simulação. O resultado é uma aproximação polinomial da exponencial.

Outro módulo do programa calcula a primitiva de uma função qualquer que lhe seja passada como parâmetro, e `python` é uma linguagem funcional portanto uma função-`python` pode ter como parâmetro uma função-`python`, quer dizer, uma função que `python` reconheça como sintaticamente correta.

Aplicamos a esta primitiva assim calculada a norma

$$||f|| = ||f||_\infty + ||f'||_\infty \tag{13}$$

mostrando que há uma boa aproximação da primitiva assim obtida com $f(x) = exp(x)$.

Referências

[1] W. Delvos F-J, Schempp. *Boolean methods in interpolation and approximation.* Pitman research notes in Mathematics Series # 230 - 1989., 1989.

[2] Antônio Jorge Neves and Tarcisio Praciano-Pereira. Aproximate solution of differential equations i
splines partition of the unity. *DDays 2012 - Colombia - Cartagena - October 2012*, 2012:4, 2012.

[3] T. Praciano-Pereira. Computational evidence of an approximate splines solution of o.d.e. Technical report, Sobral Matemática - Preprints da SobralMatematica, 2013.01.