Construção elementar de splines polinômiais

Praciano-Pereira, T

Departamento de Matemática
Universidade Estadual Vale do Acaraú
13/09/2007
tarcisio@member.ams.org
pré-prints do Curso de Matemática de Sobral
no. 2007.5
Editor Tarcisio Praciano-Pereira
tarcisio@member.ams.org

Resumo

Este artigo é o primeiro de uma série de artigos em que estou construindo, de forma elementar, splines cúbicos.

O artigo começa com introdução sobre aproximação apresentando inclusive um pouco de *polinômios trigonométricos* onde há um fator de aceleração, `dilação` é o nome atual.

O ponto central é a construção de um núcleo, um 3-splines a suporte compacto, centrado na origem. A construção está inteiramente auto-contida e os pre-requesitos para sua leitura são um bom curso de Cálculo. O projeto é a construção de uma família formada de tais núcleos cobrindo um intervalo qualquer da reta tendo assim uma base para um espaço vetorial para aproximar funções definidas neste intervalo. Esta construção difere daquela feita em [4] porque não estou usando produto de convolução na construção do núcleo. **palavras-chave:** aproximação polinomial, splines cúbicos

1 Polinômios trigonométricos

Durante mais de um século dois *sistemas de aproximação* de funções reinaram sem discussão, os *polinômios de Taylor* e os *polinômios trigonômetricos*.

Vou discutir aqui os *polinômios trigonométricos* porque eles estão mais próximos dos meus objetivos, ou menos distantes.

A idéia dos *polinômios trigonométricos* vem direto da Álgebra Linear, que é a teoria básica que usamos também em *aproximação polinomial*. Em Álgebra Linear identificamos um conjunto *especial*[1] de vetores.

Todos os cálculos deste artigo se encontram feitos nos `script`s citados do `gnuplot` e o leitor tanto pode rodar o programa com o comando

```
gnuplot elementosXX.gnuplot
```

e assim ter uma visão gráfica[2] da construção, como pode ler o `script` e acompanhar os cálculos, isto permite que o artigo fique menor, e ao mesmo tempo o uso do `script` pode motivar a leitura do artigo.

Experimente o `script elementos01.gnuplot` para depois acompanhar a discussão que farei.

1.1 Produto escalar no espaço $\mathcal{C}([a,b])$

Em algum momento você verá que este não é o espaço correto, $\mathcal{C}([a,b])$, mas agora é difícil explicar por que. Depois que tudo ficar claro, você facilmente encontrará[3] meios para mudar de espaço. Considere $[a,b]=[-\pi,\pi]$, e observe que poderiamos trabalhar com um intervalo $[a,b]$ genérico, mas as fórmulas ficariam inutilmente mais complicadas. Depois que você compreender tudo, será fácil passar para as fórmulas gerais que são simples *dilatação* e *translaçao* das fórmulas aqui apresentadas.

Neste espaço a expressão

$$\int_{-\pi}^{\pi} f(x)g(x)dx \tag{1}$$

define um produto escalar, uma operação que tem as mesma propriedades do produto escalar da Geometria Analítica entre vetores do $\mathbf{R}^3$: *homogeneidade, permite que o escalar saia fora, se distribue com uma soma de funções.* Faça algumas experiências para ficar senhor da situação:

$$< af+bg, h >= (af+bg)\cdot h = \tag{2}$$

$$\int_{-\pi}^{\pi} (af(x)+bg(x))h(x)dx = \tag{3}$$

[1] Especial por alguma razão, até 1950 senos e cosenos porque são "ondas eletromagnéticas básicas, com quantidade de energia 1 e as comunicações se faziam eletromagneticamente.

[2] Chame o `gnuplot` com o comando acima, mantenha o cursor no terminal onde chamou `gnuplot`, siga dando "enter" neste teminal, para ter uma visão dinâmica da construção.

[3] Serve como sugestão para os professores, trabalhar no caso mais simples, depois construir a ponte para o caso genérico.

$$a \int_{-\pi}^{\pi} f(x)h(x)dx + b \int_{-\pi}^{\pi} g(x)h(x)dx = \quad (4)$$

$$a < f, h > +b < g, h >= a(f \cdot h) + b(g \cdot h) \quad (5)$$

Nas equações (2) - (5) estou usando os dois sistemas usuais de notação para o produto escalar, mas vou passar usar apenas o que se encontra à esquerda no resto do artigo.

Os vetores

$$x \mapsto sen(x), x \mapsto cos(x), x \mapsto sen(2x), x \mapsto cos(2x) \quad (6)$$

todos tem módulo 1 com este produto escalar definido na equação (1). São vetores *unitários e ortogonais*. Experimente esta afirmação.

Mais ainda, todos os vetores da forma

$$x \mapsto cos(nx); x \mapsto sen(nx); \ ; n \in \mathbf{N} \quad (7)$$

com exceção de $x \mapsto cos(0)$, são vetores *unitários e ortogonais*. Mas tem uma saída para esta questão que vou ignorar agora, você encontra a explicação em [5].

Então, a menos deste probleminha, temos uma família de *vetores ortonormais*, e com uma quantidade não finita, portanto estamos trabalhando com um espaço de dimensão infinita cujos elementos são, por exemplo, as comunicações que estão sendo enviadas a partir dos telefones celulares: ondas eletromagnéticas.

No `script` *elementos01.gnuplot* você vê uma *onda* "muito pouco eletromagnética" sendo decomposta, passo a passo[4], nas suas componentes ao longo dos vetores da família descrita na equação (7).

Os coeficientes foram calculados com o `script` em `calc` `elementos01.calc`. Estes dois `scripts` podem ser encontrados aqui [3].

1.2 Um exemplo de aproximação

Veja o seguinte exemplo bem simples de aproximação, com algum esforço mental e imaginação, você vai encontrar a semelhança entre ele e as figuras produzidas pelo script do `gnuplot`.

Considere um vetor do $\mathbf{R}^3$; $u = (3, 4, 5)$.

No $\mathbf{R}^3$ os físicos estabeleceram os vetores

$$\vec{i} = (1, 0, 0); \vec{j} = (0, 1, 0); \vec{k} = (0, 0, 1) \quad (8)$$

que têm as propriedades dos vetores *ortonormais* "eletromagnéticos"

$$sen(nx), cos(nx)$$

A figura plana, (1), página 3, mostra uma aproximação do vetor $u \approx 3i + 4j$

[4]Para ver o `script` funcionar passo-a-passo, deixe o cursor no terminal onde chamou `gnuplot` e vá dando *enter* neste terminal.

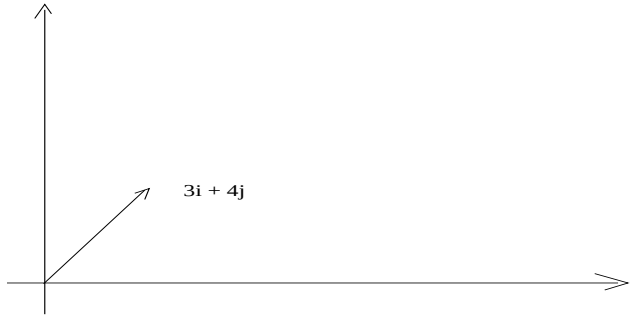

Figura 1: A aproximação plana de um vetor do espaço

em que perdemos a componente vertical. Se a parábola, no *script* do `gnuplot` representa a "realidade" que desejamos modelar (aproximar), então os polinômios trigonométricos são esta aproximação, da mesma forma como a figura plana representa uma aproximação do vetor u.

Uma observação, para uso posterior, subliminar, os vetores $\vec{i}, \vec{j}, \vec{k}$ são todos transformações de um único vetor, por exemplo do vetor $\vec{i}$, por rotações adequadas.

Quer dizer que descrevemos todo o universo tridimensional com transformações de apenas uma *informação básica*, $\vec{i}$.

2 A interpolação linear

Agora vou dar um salto significativo entre a primeira seção deste artigo, polinômios trigonométricos, e aproximação polinomial. Estamos vivendo a diferença entre século 19 e primeira metade do século 20, e os dias de hoje começando com os desenvolvimentos que ocorreram na década de 70 do século passado quando aos poucos ganharam clareza os *splines* que foram idealizados na década de 40 do século 20, mas foi preciso que surgissem os métodos computacionais adequados para que eles ganhassam vida.

Vou aqui apresentar a *interpolação linear* usando um método complicado. O objetivo da complicação é que ela representa um primeiro passo na construção dos *splines cúbicos* que farei na próxima seção.

Rode o `script` `elementos02.01.gnuplot` para acompanhar os comentários que farei sobre ele.

Se nos polinômios trigonométricos havia uma família aparentemente[5]múltipla, $sen(x), sen(2x), ...cos(x), cos(2x), ...$ aqui vamos produzir tudo a partir de um

[5]Aparentemente, porque $cos(x) = sen(x + \frac{\pi}{2})$ portanto existe um único elemento.

único *núcleo*, a função triângulo definida de uma forma esquisita, *na reta inteira é nula exceto no intervalo* $[-1,1]$ *em que ela é definida por dois segmentos de reta.*

Esta forma esquisita é que nos permite somar *translações e dilações* deste *núcleo* para compormos, praticamente, qualquer figura desejada, aproximadamente.

Observe a figura (2) página 4,

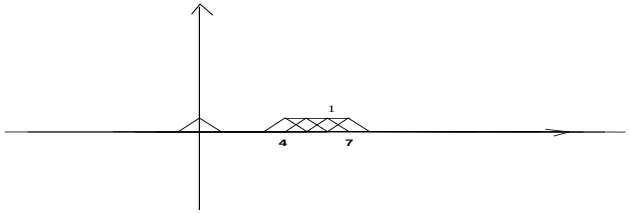

Figura 2: Soma de dois núcleos-translações do núcleo básico

A soma das três transladadas é constante 1 no interior do intervalo $[4,7]$ e podemos assim observar uma *aproximação da unidade* com a soma destas translatadas do núcleo. Uma aproximação que é válida no intervalo $[4,7]$, mas com um maior número de translatadas podemos aumentar a validade desta aproximação.

O segredo se encontra em que a soma de duas funções lineares é uma função linear (do primeiro grau) e em particular a soma de $(x-a)+1$ com $1-(x-a-1)$ é igual a um no intervalo $[a,a+1]$. Precisamos de uma notação compacta para translatadas, ou mais exatamente, precisamos de chamar a translatada de um *funcional*, uma operação executada sobre funções, aqui sobre um núcleo básico.

Definição 1 *A operação translação*

Dado uma função f *definida na reta, e um número real* $\underline{a}$ *definimos*

$$f_a(x) = f(x-a)$$

a translação de f *para o ponto* $\underline{a}$*. Observe o sinal.*

Com esta notação a soma de translatadas que podemos ver na figura (2) é

$$n_4 + n_5 + n_6 + n_7 \tag{9}$$

translatadas do núcleo básico $n_0 = n$.

Você pode repetir esta experiência rodando `elementos02.02.gnuplot`.

Se usarmos coeficientes diferentes da unidade podemos obter figuras esquisitas como o `script` `elementos02.01.gnuplot` lhe mostra e que você pode alterar para obter outras figuras e assim experimentar o poder desta ferramenta.

Uma outra operação pode alterar a precisão da análise dos núcleos que é a *dilação*, que aparece no *núcleo fundamental eletromagnético*[6] em

$$sen(2x), sen(3x), cos(2x), ...$$

Quando multiplicarmos o parâmetro por um número maior do 1, que concentramos a informação do suporte em um intervalo menor.

Você pode alterar o `elementos02.01.gnuplot` para fazer o gráfico do núcleo *dilatado*, experimente. O nome desta operação ficou *dilação*:

Definição 2 *A dilação*

Se f *for uma função definida na reta, definimos*

$$_af(x) = af(ax)$$

a dilatada de f *por* $\underline{a}$*.*

Algumas vezes um coeficente $\frac{1}{a}$ "elimina" por conveniência o fator que aparece na definição, mas na prática isto significa que este quociente faz parte do coeficiente que estivermos usando com a dilação de uma núcleo.

Então com estas duas operações, *dilação* e *translação* podemos analisar, ou codificar, com grande precisão os dados discretos que colhermos de experimentos como mostra `elementos02.01.gnuplot`.

Mas esta é a interpolação linear.

A interpolação linear tem um defeito, não leva em consideração as taxas de variação, a modelagem é rugosa. Vamos discutir na próxima seção uma saída para obter modelos macios.

Algumas definições, antes de prosseguir, e usando o material que você já viu, experimentou com ele, já lhe é familiar.

Definição 3 (Suporte) *Suporte*

Suporte de f *é o fecho do conjunto*

$$\{x\ ;\ f(x) \neq 0\}$$

é o conjunto dos pontos onde f *é essencialmente não nula, onde de fato interessa calcular a integral.*

Definição 4 (Ponto de precisão) *Ponto de precisão*

Quando uma função, f*, modelo, interpola os dados* (a_k, b_k) *, se deve ter* $f(a_k) = b_k)$ *, siginificando com isto que o gráfico de* f *passa nos pontos* (a_k, b_k) *, entendidos como os dados coletados de um experimento. Estes são os pontos de precisão, quando o "modelo" coïncide com a "realidade".*

Já falamos de *splines*, sem definí-los,

[6] Esta denominação, núcleo, está errada, as senoides não são núcleos no sentido como o entendemos hoje, quando você entender este erro, estará dominando o assunto.

Definição 5 (Splines) *Splines*

Um splines *é uma função polinomial por pedaços, de grau n e com classe de diferenciabilidade $n-1$. Ao dizer que que é de grau n se quer dizer que os pedaços de polinômios que compõem o splines são de grau menor ou igual a n.*

Os splines aqui construídos são derivados de um *núcleo* básico que é uma função à suporte compacto, quer dizer que fora de um intervalo, o suporte, ela está definida por um segmento de reta, logo um polinômio de grau 0. Os outros pedaços podem ser de grau maior do que zero, mas haverá um pedaço que será de grau n, o que caracteriza o grau do splines.

As poligonais construídas pelo `script elementos02.01.gnuplot` são 1-splines, quer dizer, splines de grau 1, cuja classe de continuidade é 0 = 1-1, são apenas contínuas, suas derivadas são descontínuas.

No caso de nós distribuido de forma não uniforme consigo a interpolação linear com triângulos isósceles determinados pelo ponto (a_k, b_k) e pelos extremos do intervalo que contém este ponto como ponto de precisão.

Finalmente a derinição de núcleo, mostrando que chamamos impropriamente $sen(x)$ de núcleo

Definição 6 (Núcleo) *Núcleo*

Chamos de núcleo a uma função positiva cuja integral seja 1.

Neste artigo os nossos núcleos serão também à suporte compacto, o que não faz parte da definição. Um exemplo de núcleo é

$$f(x) = \frac{1}{\sqrt{\pi}} e^{-x^2} \tag{10}$$

cujo suporte é a reta inteira. Esta é a *gaussiana* normalizada, multiplicada por um coeficiente que faz com que sua integral seja 1, usada pelos probabilistas para definir uma tipo de probabilidade que eles acreditam que descreve boa parte dos fenômenos do Universo.

3 Núcleo 3-splines

Nesta seção vou construir detalhadamente um 3-splines cuja integral é 1, um *núcleo-3-splines a suporte compacto*.

A metodologia consiste em partir da função segunda derivada que vou chamar de f para manter compatibilidade com o `script` do `gnuplot` que pode ser encontrado em [3, elementos03.gnuplot].

Como quero obter uma função de classe $\mathcal{C}^2$ vou começar construindo uma segunda derivada contínua, f vou obter o núcleo h calculando sucessivamente as primitivas de f e de g. Resumidamente:

$$y = f(x) y = g(x) = \int_{-\infty}^{x} f(t)dt \; ; \; y = h(x) = \int_{-\infty}^{x} g(t)dt \tag{11}$$

sendo h o núcleo que será construído.

A figura (3) página 7,

enquanto que a figura (3) é um produção artística, a figura (4) página 8,

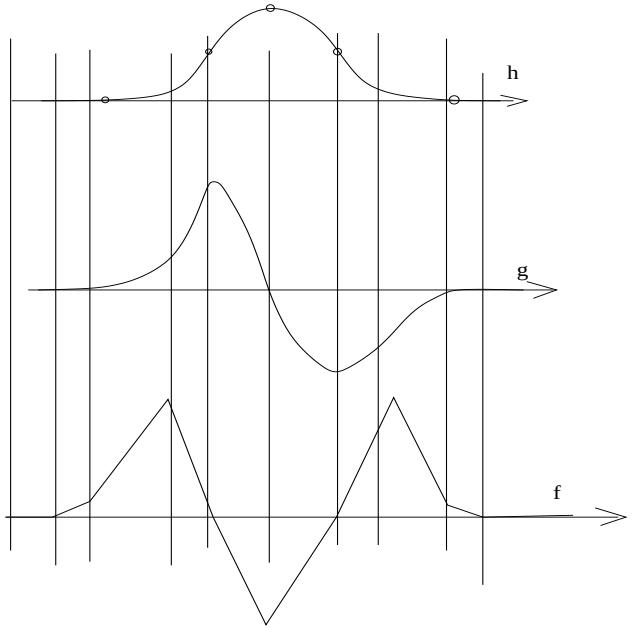

Figura 3: Sucessão de primitivas - um núcleo 3-splines

mostra o resultado da construção feita com `gnuplot` que se encontra no `script elementos03.gnuplot`.

Na próxima seção vou descrever cuidadosamente o processo de construção e ao final vou dar uma sugestão de como fazer uso deste núcleo em aplicaçoes, um trabalho mais detalhado será objeto de um outro artigo.

3.1 Sucessão de primitivas

A segunda derivada é um 1-splines, uma função contínua, linear por pedaços, construída de tal modo que sua integral seja zero. Como é linear por pedaços é

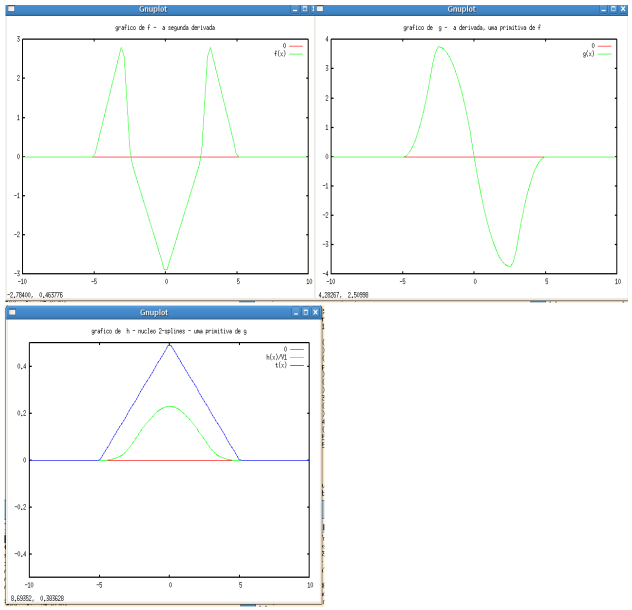

Figura 4: Construção de um núcleo 3-splines a suporte compacto

relativamente simples obter a condição de que a integral seja zero, basta fazer com que seu gráfico seja constituído de triângulos cuja somas das áreas seja zero. Isto pode ser visto na parte inicial do programa `elementos03.gnuplot`, a seleção dos nós

$$a_0 = -5, a_1, a_2, a_3, a_4, a_5, a_6 = 5 \tag{12}$$

e dos valores atribuidos nestes pontos para a soma das áreas dos triângulos seja zero.

Defini f como uma função a suporte compacto que se anula fora do intervalo $[a_0, a_6]$ e no interior do intervalo formado pelos segmentos de reta que unem os pontos

$$(a_0, b_0), (a_1, b_1), (a_2, b_2), (a_3, b_3), (a_4, b_4), (a_5, b_5), (a_6, b_6) \tag{13}$$

ver [3, elementos03.gnuplot].

Como f é uma *polinomial* do primeiro grau, qualquer primitiva será uma *polinomial* do segundo grau e agora de classe $\mathcal{C}^2$. Defini

$$g(x) = \int_{-\infty}^{x} f(t)dt \tag{14}$$

o uso do limite $-\infty$ é prático, substitui a cascatas de equações que somos obrigados a escrever num algoritmo computacional como podem ser vistas na derinição de g. Como f é a suporte compacto, qualquer valor de x anterior ao primeiro nó a_0 faz com que esta integral seja nula. Como a integral de f é zero, qualquer valor de x posterior ao último nó a_6 resulta em zero nesta integral.

Ao definir agora

$$h(x) = \int_{-\infty}^{x} g(t)dt \tag{15}$$

me beneficio dos mesmos argumentos já expostos acima para ter uma função polinomial por pedaços do terceiro grau, de classe $\mathcal{C}^2$ que se anula fora do intervalo $[a_0, a_6]$, um 2-spline a suporte compacto.

Referências

[1] `gnuplot` *um programa para fazer grafico e alguns cálculos*
http://www.gnuplot.info

[2] *A enciclopédia livre na Internet*
http://www.widipedia.org

[3] Praciano-Pereira, T *Programas para Cálculo Numérico - programas.tgz*
http://www.4shared.com/dir/2041165/e14cc331/programas.html

[4] Praciano-Pereira, T *Splines por convolução*
http://www.4shared.com/file/17757661/3515d0b6/convspl02.html

[5] Praciano-Pereira, T
Cálculo Avançado
http://www.4shared.com/file/18104572/4d05bd7e/avancado.html