Capítulo 18
Interpolação

Resumo do capítulo

- Interpolação é uma técnica para estimar valores de funções em pontos intermediários de intervalos, a partir de valores da função calculados nos extremos desses intervalos.
- Há várias técnicas para esse fim, algumas já incorporadas em funções do **MATLAB**. Outras podem ser programadas no **MATLAB**.
- Veremos aqui funções do **MATLAB** para interpolação de funções de uma ou duas variáveis com técnicas diferentes, usando
  - funções polinomiais lineares;
  - splines cúbicas;
  - triangulações.

Help

- Veja Help –> **MATLAB** Help –> **MATLAB** –> Mathematics –> Polynomials and Interpolation –> Interpolation
- Arquivos .m no repositório do curso, especificamente de *mm1801.m* a *mm1810.m*.

Funções unidimensionais

- O método *default* para desenho de gráficos de funções no **MATLAB** usa interpolação linear, traçando segmentos de reta entre dois pontos com abcissas consecutivas.
- Veja o arquivo *mm1801.m*, onde são criados vetores $x1, x2$ de abscissas no intervalo $[0, 2\pi]$, com 60 pontos ($x1$) e 6 pontos ($x2$) uniformemente distribuídos. Note a diferença de qualidade dos dois gráficos.

Outro exemplo

Vamos agora usar funções de interpolação do **MATLAB**.

- Veja o arquivo *mm1802.m* onde são expressos o limiares da audição humana em várias freqüências do som. O ouvido humano é mais sensível nas freqüências ao redor dos 3kHz, conforme se pode deduzir do gráfico gerado.
- Vamos usar a função *interp1* (interpolação de função unidimensional) para estimar o valor da função ao redor de 3 kHz.
- A forma geral da função *interp1* é

  interp1(x,y,xi,método)

  onde: x, y são os pontos e valores correspondentes da função que se quer interpolar; xi é o ponto do qual se deseja saber o valor interpolado; e *método* pode ser um de 'linear', 'cubic', 'spline', 'nearest'.

Em vários pontos

Podemos interpolar em mais que um ponto:

- Veja o arquivo *mm1803.m*, onde examinamos de perto a região da vizinhança do mínimo da função interpoladora.
- Vamos calcular o mínimo da função interpoladora.

```
%
>> [spl_min,i]=min(spli) %min e seu indice
spl_min =
   -8.4245
i =
    45
>> Hz_min=Hzi(i) % freqüência no mínimo
Hz_min =
   3.3333e+03
```

Em vários conjuntos de pontos

- É possível interpolar de uma só vez várias funções. Basta que o segundo argumento y, na invocação

  interp1(x,y,xi,método)

  seja uma matriz cujas colunas descrevam o valor das várias funções calculadas no mesmo conjunto de pontos (dado pelo primeiro argumento x.

É importante notar que a função *interp1* exige que a variável independente seja monotônica.

Interpolação em duas variáveis

- Veja o arquivo *mm1804.m* que
  - cria vetores x, y, z, onde z é a profundidade do oceano nas coordenadas x, y (todas as medidas para efeito ilustrativo somente); e
  - exibe um gráfico tridimensional do fundo do oceano na região dada por x, y, resultado da aplicação da função *mesh(x,y,z)*. Esse gráfico é obtido por interpolação linear
- Aplicações simples da função

  interp1(x,y,z,xi,yi,método)

  resultam no valor interpolado no ponto (xi, yi).

Melhorando a resolução

- Assim como foi feito em uma dimensão, podemos aumentar a resolução e interpolar com uma função não linear para obter um gráfico muito mais fiel à realidade. O arquivo *mm1805.m* mostra todos os passos necessários
- Podemos agora encontrar o pico do fundo do mar

```
>> zmax=max(max(zzi))
zmax =
  108.0520
>> [i,j]=find(zmax==zzi);
>> xmax=xi(j)
xmax =
    2.6207
>> ymax=yi(i)
ymax =
    2.9231
```

Com dados espalhados

- Nem sempre é possível ajustar pontos em um reticulado. Considere o exemplo no arquivo *mm1807.m* que gera e exibe uma triangulação de Delaunay de pontos aleatórios.
- Uma vez calculada a triangulação, é possível interpolar pontos contidos nos triângulos usando as funções
  - *tsearch(x,y,tri,xi,yi)*, que encontra a linha da triangulação *tri* que corresponde aos vértices do triângulo que contém o ponto (xi, yi). Essa função aceita mais que um ponto em xi, yi.
  - *dsearch(x,y,tri,xi,yi)*, que encontra o índice em x, y que corresponde ao ponto de *tri* mais próximo de xi, yi.

Funções correlatas

- É possivel encontrar o casco convexo (*convex hull*) de um conjunto de pontos usando a função *convhull(x,y)*. Veja o arquivo *mm1808.m* que constrói e exibe um casco convexo para um conjunto de pontos.
- Outra função similar é *voronoi(x,y,tri)*, que constrói o diagrama de Voronoi dos pontos x, y a partir da triangulação dada por *tri*. Veja o arquivo *mm1809.m*.
- A partir de uma triangulação de Delaunay é possível interpolar pontos de um reticulado sobreposto sobre ela usando a função *griddata*. Veja o arquivo *mm1809.m*.