Instituto Tecnológico de Aeronáutica – ITA
Matemática Computacional – CCI-22
Laboratório 5 – Interpolação

Professor: Marcos Ricardo Omena de Albuquerque Maximo

10 de abril de 2019

1 Tarefas

1.1 Implementação

Implementar as seguintes funções em MATLAB (cada uma em um arquivo .m separado):

1. `p = PolinomioInterpolador(x, y)`: determina o polinômio interpolador $p_n(x)$ a partir de $n+1$ pontos de interpolação, definidos pelos vetores coluna $x = [x_0, x_1, ..., x_n]^T$ e $y = [f(x_0), f(x_1), ..., f(x_n)]^T$. O vetor linha `p` é escrito na seguinte forma:
$$p = \begin{bmatrix} a_n & a_{n-1} & a_{n-2} & \cdots & a_0 \end{bmatrix} \tag{1}$$
De modo a representar um polinômio interpolador $p_n(x)$ na forma:
$$p_n(x) = a_n x^n + a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_0 \tag{2}$$
Note que este formato de `p` é coerente com o esperado pela função `polyval`.

2. `T = TabelaDiferencasDivididas(x, y)`: calcula a Tabela de Diferenças Divididas T a partir de $n+1$ pontos de interpolação, definidos pelos vetores coluna $x = [x_0, x_1, ..., x_n]^T$ e $y = [f(x_0), f(x_1), ..., f(x_n)]^T$. Adota-se o seguinte formato para T:
$$T = \begin{bmatrix} f[x_0] & f[x_0, x_1] & f[x_0, x_1, x_2] & \cdots & f[x_0, x_1, ..., x_n] \\ f[x_1] & f[x_1, x_2] & f[x_1, x_2, x_3] & \cdots & 0 \\ f[x_2] & f[x_2, x_3] & f[x_2, x_3, x_4] & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f[x_n] & 0 & 0 & \cdots & 0 \end{bmatrix} \tag{3}$$
Note ainda que T tem dimensão $(n+1) \times (n+1)$.

3. `yq = InterpolacaoFormaNewton(dd, x, xq)`: calcula os valores interpolados $y_q = p_n(x_q)$ utilizando polinômio interpolador $p_n(x)$ na Forma de Newton. O vetor `dd` $= [f[x_0], f[x_0, x_1], ..., f[x_0, x_1, ..., x_n]]$ contém os operadores diferenças divididas necessários para o cálculo e $x = [x_0, x_1, ..., x_n]^T$ são os $n+1$ pontos de interpolação. No caso de x_q ser um vetor, y_q é um vetor tal que $y_q(i) = p_n(x_q(i))$. Neste caso, x_q e y_q são vetores coluna.

1.2 Análise

1. Considere os seguintes pontos amostrados de uma certa função $f(x)$:

x	-1	0	1	2	3
$f(x)$	1	1	0	-1	-2

Tabela 1: Pontos amostrados de uma certa função $f(x)$.

Obtenha polinômios interpoladores a partir dos dois métodos possíveis a partir das funções implementadas (resolução de sistema linear e Forma de Newton) e mostre que eles coincidem. Para isto, é suficiente mostrar que os gráficos dos polinômios interpoladores obtidos a partir de ambos os métodos coincidem.

2. Seja a função de Runge:

$$f(x) = \frac{1}{1+25x^2} \tag{4}$$

Considere ainda $n+1$ pontos equidistantes no intervalo $[-1,1]$. Pode-se obter o polinômio interpolador $p_n(x)$ a partir destes $n+1$ pontos de interpolação. Mostre $p_n(x)$ para $n \in \{1,2,4,8,16\}$ juntamente com $f(x)$ num único gráfico. Utilize a função `InterpolacaoFormaNewton` para realizar a interpolação. Observe as curvas obtidas e discuta os resultados com base no Fenômeno de Runge. Aumentar n sempre melhora a qualidade da interpolação?

Então, considere $E(n) = \max_{x\in[-1,1]} |f(x) - p_n(x)|$, a função que mede o máximo desvio de $p_n(x)$ em relação a $f(x)$ no intervalo $[-1,1]$. Plote um gráfico de como varia $E(n)$ com n para $n \in \{1,2,...,20\}$ e tente determinar o n que minimiza $E(n)$. Para determinar $\max_{x\in[-1,1]} |f(x) - p_n(x)|$, use como aproximação $\max_{x\in\{-1:h:1\}} |f(x) - p_n(x)|$ com h pequeno. Considerando $E(n)$ como medida de custo da interpolação, este $p_n(x)$ seria o melhor polinômio interpolador. Comente o resultado obtido.

Note que o script `FenomenoRunge` plota os gráficos pedidos neste item, desde que as funções `TabelaDiferencasDivididas` e `InterpolacaoFormaNewton` tenham sido implementadas.

2 Instruções

- A primeira etapa do processo de correção consistirá em submeter as funções implementadas a vários casos de teste de forma automatizada. Assim, os cabeçalhos das funções devem ser seguidos **rigorosamente**. Arquivos .m com os nomes destas funções e os cabeçalhos já implementados foram fornecidos juntamente com este roteiro. Dê preferência a implementar seu laboratório a partir destes arquivos .m fornecidos para evitar erros.

- **Não** é permitido o uso de funções ou comandos prontos do MATLAB que realizem toda a funcionalidade atribuída a uma certa função. Entretanto, o uso destas funções para verificação das implementações realizadas é encorajado. Em caso de dúvida quanto à permissão de uso de alguma função ou comando, recomenda-se consultar o professor.

- Não é necessário reimplementar métodos de laboratórios anteriores. Assim, funções implementadas em laboratórios anteriores podem ser utilizadas. Caso prefira, também é permitido utilizar funções equivalentes do MATLAB. Por exemplo, na implementação da função `PolinomioInterpolador`, é necessário resolver um sistema linear de equações, de modo que é permitido usar diretamente a função `linsolve` do MATLAB.

- Não é necessário se preocupar com verificação dos dados de entrada: assuma que x e y tem mesma dimensão, que `dd` esteja conforme formato especificado no cabeçalho da função etc.

- Os arquivos .m implementados devem ser entregues juntamente com um relatório.

- No relatório, não é necessário demonstrar que as funções implementadas funcionam corretamente (isto será verificado separadamente). Basta incluir resultados e conclusões relativos à **Análise**.

3 Dicas:

- Após ter obtido o polinômio interpolador com `PolinomioInterpolador`, utilize `polyval` para calcular os valores de $p_n(x)$ desejados.

- Para o método de determinação de polinômio interpolador através da solução de sistema linear de equações, a convenção utilizada nos slides é considerar o vetor coluna:
$$a = \begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_n \end{bmatrix}^T \tag{5}$$
Para representar o polinômio:
$$p_n(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n \tag{6}$$
Note que esta convenção é diferente da utilizada na função `PolinomioInterpolador`. A conversão entre as duas convenções pode ser obtida através do comando `p = fliplr(a')`.

- Para a implementação de algumas das funções, pode ser conveniente utilizar operações elemento a elemento de vetores: `a .* b`, `1 ./ a`, `a.^2` etc.

- Perceba que o vetor `dd` passado como parâmetro para `InterpolacaoFormaNewton` é a primeira linha da tabela `T` retornada por `TabelaDiferencasDivididas`, i.e. `dd = T(1,:)`.