SME0300 - Cálculo Numérico

Exercício de Aplicação 1

Professor: Eduardo F. Costa
Autor: João P.C. Bertoldo
Atualização: Ricardo A. Cardona

August 27, 2018

1 Introdução

O objetivo principal é que você se familiarize com o MATLAB e aprenda o básico. O tema de nossos "experimentos" no MATLAB vai ser conversão de um número em $\Re$ para uma representação em base $F(\beta, t, m, M)$.

2 Sistemas de números discretos

2.1 Contextualização

Quando fazemos cálculos 'na mão' (na verdade, no cérebro...), consideramos que o conjunto dos $\Re$ é contínuo, ou seja, não há 'saltos' entre os números. Porém, em uma máquina - 'máquina' pode se referir a uma calculadora, um computador, um Arduíno, etc -, a história é outra... Quando realizamos uma conta em um computador, é preciso lidar com descontinuidades entre os números. As máquinas têm precisão limitada, ou seja, elas têm uma quantidade FINITA de dígitos. Então, vamos tentar entender um pouco melhor o que acontece com os números reais, que são contínuos, quando são representados em um sistema discreto, que é descontínuo. Há duas representações possíveis: ponto fixo e ponto flutuante - lembra-se que em C havia o "float"? Pois é, o nome vem daí. A representação mais utilizada é a de ponto flutuante (são os floats e doubles de seus programas C, C++, Java, etc). Por isso, vamos lidar com eles. Quando lidamos com números em ponto flutuante:

1. Há uma quantidade predeterminada de dígitos;
2. Os números são representados como $0.d_1d_2d_3...$, onde d significa "dígito" - isto é a **mantissa**;
3. A mantissa é acompanhada de um outro número: que é o **expoente** da base (β);

4. A mantissa é sempre menor que um e maior que β^{-1}. Ex: base 10, mantissa maior que 0.1. Pense bem: se pudesse ser menor, na verdade o expoente cairia e não haveria a necessidade.

Exemplo: se estamos em uma base 10, com 4 dígitos e queremos o número 764.3, ele é representado por $0.7643 * 10^3$. Mas, se tivéssemos só 2 dígitos, seria $0.76 * 10^3$, cujo valor real é 760. Ou seja, perdemos precisão. Perdemos informação sobre o número, porque estamos lidando com informação limitada. Ainda há um porém: o próprio expoente também é limitado...

2.2 Ponto fixo vs ponto flutuante

Ponto fixo e ponto flutuante são duas formas de se representar os números digitalmente como mencionado anteriormente. Para entender melhor o ponto fixo imagine um sistema decimal de 10 digitos, 5 para a parte inteira e 5 para a parte fracional. Teremos IIIII.FFFFF representado. Dessa maneira o menor absoluto possível e o maior são respectivamente os números 00000.00001 e 99999.99999.

Como foi mostrado no ponto flutuante é utilizada uma **mantissa** ou **significando** em conjunto com um expoente. Imagine agora um sistema com 8 digitos para representar a **mantissa** e 2 para representar o exponente, totalizando 10 digitos como no exemplo anterior. O menor absoluto e o maior são respectivamente os números $0.100000000 * 10^{-49}$ e $0.99999999 * 10^{50}$ (note que $49 + 50 = 99$). Dessa maneira o ponto flutuante se mostra muito interessante para se representar números muito grandes e muito pequenos no computador portanto estaremos estudando essa forma de representação.

3 Sistema $F(\beta,t,m,M)$

3.1 Representação

Para mostrar de forma compacta estes limites, escrevemos nosso sistema de numeração na base $F(\beta, t, m, M)$, onde:

- β é a base
- t é o número de dígitos
- m é o menor expoente possível
- M é o MAIOR expoente possível

Dessa maneira, um número é representado da seguinte maneira:

$$x = \pm 0.d_1 d_2 d_3 ... \times \beta^e$$

Respeitando as seguintes condições:

1. $d_1 \neq 0$

2. $0 \leq d_i < \beta$

3. $\beta^{-1} \leq 0.d_1d_2d_3... < 1$

4. $-m \leq e \leq M$

Exemplo sem arredondamento:

- 10.57848 em $F(10,3,2,5)$ é $0.105 * 10^2$;
- 0.0154121 em $F(10,4,2,5)$ é $0.1541 * 10^{-1}$;

3.2 Arredondamento

Para se representar os números é necessário realizar o arredondamento e o truncamento. O truncamento é simples, basta considerar os dígitos até o valor especificado de t, para o arredendamento uma estratégia pode ser utilizada, isto é, a soma da metade do menor valor que o sistema pode representar na **mantissa** com $(\beta^{-t}/2)$ antes do truncamento. Observe os exemplos:

- 10.57848 em $F(10,3,2,5)$: $0.1057848 + 0.0005 = 0.1062848$ truncando fica $0.106 * 10^2$;
- 0.0154121 em $F(10,4,2,5)$: $0.154121 + 0.00005 = 0.154171$ truncando fica $0.1541 * 10^{-1}$;

Para o arredondamento vamos considerar s como sendo a **mantissa**, x sendo o valor de entrada e $\bar{x}$ sendo o valor arredondado. Dessa forma:

$$\mid x \mid = s \times \beta^e$$

Tal que:

$$\beta^{-1}\Big(1 - \frac{\beta^{-t}}{2}\Big) \leq s < \Big(1 - \frac{\beta^{-t}}{2}\Big)$$

Assim calcula-se:

$$s + \frac{\beta^{-t}}{2} = 0.d_1d_2d_3...$$

Com o truncamento:

$$\bar{x} = (sinalx)0.d_1d_2d_3..d_t \times \beta^e$$

Por exemplo, para $F(10,3,2,5)$ a faixa de s é $0.09995 \leq s < 0.9995$. O limite superior faz muito sentido porque uma variação da metade do menor valor da mantissa $(0.001/2)$. Mas e quanto o limite superior? Considere o valor de mantissa 0.9996. Está fora do limite, mas é um valor possivel de se representar, assim ele é colocado em novo formato de mantissa 0.09996. Agora a mantissa se encontra dentro da faixa, daí vem o motivo do limite inferior.

Um detalhe importante de resaltar é o de se realizar uma verificação de *overflow*, *underflow* e nulo antes da conversão. Isso é realizado mediante a comparação o menor e maior valor que o sistema é capaz de respresentar e o valor nulo propriamente dito.

Bem, isto é um resumo. Para maiores informações, consulte o livro da Neide. Em especial, vamos nos basear em informações das seções: 2.2 e 2.3.

4 MATLAB

Vai funcionar assim: primeiro você vai ver um script de MATLAB lendo todos os comentários e comandos - na verdade há poucos comandos, dê atenção aos comentários e as instruções; depois de lê-lo, volte a este PDF para saber o que fazer - eventualmente você deverá fazer alguns exercícios sobre a matéria; em seguida vamos fazer uma função em MATLAB que converte um número real para uma representação em uma base $F(\beta, t, m, M)$ qualquer. A implementação será feita por partes, então você verá que há vários arquivos com o mesmo nome e com um v_1, v_2, etc no final - "v" de "versão". A cada versão, vamos evoluir um pouco para tentar usar as funcionalidades do MATLAB. **MUITO IMPORTANTE**: na realidade, o MATLAB lida com representações numéricas de reais (com floats, doubles, etc). Porém, vamos converter os números para bases com precisão muito menor, por isso vamos considerar que, para a nossa função, os inputs têm precisão infinita (como se fossem números reais mesmo).

5 Partiu!

1. Resolva os exercícios

2. Abra o MATLAB.

3. No navegador de diretório (botão logo à esquerda da barra de endereço), navegue até a pasta onde estão os arquivos e a selecione.

4. Na barra à esquerda, onde você deve ter vários arquivos aparecendo, abra o arquivo "HelloMatlab.m" e siga as instruções dos comentários.

5. Abra o arquivo "paraFv1.m" e siga o fluxo. Tente entender o código todo, é a 1^a versão da nossa função que converte um $\Re$ em uma representação em base $F(\beta, t, m, M)$. Vamos assumir que $\beta = 10$; $m = 3$; $M = 3$. O valor de t será passado à função.

6. Agora, vamos testar esta função usando o valor de $t = 3$ nas chamadas da função. Abra o arquivo "TesteParaFv1.m", leia-o e execute-o.

7. Muito bem, vamos fazer melhor. Abra e leia a versão 2 (arquivo "paraFv2.m").

8. Agora, rode o script de teste versão 2.

9. Na versão 3, vamos generalizar os termos m e M. Então, se você não entender o código, faça os exercícios e tente perceber o padrão com os exemplos numéricos para chegar à generalização. Faça os Exercícios 6 até 8 se tiver problemas com a parte de underflow. Faça os Exercícios 9 até 12 se tiver problemas com OVERFLOW.

10. Corra para a versão 3, já vai ficar bem mais interessante... E rode os testes também!

11. Certo, é hora de generalizar β no código. Outra coisa que vamos mudar: vamos fazer a função não precisar receber a base $F(\beta, t, m, M)$, porque a chamada da função fica carrega - toda vez temos que passar a mesma coisa. Porém, também não queremos 'hardcode', queremos manter o código genérico. Como fazer isso? Vá ler a versão 4 da função e dos testes...

12. Para acabar, veja a versão 5 e rode os testes.

13. FIM.

Agora você já viu como se escrevem alguns dos principais comandos em MATLAB.

6 Exercícios

1. Represente o número $(1101)_2$ na base 10.

2. Represente o número $(0.101)_2$ na base 10.

3. Represente o número $(13)_{10}$ na base 2.

4. Represente o número $(3.8)_{10}$ na base 2.

5. Represente os números no sistema $F(10, 3, 5, 5)$

   5.1. 1234.56
   5.2. -0.000549624
   5.3. 0.9995
   5.4. 123456.7
   5.5. -0.0000001

6. Encontre o menor valor em absoluto possível para o sistema $F(10, 1, 3, 3)$.

7. Encontre o menor valor em absoluto possível para o sistema $F(10, 2, 3, 3)$.

8. Encontre o menor valor em absoluto possível para o sistema $F(10, 2, 3, 3)$.

9. Encontre o menor valor em absoluto possível para o sistema $F(10, 3, 3, 3)$.

10. Percebeu o padrão? Então faça para $F(10, t, 3, 3)$, com t genérico.

11. Agora encontre o valor máximo do sistema $F(10, t, 3, 3)$. Sugestão: faça dos outros para perceber o padrão antes de fazer o genérico.
12. Encontre o menor em absoluto possível para o sistema $F(10, 3, 3, 3)$.
13. Encontre o menor valor em absoluto possível para o sistema $F(10, 3, 2, 3)$.
14. Encontre o menor valor em absoluto possível para o sistema $F(10, 3, 1, 3)$.
15. Encontre o MAIOR valor possível para o sistema $F(10, 3, 3, 3)$.
16. Encontre o MAIOR valor possível para o sistema $F(10, 3, 3, 2)$.
17. Encontre o MAIOR valor possível para o sistema $F(10, 3, 3, 1)$.