Exemplos simples de simulação estocástica em computador

8ª série de exercícios

AMOSTRAGEM E DESVIOS ESTATÍTICOS

Uma **amostra** é um conjunto de N resultados numéricos de um processo aleatório, que chamaremos de x_i, i = 1,2,...N

A **média amostral** desses resultados é $\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$

O valor de $\overline{x}$ varia de amostra para amostra, e é uma estimativa do valor médio verdadeiro (que é obtido no limite $N \to \infty$).

O **desvio padrão da média** é um número que mede a dispersão das médias calculadas usando amostras de tamanho N.

Para N >> 1, o **desvio padrão dos valores** de uma amostra é $\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2} = \sqrt{\overline{x^2} - \overline{x}^2}$.

Pode-se mostrar que o desvio padrão da média é $\sigma_M = \frac{\sigma}{\sqrt{N}}$.

Portanto, temos a seguinte receita prática para estimar médias estatísticas a partir de uma amostra grande (N >> 1) de medidas:

1. Calcule as médias amostrais $\overline{x}$ e $\overline{x^2}$.
2. Calcule σ e σ_M
3. O valor verdadeiro da média será $(\overline{x} \pm 3\sigma_M)$, com confiabilidade de 99,7%.

Esse resultado pressupõe que as N medidas da amostra sejam independentes umas das outras.

EXEMPLO 1: Simulação de lançamento de um dado *(e teste do gerador de números pseudo-aleatórios)*

Cada vez que o usuário aperta "Enter", o programa abaixo gera um lote de N lançamentos de um dado. Teoricamente, os valores possíveis são {1, 2, 3, 4, 5, 6}, cada um com probabilidade 1/6. O programa acumula os dados dos lotes, de modo que, a cada passo, temos N dados adicionais. Após cada conjunto de N dados, calculamos a média amostral e o desvio padrão da média. Esse valor é comparado com o valor teórico (1/6).

O programa termina quando o usuário manda Ctrl+Z.

NOTE o uso de variáveis inteiras do tipo "unsigned long int", para lidarmos com números inteiros grandes sem *overflow*.

```
#include <stdio.h>
#include <stdlib.h>

// An int variable can store an integer value in the range -32768 to +32767
// An unsigned int can store an integer value in the range 0 yo 65.535
// A long int can store an integer value in the range -2.147.483.648 to 2.147.483.647
// An unsigned long int can store an integer value in the range 0 to 4.294.967.295

/* tamanho do lote */
unsigned long int N = 500;


int main(){
   unsigned long int i, lote;
   int face, semente;
   char c, pm = 241;  // sinal +-
   unsigned long int s[7];
   double media, desvio, desvio_media, x1, x2, teorico;

   printf("\nEntre com uma semente entre 1 e 32767:\n");
   scanf("%d", &semente); printf("   semente = %d\n\n", semente);
   srand(semente);

   lote = 0;
   for(i=1;i<=6;i++)s[i] = 0;
```

```
while(1){
   c = fgetc(stdin);
   switch (c){
        case EOF : exit(0);
                   break;
        case '\n' :
             printf("\n");
             lote++;
             for(i=1;i<=N;i++){
               face = 1+rand()%6;
               s[face]++;
             }
             /* neste caso, temos x = 1 ou 0, de modo que <x> = <x*x> */
             printf("Lote %ld (total de %ld valores):\n", lote, (lote*N));
             for(i=1;i<=6;i++){ media = ((double) s[i])/((double)(lote*N));
                                /* intervalo de 99,7% */
                                desvio = 3.0*sqrt(media*(1.0 - media));
                                desvio_media = desvio/sqrt(lote*N);
                                x1 = media - desvio_media;  x2 = media + desvio_media;
                                teorico = 1.0/6.0;
                                printf("    face %d : %lf %c %lf [%lf,%lf] teorico = %lf\n",
                                        i, media, pm, desvio_media, x1, x2, teorico);
                              }
             break;
        default : break;
   }
   }

   printf("\n");
   system("pause");
   return 0;
}
```

Saídas típicas desse programa são como abaixo:

```
Lote 1 (total de 500 valores):
    face 1 : 0.164000 ± 0.049678 [0.114322,0.213678] teorico = 0.166667
    face 2 : 0.170000 ± 0.050396 [0.119604,0.220396] teorico = 0.166667
    face 3 : 0.170000 ± 0.050396 [0.119604,0.220396] teorico = 0.166667
    face 4 : 0.176000 ± 0.051092 [0.124908,0.227092] teorico = 0.166667
    face 5 : 0.158000 ± 0.048935 [0.109065,0.206935] teorico = 0.166667
    face 6 : 0.162000 ± 0.049433 [0.112567,0.211433] teorico = 0.166667
```

```
Lote 200 (total de 100000 valores):
    face 1 : 0.167480 ± 0.003542 [0.163938,0.171022] teorico = 0.166667
    face 2 : 0.165740 ± 0.003528 [0.162212,0.169268] teorico = 0.166667
    face 3 : 0.165630 ± 0.003527 [0.162103,0.169157] teorico = 0.166667
    face 4 : 0.165240 ± 0.003523 [0.161717,0.168763] teorico = 0.166667
    face 5 : 0.167650 ± 0.003544 [0.164106,0.171194] teorico = 0.166667
    face 6 : 0.168260 ± 0.003549 [0.164711,0.171809] teorico = 0.166667
```

```
Lote 2495 (total de 1247500 valores):
    face 1 : 0.166297 ± 0.001000 [0.165297,0.167298] teorico = 0.166667
    face 2 : 0.166366 ± 0.001000 [0.165366,0.167367] teorico = 0.166667
    face 3 : 0.166721 ± 0.001001 [0.165720,0.167722] teorico = 0.166667
    face 4 : 0.166101 ± 0.001000 [0.165101,0.167101] teorico = 0.166667
    face 5 : 0.166939 ± 0.001002 [0.165937,0.167940] teorico = 0.166667
    face 6 : 0.167576 ± 0.001003 [0.166573,0.168579] teorico = 0.166667
```

☞ Observe que, como o desvio padrão da média diminui com $\sqrt{N}$, é necessário 100 vezes mais medidas para se diminuir o desvio de 10. No exemplo acima, seriam necessários cerca de 125.000.000 de lances para se estimar a média com desvio de 0,0001. *(isso iria levar mais de uma hora de CPU em um INTEL CORE2 QUAD 2.4GHz).*

EXERCÍCIO 1: Modifique o programa acima para estimar a probabilidade de se obter, em um lançamento de dois dados, a soma das faces maior do que 7. Qual o valor teórico?

EXEMPLO 2: O programa "caça-casamento" (ou o problema do fornecedor)

Imagine alguém que esteja preocupado em encontrar uma boa estratégia para se casar. O objetivo é fazer um bom casamento. Como casar é dispendioso e é uma decisão delicada, uma boa pergunta é "quantos candidatos devo examinar até decidir me casar?".

Vamos supor que essa pessoa venha a conhecer, dentro de um tempo máximo, N candidatos em potencial para se casar. Naturalmente, uns são melhores que outros. Mas, se ela recusar um candidato (ou uma candidata), dificilmente poderá voltar atrás.

Essa pessoa resolve então fazer o seguinte: examinar K candidatos, e só aceitar o que vier após esses K e que seja melhor que qualquer um desses K. Note que, se ele não encontrar candidato melhor que os K anteriores, então vai ter de se casar com o último que aparecer.

Qual o melhor valor de K (ou seja, qual o valor de K que maximiza a probabilidade dela ter um bom casamento)?

O problema é que os candidatos aprecem um tanto que ao acaso na vida da pessoa: pode ser que o melhor deles venha logo no inicio, ou apenas no final. Note que, se ela escolher, digamos, K = 2, e se o melhor vier primeiro, ela já perdeu esse pretendente. Se a pessoa esperar muito (K >> 1), pode perder todas as boas oportunidades. Se esperar pouco (K ~ 1), pode deixar de examinar bons candidatos que poderiam vir no futuro.

Por exemplo, suponha que alguém esteja disposto a examinar no máximo 5 candidatos. Vamos dizer que esses candidatos apareçam em uma seqüência qualquer. E cada candidato tem um valor numérico meio que aleatório, entre 0 e 10; por exemplo (2,5; 4,8; 3,6; 6,8; 5,3) . Se o casamenteiro escolher K = 0, ele se casa com o primeiro que vier, e seu casamento terá nota 2,5. Se ele escolher K = 1, ele vai se casar com o segundo, e o casamento será 4,8. Para K = 2, ele vai recusar o terceiro e vai aceitar o quarto, conseguindo 6,8; o mesmo vai acontecer para K = 3. Se K = 4, ele vai ter um casamento 5,3. Portanto, a melhor escolha, para esse caso, seria K = 2 ou 3. Mas se a sequencia fosse (7,8; 8,9; 5,2; 4,5; 2,3), K=2 ou K=3 seria um péssimo negócio.

O programinha abaixo dá uma ajuda ao casamenteiro em dúvida. Ele examina um grande número de possibilidades, e dá uma nota de 0 a 10 para cada escolha de K. Naturalmente, o modelo matemático é simples demais e ignora mil e um fatos da vida que, na verdade, acabam definindo os casamentos.

Os passos são os seguintes:

1. O usuário escolhe um valor de Nc = número máximo de candidatos que ele estaria disposto a examinar antes de se casar. Uma pessoa com uma vida agitada e muitos amigos pode escolher um valor grande para Nc (talvez dez, quinze ou vinte...). Já uma pessoa mais quieta e tímida escolheria, digamos, Nc = 4 ou 5.

2. O programa então dá notas aleatórias, entre 0 e 10, a cada um dos Nc candidatos.

3. Em seguida, examina a nota do candidato que será aceito para cada valor de K, de 0 a (Nc-1).

4. Repete os passos 2 e 3 para um número grande N de seqüências de possíveis Nc candidatos.

5. Calcula a média das notas dos casamentos que serão realizados, e o desvio padrão da média.

6. Cada vez que o usuário aperta "Enter", o programa examina mais um lote de N possíveis seqüências e acrescenta esses dados ao lote anterior.

7. Quando o usuário estiver satisfeito (ou seja, quando os resultados tiverem desvios padrões da média satisfatoriamente pequenos), ele pode usar a tabela final para decidir quantos candidatos ele deve esperar antes de escolher um que seja melhor do que qualquer um dos anteriores.

8. O programa termina com Ctrl+Z.

NOTE que o modelo pode ser modificado à vontade:

- mudando o modo de se dar notas aos candidatos
- mudando a resolução das notas dos candidatos (o programa abaixo dá notas de 0.2 em 0.2)
- mudando o critério de "melhor do que qualquer um dos anteriores"
- etc. etc. etc.

```
#include <stdio.h>
#include <stdlib.h>

unsigned long int N = 50;   // tamanho do lote

/* Numero maximo de candidatos + 1 */
#define NcMax 21

int main(){
   unsigned long int lote, n;
   int Nc, i, semente, Nnotas, j, l, k, encontrou;
   char c, pm = 241;  // sinal +-
   double Cand[NcMax], Score[NcMax], Score2[NcMax], media, media2, desvio, desvio_media, x1, x2;

   Nc = NcMax;
   while( (Nc > NcMax-1) || (Nc < 2) ){
      printf("\nEntre com o numero esperado de candidatos:\n");
      scanf("%d", &Nc);
      if( Nc > NcMax-1 )printf("\n ---> No maximo %d candidatos !!!\n", (NcMax-1));
      if( Nc < 2 )printf("\n ---> No minimo dois !!!\n");
   }
   printf("   %d candidatos\n\n", Nc);

   printf("\nEntre com uma semente entre 1 e 32767:\n");
   scanf("%d", &semente); printf("   semente = %d\n\n", semente);
   srand(semente);

   lote = 0;
   // inicializacao dos scores e dos quadrados
   for(i=1;i<=Nc;i++)Score[i] = Score2[i] = 0;

 while(1){
   c = fgetc(stdin);
   switch (c){
         case EOF : exit(0);
                  break;
         case '\n' :
               printf("\n");
               lote++;
               // Loop da amostra com N medidas cada
               for(n=1;n<=N;n++){
                  // As notas dos candidatos, aleatoriamente distribuidas entre 0 e 10 com resolucao de 0.2
                  for(i=1;i<=Nc;i++)Cand[i] = ((double)(rand()%51))/5.0;
                  // procura pelo proximo melhor que os (i-1) anteriores
                  i = 1; // neste caso, fica com o primeiro que aparece
                  Score[1] += Cand[1];  Score2[1] += Cand[1]*Cand[i];
                  for(i=2;i<=Nc;i++){
                    for(k=i; k<=Nc; k++){
                      encontrou = 1;
                      for(l=1;l<k;l++)if(Cand[l] > Cand[k]){ encontrou = 0;  break;}
                      // se nao ha melhor candidato, fica com o ultimo
                      if (encontrou){Score[i] += Cand[k];  Score2[i] += Cand[k]*Cand[k]; break;}
                    }
                    if (!encontrou){Score[i] += Cand[Nc];  Score2[i] += Cand[Nc]*Cand[Nc];}
                  }
               }
               printf("Lote %ld  (%ld casos examinados):\n", lote, (lote*N));
               // calcula a media das notas e o desvio padrao da media em cada caso
               /* intervalo de 99,7% */
               for(i=1;i<=Nc;i++){ media = ((double) Score[i])/((double)(lote*N));
                                   media2 = ((double) Score2[i])/((double)(lote*N));
                                   /* intervalo de 99,7% */
                                   desvio = 3.0*sqrt(media2 - media*media);
                                   desvio_media = desvio/sqrt(lote*N);
                                   x1 = media - desvio_media;  x2 = media + desvio_media;
                                   printf("    Tomando o melhor apos %d : %lf %c %lf [%lf,%lf]\n",
                                             (i-1), media, pm, desvio_media, x1, x2);
               }
               break;
         default : break;
   }
 }

 printf("\n");
 system("pause");
 return 0;
}
```

Saídas típicas desse programa são como abaixo:

```
Entre com o numero esperado de candidatos:
8
   8 candidatos


Entre com uma semente entre 1 e 32767:
5691
   semente = 5691


Lote 1  (50 casos examinados):
    Tomando o melhor apos 0 : 5.052000 ± 1.323901 [3.728099,6.375901]
    Tomando o melhor apos 1 : 7.028000 ± 1.112706 [5.915294,8.140706]
    Tomando o melhor apos 2 : 6.920000 ± 1.128128 [5.791872,8.048128]
    Tomando o melhor apos 3 : 6.536000 ± 1.207231 [5.328769,7.743231]
    Tomando o melhor apos 4 : 6.268000 ± 1.218648 [5.049352,7.486648]
    Tomando o melhor apos 5 : 6.168000 ± 1.295818 [4.872182,7.463818]
    Tomando o melhor apos 6 : 5.548000 ± 1.282464 [4.265536,6.830464]
    Tomando o melhor apos 7 : 4.992000 ± 1.240705 [3.751295,6.232705]
```

```
Lote 500  (25000 casos examinados):
    Tomando o melhor apos 0 : 5.006376 ± 0.055519 [4.950857,5.061895]
    Tomando o melhor apos 1 : 6.930840 ± 0.047122 [6.883718,6.977962]
    Tomando o melhor apos 2 : 7.185848 ± 0.047925 [7.137923,7.233773]
    Tomando o melhor apos 3 : 6.986632 ± 0.052217 [6.934415,7.038849]
    Tomando o melhor apos 4 : 6.607592 ± 0.056030 [6.551562,6.663622]
    Tomando o melhor apos 5 : 6.105528 ± 0.058232 [6.047296,6.163760]
    Tomando o melhor apos 6 : 5.571008 ± 0.058252 [5.512756,5.629260]
    Tomando o melhor apos 7 : 4.978472 ± 0.055769 [4.922703,5.034241]
```

Os resultados acima indicam que, para 8 candidatos no total, o melhor parece ser recusar os dois primeiros, casando com o primeiro que vier depois e que seja melhor que esses dois. É mau negócio recusar mais do que 4 ou 5 candidatos.

EXERCÍCIO 2: Modifique a linha que dá notas aos candidatos, de modo a escolher notas entre 2,0 e 10,0 com resolução de 0,4.

EXEMPLO 3: Monty-Hall era um programa de auditório na TV americana que consistia no seguinte:

1. Havia três portas no palco. Atrás de uma das portas tinha um carro. Atrás das outras havia cabras;
   (a posição do carro e das cabras não mudava durante o programa)

2. O apresentador dizia a um participante para escolher uma das portas, tentando adivinhar onde estaria o carro. O participante então apontava para uma das portas.

3. Em seguida, o apresentador (que sabia em que porta estava o carro) ia a uma das outras duas portas que não haviam sido escolhidas e abria uma que tinha uma cabra. Sobravam então duas portas fechadas.

4. Num lance de suspense, o apresentador perguntava ao participante se ainda desejava manter a escolha, ou se agora preferia escolher a outra porta que ainda estava fechada.

EXERCÍCIO 3: Qual a melhor opção? Manter a escolha original ou trocar de porta? Ou tanto faz, "dá na mesma"?

O programa abaixo simula esse joguinho (não fica muito atraente, porque ele roda numa tela sem-graça e só com caracteres ASCII).

Na verdade, é possível demonstrar matematicamente qual é a melhor opção, sem muita dificuldade. Mas o joguinho tem seu charme, de qualquer jeito. E pode ser modificado para estimar as probabilidades de ganhar o carro caso o participante troque de porta ou caso mantenha a escolha inicial.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>

double w = 1.0; // fator de espera para fazer suspense

void wait(double seconds){
  time_t time_fin, time_ini, d_time;   long elapsed_time;   int rest, ns;   time_ini = time(NULL);
  ns = 0;
  while(ns<seconds){
              time_fin = time(NULL);
              d_time = time_fin - time_ini;
              elapsed_time = (long) d_time;
              rest = elapsed_time % 3600;
              ns = rest % 60; }
}
```

```
int main(){
 unsigned int tim;
 int p[4], i, k, j, m, n, ok;
 char b, c, t, r[]={' ','A','B','C'}, x, y, z;

 // semente inicial - usando o tempo de calendario
 tim = (int) M_PI*((long) time(NULL))%100000;
 srand(tim);

b = 's';
 while( b == 's' || b == 'S'){
   // colocar um carro atras de uma das portas A=p[1], B=p[2] ou C=p[3]  (k)
   for(i=1;i<=3;i++)p[i]=0;
   k = 1+rand()%3;
   p[k]=1;

   system("cls");
   printf("\n Voce esta diante de tres portas A B C \n");
   printf("\n           A   B   C\n");
   printf("\n Apenas uma delas esconde um carro.\n\nQual delas voce escolhe?\n\n");

   i = 0;
   while(i==0){
     scanf("%c", &c);
     switch (c){
          case 'A' : i=1; printf("\n          x   B   C\n"); break;
          case 'B' : i=2; printf("\n          A   x   C\n"); break;
          case 'C' : i=3; printf("\n          A   B   x\n"); break;
          default  : break;
     }
     if(i==0)printf("\n   ---> escolha A B ou C <---\n");
  }

  // escolher aleatoriamente uma das duas portas que sobrou e que nao contem o carro (j)
  ok = 0;
  while(!ok){
    j = (1+rand()%3);
    if( (p[j] == 0) && (j != i) )ok = 1;
  }

  // indice da porta que resta  (n)
  ok = 0; n = 0;
  while(!ok){
    n++; ok=1;
    if( (n == i) || (n == j) )ok=0;
  }
  wait(2.5*w); // esperar um pouco para fazer suspense

  printf("\n Muito bem. Agora, eu digo a voce que o carro TAMBEM NAO ESTA na porta %c\n", r[j]);
  // formato de impressao das portas
  if (j==1)x = '_'; else if (j==2) y = '_'; else z = '_';
  if (i==1)x = 'x'; else if (i==2) y = 'x'; else z = 'x';
  if (n==1)x = 'A'; else if (n==2) y = 'B'; else z = 'C';
  printf("\n           %c   %c   %c\n", x, y, z);

  wait(7.0*w); // esperar um pouco para fazer suspense

  printf("\n Voce continua apostando que o carro esta na porta %c,", r[i]);
  printf("\n      ou prefere trocar sua escolha para a porta %c ?\n", r[n]);

   printf("\n Troca(s) ou nao troca(n) ?\n");
   scanf("%c", &t);
   while( t != 's' && t != 'S' && t != 'n' && t != 'N' )scanf("%c", &t);

   m = i;
   if (t == 's' || t == 'S')for(m=1;m<=3;m++)if( (m != i) && (m != j) )break;
   i = m;

   if (i == k)printf("\n ***** :-)) VOCE GANHOU O CARRO  :-)) *****\n");
   else printf("\n   :-( Sinto muito... o carro estava na porta %c ... :-( \n", r[k]);

   printf("\n s para jogar novamente, n para sair\n");
   scanf("%c", &b);
   while( b != 's' && b != 'S' && b != 'n' && b != 'N' )scanf("%c", &b);
 }

  return 0;
}
```