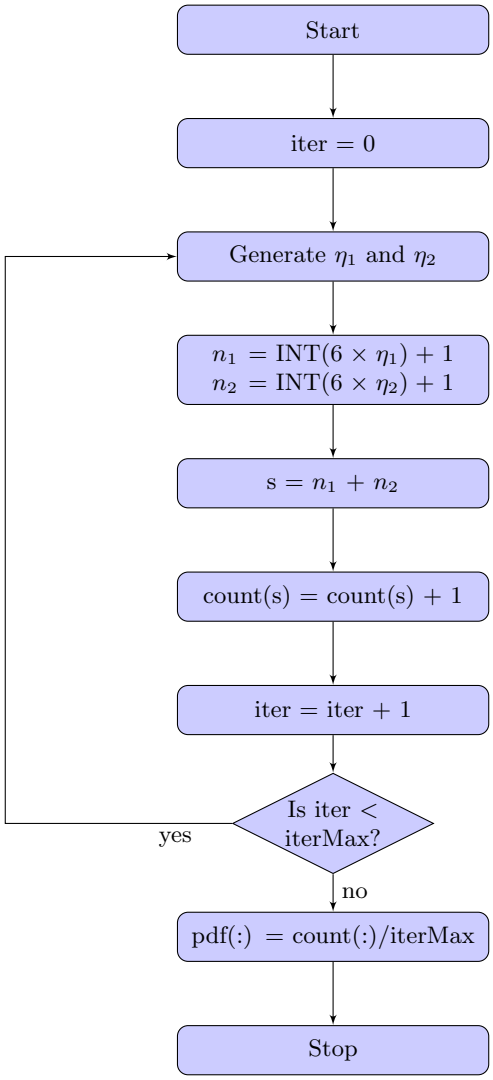
NSEG-5984 Monte Carlo Methods for Particle Transport

Nagendra Krishnamurthy

August 31, 2011

Problem 1: *Diagram a flowchart for an algorithm for randomly selecting the sums of the top faces (n1,n2) on a pair of well-balanced cubical dice based on generated random numbers h's. Write a program to obtain the pdf for the random variable s (=n1+n2).*

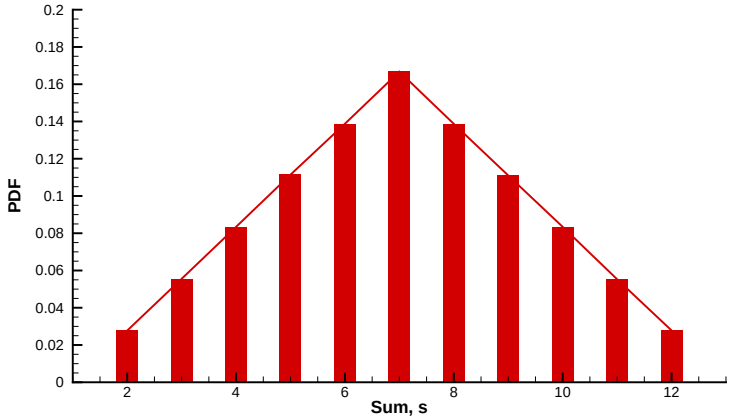


Source code:

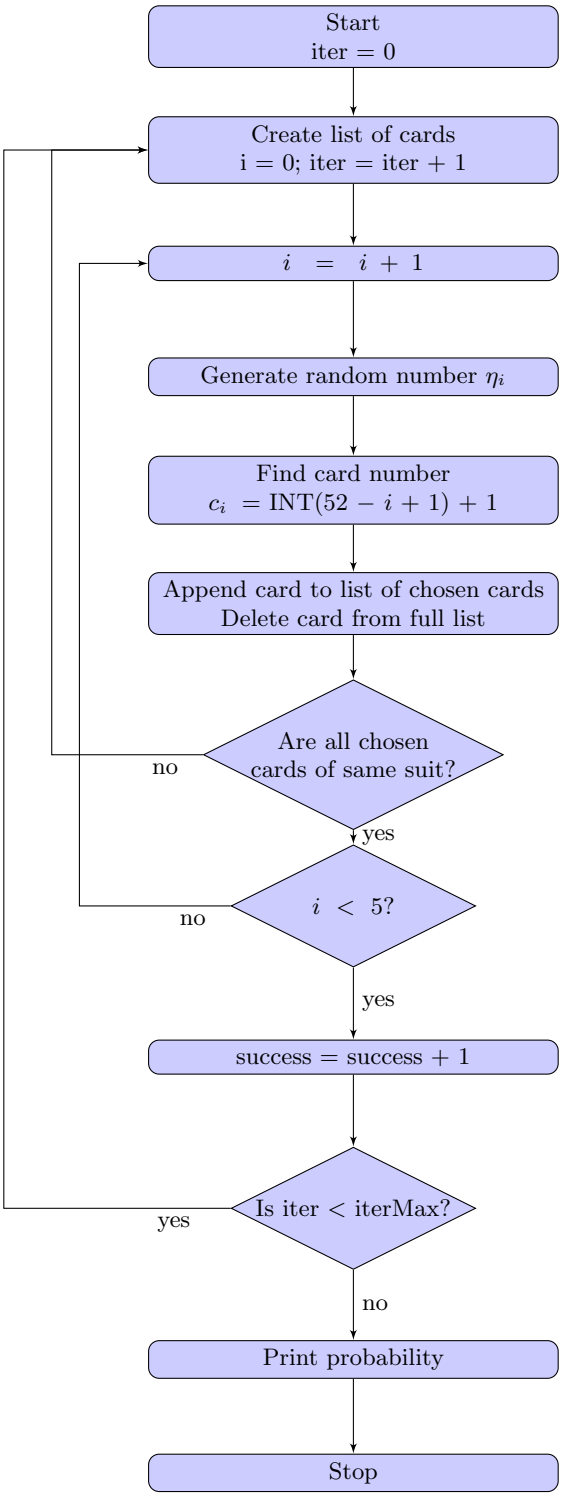
```
PROGRAM sumDice

  INTEGER :: iter, maxIter, n1, n2, sum, unitPdfFile
  REAL :: ranNum_1, ranNum_2, pdfBin(2:12)
  CHARACTER(80) :: namePdfFile

  ! Start

  PRINT*, 'Enter the maximum number of iterations:'
  READ*, maxIter

  CALL random_seed()

  iter = 0

  DO WHILE (iter .le. maxIter)

  ! Increment iteration number

  iter = iter + 1

  ! Generate two random numbers
  CALL random_number(ranNum_1)
  CALL random_number(ranNum_2)

  ! Compute values for n1 and n1 based on the random numbers
  n1 = INT(6.*ranNum_1) + 1
  n2 = INT(6.*ranNum_2) + 1

  ! Computer sum of the two die values
  sum = n1 + n2

  ! Bin the sum for computation of pdf
  pdfBin(sum) = pdfBin(sum) + 1

  END DO

  ! Computer pdf by dividing the bin by total number of iterations
  pdfBin(:) = pdfBin(:)/maxIter

  unitPdfFile = 101
  namePdfFile = 'pdf.dat'

  OPEN (UNIT = unitPdfFile, FILE = namePdfFile, STATUS = 'replace',  &
      POSITION = 'rewind', FORM = 'formatted', ACTION = 'write')

  DO i = 2, 12
    WRITE(unitPdfFile,501) i, pdfBin(i)
  END DO

501 FORMAT (i3.3, 1X, 1(pe12.5, 1X))

END PROGRAM sumDice
```

problem_1.f90

The following plot provides the probability density function obtained. The code was executed for iterMax = 1000000.



Problem 2: *Diagram a flowchart and write a program for an algorithm for randomly selecting a "flush poker" hand based on generated random numbers, η's, considering that all cards are from the same suit. A flush poker hand consists of 5 cards from a deck of 52 cards. A deck consists of 13 cards in each of 4 suits, spades, hearts, diamonds, and clubs. The 13 cards in each suit are numbered from 1 to 10 plus jack, queen, and king.*



Source code:

```
PROGRAM flushPokerHand

  IMPLICIT NONE
```

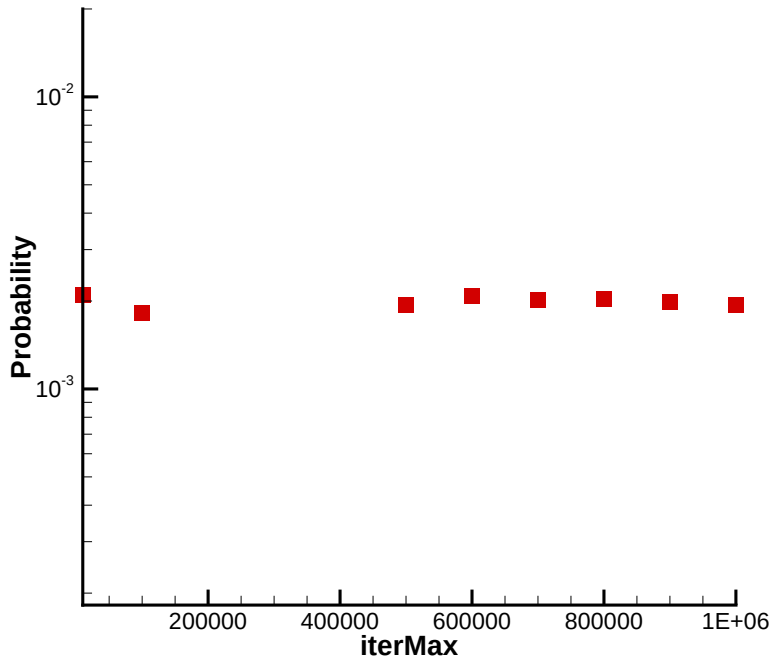
```
  TYPE card
    INTEGER :: cardNum, cardType
    TYPE (card), POINTER :: next
  END TYPE card

  INTEGER :: iter, maxIter, c1, unitProbFile, i, nCards,              &
      successCount, iCard
  REAL :: ranNum, probFlush
  CHARACTER(80) :: nameProbFile

  TYPE (card), POINTER :: cardList, cardCurrent, cardLast, handList,  &
      handCurrent, handLast, cardTemp

  ! Start

  PRINT*, 'Enter maximum number of iterations'
  READ*, maxIter

  successCount = 0
  nCards = 5
  CALL random_seed()

  ALLOCATE(cardList)
  NULLIFY(cardList%next)
  cardLast => cardList

  ALLOCATE(handList)
  NULLIFY(handList%next)
  handLast => handList

  iter = 0

  DO WHILE (iter .lt. maxIter)

300 CONTINUE

    CALL deleteList(cardList)
    cardLast => cardList
    CALL deleteList(handList)
    handLast => handList

    ! Increment iteration number
    iter = iter + 1

    ! Create a list of cards
    DO iCard = 1, 52

      ALLOCATE(cardLast%next)
      NULLIFY(cardLast%next%next)

      cardLast => cardLast%next
      cardLast%cardNum = iCard
      cardLast%cardType = CEILING(iCard/13.0)

    END DO

    DO i = 1, nCards

      ! Generate random number
      CALL random_number(ranNum)

      ! Compute values for c1 based on the random numbers
      c1 = INT((52.0-i+1)*ranNum) + 1

      ! Find the card
```

```
      cardCurrent => cardList
      DO iCard = 1, c1
        cardCurrent => cardCurrent%next
      END DO

      ! Append card to handList
      ALLOCATE(handLast%next)
      NULLIFY(handLast%next%next)

      handLast => handLast%next
      handLast%cardNum = cardCurrent%cardNum
      handLast%cardType = cardCurrent%cardType

      ! Remove card from cardList
      cardCurrent => cardList
      DO iCard = 1, c1-1
        cardCurrent => cardCurrent%next
      END DO
      cardTemp => cardCurrent%next
      cardCurrent%next => cardCurrent%next%next
      DEALLOCATE(cardTemp)

      ! Check if all cards in hand are of same suit
      handCurrent => handList
      DO WHILE (ASSOCIATED(handCurrent%next%next))
        IF (handCurrent%next%cardType .NE.                                  &
            handCurrent%next%next%cardType) THEN
          GOTO 300
        END IF
        handCurrent => handCurrent%next
      END DO

    END DO

    ! Success
    successCount = successCount + 1
    handCurrent => handList
    DO WHILE (ASSOCIATED(handCurrent%next))
      handCurrent => handCurrent%next
    END DO

  END DO

  ! Compute probability of flush hand
  probFlush = 1.0*successCount/maxIter

  unitProbFile = 101
  nameProbFile = 'probFlush.dat'

  OPEN (UNIT = unitProbFile, FILE = nameProbFile,                           &
      POSITION = 'append', FORM = 'formatted', ACTION = 'write')

  WRITE(unitProbFile,501) maxIter, probFlush

501 FORMAT (i7.7, 1X, 1(e12.5, 1X))

CONTAINS

SUBROUTINE printList(list)

  IMPLICIT NONE

  TYPE (card), POINTER, INTENT(IN) :: list
  TYPE (card), POINTER :: current

  current => list
```

```
  DO WHILE (ASSOCIATED(current%next))
    current => current%next
    PRINT*, current%cardNum, current%cardType
  END DO

END SUBROUTINE printList

SUBROUTINE deleteList(list)

  IMPLICIT NONE

  TYPE (card), POINTER, INTENT(IN) :: list
  TYPE (card), POINTER :: current, previous

  current => list%next
  DO WHILE (ASSOCIATED(current))
    previous => current
    current => current%next
    DEALLOCATE(previous)
  END DO
  NULLIFY(list%next)

END SUBROUTINE deleteList

END PROGRAM flushPokerHand
```

problem_2.f90

The code was executed was various values of iterMax. From the plot, it is concluded that the probability of a flush hand is approximately 1.9×10^{-3}.



Problem 3: *Consider a continuous random variable x defined in a range $[0, 3]$ with a distribution $f(x) = x^2$. a) Determine the pdf of this random variable; b) Write a program for selecting x using a random number (η).*

Given distribution function:

$$f(x) = x^2 \tag{1}$$

Probability density function (PDF) will be given by:

$$p(x) = \frac{f(x)}{\int_0^3 f(x)\,dx} \tag{2}$$

$$p(x) = \frac{x^2}{\int_0^3 x^2\,dx} = \frac{x^2}{\frac{x^3}{3}\Big|_0^3} \tag{3}$$

$$p(x) = \frac{x^2}{9} \tag{4}$$

Fundamental formulation of Monte Carlo:

$$\int_0^x p(x')\,dx' = \eta \tag{5}$$

$$\int_0^x \frac{x'^2}{9}\,dx' = \eta \tag{6}$$

$$\frac{x^3}{27} = \eta \tag{7}$$

$$x = 3\sqrt[3]{\eta} \tag{8}$$

Source code:

```
PROGRAM randDist

  IMPLICIT NONE

  INTEGER :: iter, bin(30), i, maxIter, unitProbFile
  REAL :: power, randNum, x, xBin, pdf(30)
  CHARACTER(80) :: nameProbFile

  ! Start

  PRINT*, 'Enter maximum number of iterations'
  READ*, maxIter

  CALL random_seed()

  power = 1./3.

  iter = 0
  bin(:) = 0

  DO WHILE (iter .lt. maxIter)
```

```
    iter = iter + 1

    CALL RANDOM_NUMBER(randNum)

    x = 3.0*randNum**(power)
    xBin = CEILING(x*10.0)

    bin(xBin) = bin(xBin) + 1

  END DO

  ! Compute probability of flush hand
  pdf(:) = 1.0*bin(:)/maxIter

  unitProbFile = 101
  nameProbFile = 'pdf_x2.dat'

  OPEN (UNIT = unitProbFile, FILE = nameProbFile,                    &
      POSITION = 'rewind', FORM = 'formatted', ACTION = 'write')

  DO i = 1, 30
    WRITE(unitProbFile,501) i, pdf(i)
  END DO

501 FORMAT (i3.3, 1X, 1(e12.5, 1X))

END PROGRAM randDist
```

problem_3.f90