HW #4 Due: 26 November 2001

1. Estimate the time needed to solve the Nth order system of linear equations

$$Ax = r$$

using Strassen's "divide and conquer" algorithm. That is, what is the coefficient K in the term $K\left(\mu N^{\lg 7}\right)$?

Solution:

Write

$$Ax = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \equiv \begin{pmatrix} I & 0 \\ A_{21}A_{11}^{-1} & I \end{pmatrix}\begin{pmatrix} A_{11} & A_{12} \\ 0 & z \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \equiv \begin{pmatrix} I & 0 \\ A_{21}A_{11}^{-1} & I \end{pmatrix}\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \end{pmatrix}$$

where

$$z \overset{df}{=} A_{22} - A_{21}A_{11}^{-1}A_{12} \equiv A_{22} - w\,A_{12}$$

and

$$w \overset{df}{=} A_{21}A_{11}^{-1}\,.$$

We see that

$$\begin{aligned} u_1 &= r_1 \\ u_2 &= r_2 - w\,r_1 \\ z\,x_2 &= u_2 \\ x_1 &= A_{11}^{-1}\left(r_1 - A_{12}\,x_2\right). \end{aligned}$$

If it takes time L_N to solve N equations; time M_N to multiply, and T_N to invert $N \times N$ matrices; then divide and conquer gives (recall $T_N \underset{N\to\infty}{\to} M_N$)

$$L_N = L_{N/2} + 3M_{N/2} + \frac{3}{4}\mu N^2 \to L_{N/2} + \frac{3}{7}\mu N^{\lg 7} + \frac{3}{4}\mu N^2\,.$$

Letting $\frac{L_N}{N^{\lg 7}} = \tau_k$ where $k = \lg N$, we have, for large N,

$$\tau_k = \frac{1}{7}\tau_{k-1} + \frac{3}{7}\mu$$

whose solution is $\tau_k = \mu/2$, giving $L_N \sim \frac{\mu}{2}N^{\lg 7}$.

2. Solve the recursion relation

$$\tau_N = N\lambda + 2\,\tau_{N/2}$$

Solution

We let $N = 2^k$ and write $\frac{\tau_N}{N} = \sigma_k$; then

$$\sigma_k = \sigma_{k-1} + \lambda$$

whose solution is $\sigma_k = (k+1)\lambda$. Thus, $\tau_N = \lambda N(1 + \lg N) \sim \lambda N \lg N$.

3. Using the inverse formula, a good uniform prng and a root-finder of your choice, generate a table of 10,000 random variates from the semi-circular distribution

$$p(x) \overset{df}{=} 6\,x(1-x)\,,\quad 0 \le x \le 1$$

Make a histogram of these values and compare with the distribution function.

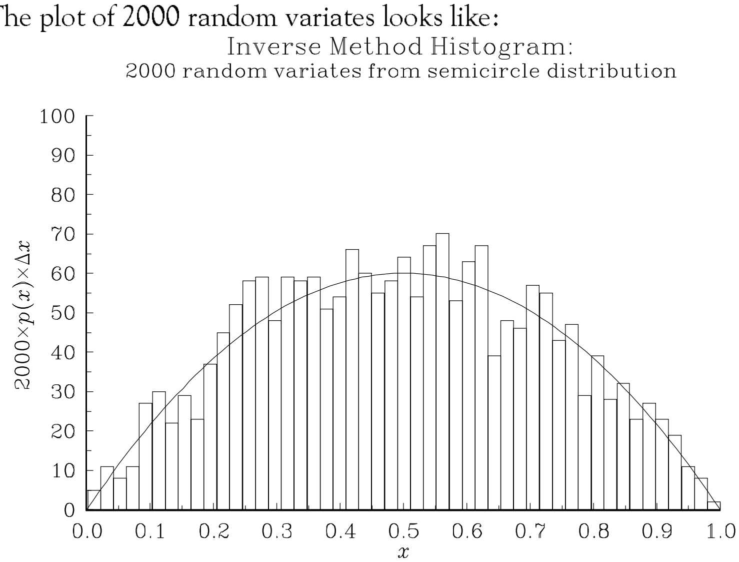
Solution

The program that does this is shown below. (It also includes the Von Neumann-Metropolis selection method.)

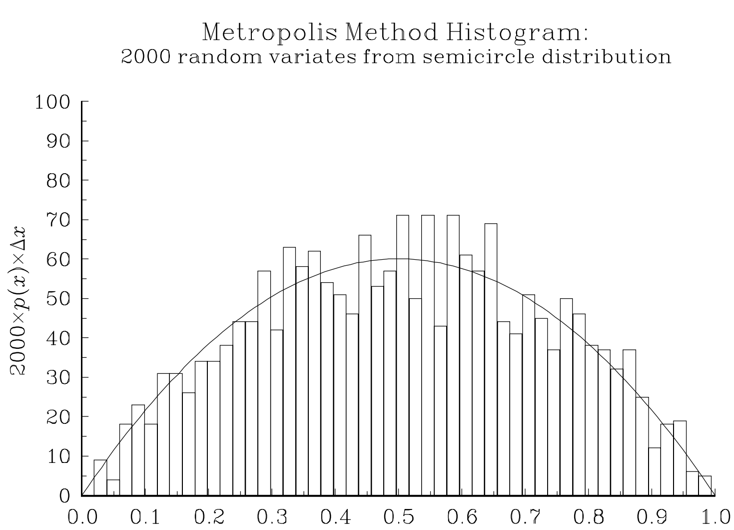
```
\ generate 10^4 random variates from the
\ semicircular distribution  December 3rd, 2001 - 16:28

\ ---------------------------------------------------
\     (c) Copyright 2001  Julian V. Noble.          \
\       Permission is granted by the author to      \
\       use this software for any application pro-  \
\       vided this copyright notice is preserved.   \
\ ---------------------------------------------------

\ This is an ANS Forth program requiring the
\   FLOAT, FLOAT EXT, FILE and TOOLS EXT wordsets.
\
\ Environmental dependences:
\       Assumes independent floating point stack


marker  -hw4

include prng.f      test
include ansfalsi.f

FVARIABLE xi

: f^2    FDUP   F*   ;

: f-rot    FROT   FROT   ;

: f0    ( f: x -- 6*x*[1-x])
    1e0  FOVER  F-   F*   6e0   F*   ;
```

```
: f1    ( f: x -- [3*x^2-2*x^3 - xi])
    FDUP  f^2                   ( f: x x^2)
    3e0   FOVER F*              ( f: x x^2 3*x^2)
    f-rot   2e0  F*  F*  F-     ( f: 3*x^2-2*x^3)
    xi F@  F-   ;

: invP      ( f: xi -- X[xi])   xi F!
    use( f1  0e0  1e0  1e-6 )falsi    ( f: -- e^{-X})
;

: variates    0 do  prng  invP   cr  f.  loop  ;

: metropolis    ( # of variates)
    BEGIN
        prng  FDUP  f0
        prng  1.5e0 F*  F>    IF  CR F.  1-  ELSE  FDROP  THEN
    ?DUP  0=  UNTIL
;
```

The plot of 2000 random variates looks like:



4. Repeat problem #3 but use the Von Neumann-Metropolis selection method to generate the random variates. Here is the histogram:



5. Use 5 point Gauss-Hermite integration (look up the points and weights in Abramowitz & Stegun) to evaluate the following integrals on the interval from $-\infty$ to $+\infty$:

   a. $e^{-x^2}\cos x$

   b. $e^{-x^2}\cos 2x$

   c. $e^{-x^2}\cos 3x$

   d. $e^{-x^2}\left(x^4-2x^2+1\right)$

   Compare the results with the exact answers and discuss the loss of precision (if any!); if you do not know how to do these integrals look them up in tables.

Solution: A program that does this, with results, is

```
\ 5 point Gauss-Hermite integration

MARKER -gauss

needs ftran111.f

 2.020182870456086e  FVARIABLE x2  x2  F!
 0.958572464613819e  FVARIABLE x1  x1  F!
 0e                  FVARIABLE x0  x0  F!
 9.453087204829e-1   FVARIABLE w0  w0  F!
 3.936193231522e-1   FVARIABLE w1  w1  F!
 1.995324205905e-2   FVARIABLE w2  w2  F!

v: fdummy

: )int      ( f: -- integral)  ( xt --)
    defines  fdummy
    f" w0 * fdummy(x0) + w1 * (fdummy(x1) + fdummy(-x1))
        + w2 * (fdummy(x2) + fdummy(-x2)) "
;

FALSE [IF]
Examples:

10 set-precision  ok

use( fcos )intf. 1.380390076  ok    \ exact = 1.380388447043

: f1   f2*  fcos  ;  ok
use( f1 )int f. .6532237524  ok     \ exact = 0.6520493321733

: f2   3e0 f*   fcos  ;  ok
use( f2 )int f. .2246529014  ok     \ exact = 0.1868152614571

: f3   fcos  f^2  ;  ok
use( f3 )int f. 1.212838802  ok     \ exact = 1.212251591539

: f4   f^2   fdup  -2e0  f+  f*  1e0 f+  ;  ok
use( f4 )int f. 1.329340388  ok     \ exact = 1.329340388179
[THEN]
```

6. Use 5-point Gauss-Laguerre integration to evaluate the integrals (on the interval 0 to +∞)

a. $e^{-x}\cos x$

b. $e^{-x}\cos 2x$

c. $e^{-x}\sin x$

d. $e^{-x}\left(x^{10}-2x^{5}+1\right)$

Evaluate the integrals exactly and compare with the numerical results; discuss the loss of precision (if any!) for each case.

Solution: A program that does this, with results, is

```
\ 5 point Gauss-Laguerre integration

MARKER -gauss

needs ftran111.f

12.640800844276e0    FVARIABLE x4  x4  F!
 7.085810005859e0    FVARIABLE x3  x3  F!
 3.596425771041e0    FVARIABLE x2  x2  F!
 1.413403059107e0    FVARIABLE x1  x1  F!
 0.263560319718e0    FVARIABLE x0  x0  F!

 5.21755610583e-1    FVARIABLE w0  w0  F!
 3.98666811083e-1    FVARIABLE w1  w1  F!
 7.59424496817e-2    FVARIABLE w2  w2  F!
 3.61175867992e-3    FVARIABLE w3  w3  F!
 2.33699723858e-5    FVARIABLE w4  w4  F!

v: fdummy

: )int      ( f: a b -- integral)  ( xt --)
    defines  fdummy
    f" w0 * fdummy(x0) + w1 * fdummy(x1) + w2* fdummy(x2)
        + w3 * fdummy(x3) + w4 * fdummy(x4) "
;

FALSE [IF]
Examples:

10 set-precision

use( fcos )int f. .5005384852  ok       \ exact = 0.5

: f1   f2*  fcos  ;
use( f1 )int f. .1183827839  ok         \ exact = 0.2

use( fsin )int f. .4989033210  ok       \ exact = 0.5

: f2   f2*  fsin  ;  ok
use( f2 )int f. .4494545483  ok         \ exact = 0.4

( over )
```

```
: f3   fdup  f^4  f*   fdup  2e0 f-  f*  1e0 f+  ;
use( f3 )int f. 3614161.000  ok         \ exact = 3628561

: f4   f^4  fdup  2e0 f-  f*  1e0  f+  ;  ok
use( f4 )int f. 40273.00000  ok         \ exact =  40273
[THEN]
```