Homework 4: ECON512

Joonkyo Hong

For question 1 to 4, I use 100 draws (or nodes) to approximate π.

Q1. (Quasi Monte Carlo with Dart Throwing: q-MC with DT)

1. Draw $\{x_i, y_j\}_{i=1}^{100}{}_{j=1}^{100}$ from `qnwequi` over $[0,1] \times [0,1]$.
2. Compute

$$z_{ij} \begin{cases} 1, & \text{if } x_i^2 + y_j^2 \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

3. Approximate $\frac{\pi}{4}$ as $\frac{\#\text{ of } z_{ij}=1}{100 \times 100}$

Q2. (Newton-Cotes with Dart Throwing: NC with DT)

1. Create grids $\{x_i, y_j\}_{i=0}^{100}{}_{j=0}^{100}$ where $x_i = \frac{i}{100}$ and $y_j = \frac{j}{100}$.
2. Create the matrix Z whose (i, j)-th element is

$$z_{ij} \begin{cases} 1, & \text{if } x_i^2 + y_j^2 \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

3. Following Simpson, create the 101 by 1 weight vector w as

$$w_0 = w_{101} = \frac{0.01}{3}$$
$$w_k = \begin{cases} \frac{4}{3} \times 0.01, & \text{if } k \text{ is even} \\ \frac{2}{3} \times 0.01, & \text{if } k \text{ is odd} \end{cases}$$

4. Approximate $\frac{\pi}{4}$ as $w'Zw$

Q3. (Quasi Monte Carlo with Pythagorean formula: q-MC with PYT)

1. Draw $\{x_i\}_{i=1}^{100}$ from `qnwequi`
2. Approximate $\frac{\pi}{4}$ as $\frac{1}{100}\sum_{i=1}^{100}\sqrt{1-x_i^2}$

Q4. (Newton-Cotes with Pythagorean formula: NC with PYT)

1. Create grids $\{x_i\}_{i=0}^{100}{}_{j=0}$ where $x_i = \frac{i}{100}$.
2. Create the vector f whose i-th element is
$$f_i = \sqrt{1 - x_i^2}$$
3. Following Simpson, create the 101 by 1 weight vector w as
$$w_0 = w_{101} = \frac{0.01}{3}$$
$$w_k = \begin{cases} \frac{4}{3} \times 0.01, & \text{if } k \text{ is even} \\ \frac{2}{3} \times 0.01, & \text{if } k \text{ is odd} \end{cases}$$
4. Approximate $\frac{\pi}{4}$ as $w'f$

Here are the approximated π's and the relevant absolute errors to the true π.

Table 1: Approximated π

Method	Approx. π	Abs. Error
q-MC with DT	3.1513	0.0304
NC with DT	3.1425	0.0009
q-MC with PYT	3.1387	0.0104
NC with PYT	3.1416	0.0005

Q5.

I randomize Quasi Monte Carlo by doing so:

For each simulation,

1. Draw $\{x_i\}_i^N$ from `qnwequi` over $[0,1]$.
2. Draw $U_i \sim i.i.d.Unif(0,1)$, $i = 1, \ldots, 100$ and create $y_i = \mod(x_i + U_i, 1)$.
3. Conduct the quasi-Monte Carlo method with $\{y_i\}_i^N$.

Here are the Mean Squared errors for each methods with draws (grids) 1,000, 5,000, and 10,000, respectively (For Newton-Cotes, the reported values are squared error).

Table 2: Approximated π

Method	N=1,000	N=5,000	N=10,000
q-MC with DT	0.0016	0.0003	0.0002
NC with DT	1.127e-09	1.705e-11	8.551e-13
q-MC with PYT	0.0007	0.0001	8.477e-05
NC with PYT	2.109e-10	1.687e-12	2.109e-13

Overall, Newton-Cotes with Pythagorean outperforms all the other methods in terms of squared errors.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Homework #4 ECON 512                                    %
% Written by Joonkyo (Jay) Hong, 20 Oct 2018              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;

addpath('./CEtools/');   % First, add path CEtools %
N = 100;                 % # of draws or nodes

%% Questions 1 (Dart-throwing method with quasi-MC approach)

     [x1, ~]= qnwequi(N, [0 0], [1 1]);
     z   =  indic_fcn(x1(:,1),x1(:,2));

     pi1 =  4*mean(mean(z));
     error1 = abs(pi1 - pi);

%% Question 2 (Dart-throwing method with Newton-Coates approach)

     pi2 = 4*Int_indic([0 0],[1 1],N,N);
     error2 = abs(pi2 - pi);

%% Questions 3 (Pythagorean method with quasi-MC approach)

     [x3, ~]  = qnwequi(N,0,1);
     pi3 = 4*mean(sqrt(1-x3.^2));
     error3 = abs(pi3 - pi);

%% Questions 4 (Pythagorean method with Newton-Coates approach)

     pi4 = 4*Int_simp(@(x) sqrt(1-x.^2), 0, 1, N);
     error4 = abs(pi4 - pi);


 result_from_1_to_4 =  ...
  [" ",           "approximate pi", "absolute error";
   "q-MC with DT",     pi1,              error1      ;
   "NC with DT"  ,     pi2,              error2      ;
   "q-MC with PYT",    pi3,              error3      ;
   "NC with PYT" ,     pi4,              error4      ];
%% Question 5 (Randomizing quasi-MC)

N       = [1000; 5000; 10000];
```

```
num_sim = 200;                   % # of simulations
store_array = zeros(2,length(N),num_sim);  % 3D-array that will store the Squared errors


for n=1:length(N)

    [x1, ~]  = qnwequi(N(n), [0 0], [1 1]);
    [x3, ~]  = qnwequi(N(n),0,1);

    for i=1:num_sim

        % Dart-Throwing with q-MC

          y1 = mod(x1+rand(N(n),2),1);    % Randomizing q-MC

          z   =  indic_fcn(y1(:,1),y1(:,2));
          pi1 =  4*mean(mean(z));
          store_array(1,n,i) = (pi1 - pi)^2;

          % Pythagorean with q-MC

          y3 = mod(x3+rand(N(n),1),1);     % Randomizing q-MC

          pi3 = 4*mean(sqrt(1-y3.^2));
          store_array(3,n,i) = (pi3 - pi)^2;

    end

          % Dart-Throwing with Newton-Cotes

          pi2 = 4*Int_indic([0 0],[1 1],N(n),N(n));
          store_array(2,n,:) = (pi2 - pi)^2;

          % Pythagorean with Newton-Cotes

          pi4 = 4*Int_simp(@(x) sqrt(1-x.^2), 0, 1, N(n));
          store_array(4,n,:) = (pi4-pi)^2;


end
```

```
results_mat = mean(store_array,3);   % Calculate mean over the simulations
results_mat= [[" ", "N=1,000", "N=5,000", "N=10,000"];
 ["q-MC with DT"; "NC with DT"; "q-MC with PYT"; "q-MC with PYT"], (results_mat)];

disp(" ");
disp(" ");
disp("Question 1~4. Approximated pi and abs error: N=100");
disp(result_from_1_to_4);

disp(" ");
disp("Question 5. Mean Squared Errors");
disp(results_mat);
```