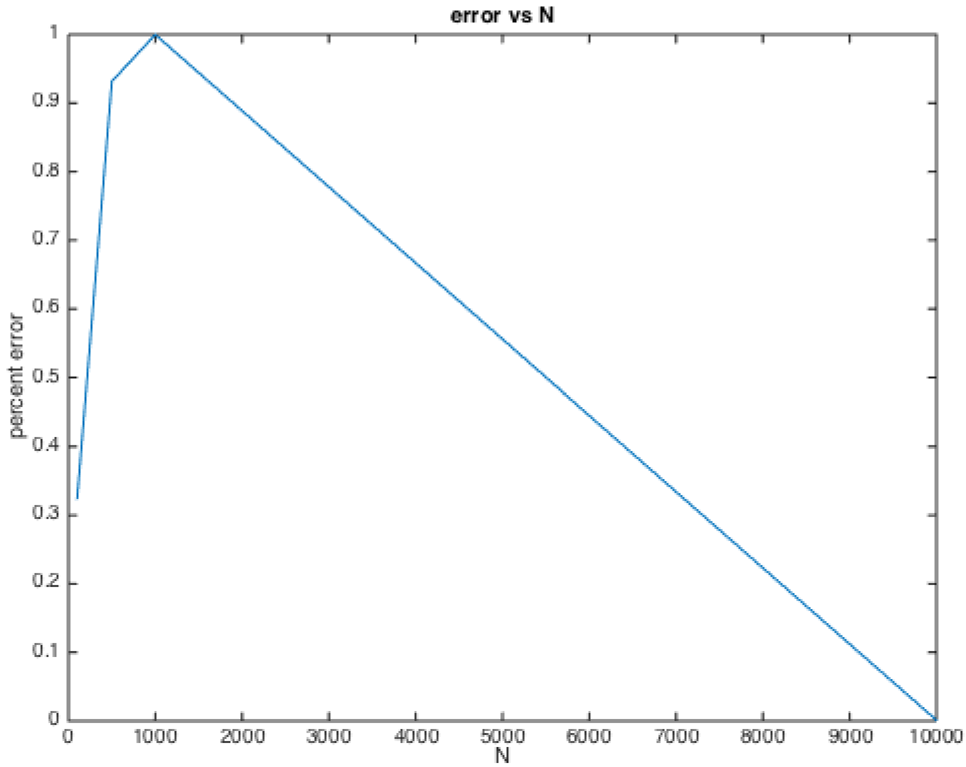
```
% Alexander Lerma
% Math 485

% 2. Monte Carlo (MC) Simulations. Implement (in Matlab) the Monte Carlo
% technique for computing prices of a generic European option, under the
% Black-Scholes-Merton model. Attach a printed version of the code to the
% report.

% Throughout this problem fix the model parameters
% r=0.02,?=0.25,and S0 =100.

% Step 1. Consider a given option V , with the
% payoff being an explicit function of ST (for simplicity). Use the scheme
% we discussed in class to simulate one path of the Geometric Brownian
% motion.

% Step 2. Simulate many paths, e.g., N = 10, 000. Generally
% speaking, the order of error is O(1/N ). Use a small enough time step
% size (e.g., ?t = 0.01).

% Step 3. For each path, compute the payoff vn of
% the option V , and store it in a vector.

% Step 4. Compute the average of vn? s: V0 = 1 (v1 + · · · + vN ) . N In view of
% MC method, the number V0 is an approximation of the true value of the
% price V0. The more path you simulate, and smaller time step you take, the
% closer to the true price you get.


% (a) Using this program, compute the
% price of the call option K = 100, T = 1, for different number of
% simulations N = 100, 500, 1000, 5000, and 10000. Also compute the price
% of this call option by using the explicit Black-Scholes formula. Compare
% the obtained prices, in particular, discuss how the number of simulations
% affects the error.


% fixed variables
r = 0.02;
sigma = 0.25;
s0 = 100;


K = 100;
T = 1;
N = [100, 500, 1000, 10000];

disp('Price of European call using explicit B-S formula')
[bs_call] = black_scholes(s0, T, K, r, sigma);
bs_call

errors = NaN([1, length(N)]);
```

```
mc_results = NaN([1, length(N)]);
i = 1;
for n = N
    fprintf('Price of European call using MC Simulation for n = %.0f', n)
    dt = 1 / n;
    mc_results(i) = monte_carlo(s0, dt, K, r, sigma, n);
    errors(i) = abs(mc_results(i) - bs_call) / bs_call;
    mc_results(i)
    i = i + 1;
end

figure('Name', 'error vs N');
plot(N, errors)
title('error vs N');
xlabel('N');
ylabel('percent error');




% (b) Consider the option VT = (180ST ? ST2 ? 7200)+ =
% max(0, 180ST ? ST2 ? 7200). Find the price of this option by MC
% simulations.

N = 10000;
payoff = @(st) max(0, (180 * st) - (st ^ 2) - 7200);

disp('Price of european call using MC simulations with explicit payoff')
monte_carlo(s0, T / N, K, r, sigma, N, payoff)
```

```
Price of European call using explicit B-S formula

bs_call =

   10.8706

Price of European call using MC Simulation for n = 100
ans =

    7.3424

Price of European call using MC Simulation for n = 500
ans =

    0.7415

Price of European call using MC Simulation for n = 1000
ans =

    0.0033

Price of European call using MC Simulation for n = 10000
ans =
```

```
   10.8589

Price of european call using MC simulations with explicit payoff

ans =

  840.2041
```




*Published with MATLAB® R2014b*