```
% NOTE: Use the batched version(q1d_batched.m) if this code is killed
 on execution. Since
% the samples are drawn simultaneously, memory used may be a bit high.

rng(4);
% number of times to repeat experiment for a given M
k = 1000;
%lowest value of M
low = 1;
%highest value of M
high = 500000;

% using binary search to find the correct value of M
% although the probability that we will get wont be sorted exactly but
% would be increasing in general. So, this would give a good estimate
 of M
while low<high-1
    %middle value for binary search
   m = floor((low+high)/2);
   % sampling the points. There are k experiments, and for each
 experiment
   % m points. SO, the sample is of size k*m*2.
   total_sample = single(2*rand(k,m,2)-1);
   % for each experiment, calculating how many samples are within a
   % radius of 1 from origin
   % y = sample(:,:,1).^2 + sample(:,:,2).^2 finds the distance of
 point
   % from origin and then we compare it with 1. It stores a 1 if the
 point
   % lies inside the circle otherwise 0.
   % for each experiment, calculate the estimate of pi. sum(y<=1,2) is
 total
   % number of points inside circle. So, probability is sum(y<=1,2)/m.
   pi_e = single(4*sum(single(total_sample(:,:,1).^2 +
 total_sample(:,:,2).^2 <= 1),2)/m);
   % probability that the estimate is in range of 0.01 from true value
   p = sum(abs(pi_e-pi) <= 0.01,"all")/k;
   % applying the recursive step for binary search
   if p==0.95
       break;
   else
       if p > 0.95
        high = m;
       else
        low = m;
       end
   end
   % applying the recursive step for binary search
   disp(string(m)+": "+string(p))
   disp("--------------------")
end
```

```
% if no p=0.95, the algorithm may return M for which p is just lower
 or
% just higher, So ensuring p is higher
if p<0.95
    m=m+1;
end
disp("found value of m: " + num2str(m));
```

```
250000: 0.999
--------------------
125000: 0.962
--------------------
62500: 0.868
--------------------
93750: 0.913
--------------------
109375: 0.961
--------------------
101562: 0.942
--------------------
105468: 0.953
--------------------
103515: 0.944
--------------------
104491: 0.955
--------------------
104003: 0.961
--------------------
103759: 0.938
--------------------
103881: 0.945
--------------------
103942: 0.951
--------------------
103911: 0.945
--------------------
103926: 0.948
--------------------
103934: 0.952
--------------------
103930: 0.952
--------------------
103928: 0.954
--------------------
103927: 0.959
--------------------
found value of m: 103927
```

Published with MATLAB® R2020a