```
% AmirHosein Sadeghimanesh
% 2020 February, modified 2021 August
%
% This script is to check how much time Matlab needs to solve the system of
% equestions in Figure 8b of the paper after fixing all values of the
% parameters other than k7 and k8 for 1000 sample points (k7, k8) from the
% region B = [(0, 0), (5, 5)].
%
tic
%
% The symbolic variables of this file.
%
syms t T1 T2 % T1 and T2 are k7 and k8 of the paper.
%
% Read the documentation for Matlab's vpasolve command. In univariate case
% it ignores the assumptions added by assume(). Therefore instead of using
% asssume(t>0), we will add the interval [0,Inf] as an option to vpasolve.
%
% The system of equations of interest is equivalent with the following
% single univariate parametric equation. See the paper "Kac-Rice formulas
% and the number of solutions of parametrized systems of polynomial
% equations, Elisenda Feliu, AmirHosein Sadeghimanesh, 2020".
%
eqn = (2518322.5)*(t^3)+((366450)*T1-(2518322.5)*T2+(63502.1205))*(t^2)+((537142.41)↙
*T1-(63502.1205)*T2+(26857.1205))*t-(26857.1205)*T2;
%
NN = 1000; % Number of the sample points.
B = [0, 5; 0, 5]; % The parameter region.
%
% Preallocation of four arrays to save the parameter points with 0, 1, 2
% and 3 positive real solutions in them. When all 1000 points got checked,
% we shrink the preallocated arrays to have proper sizes.
%
L0 = zeros(NN, 2); % List of the points with no solution.
L1 = zeros(NN, 2); % List of the points with 1 solution.
L2 = zeros(NN, 2); % List of the points with 2 solutions.
L3 = zeros(NN, 2); % List of the points with 3 solutions. We know that 3 is the upper↙
bound, so we do not need more lists.
%
% To track the length of L_i lists, we introduce the following counters.
%
idx0 = 0;
idx1 = 0;
idx2 = 0;
idx3 = 0;
%
for idx = 1:NN
    AA = sampleo(B(1, 1), B(1, 2)); % Generating a random sample for k7 from the↙
uniform distribution on the interval (0,5).
    BB = sampleo(B(2, 1), B(2, 2)); % Generating a random sample for k8 from the↙
uniform distribution on the interval (0,5).
    Equations = subs(eqn, [T1, T2], [AA, BB]); % Substituting the values of k3 and k8↙
in the equations.
    tSol = vpasolve(Equations, t, [0, Inf]); % Solving the system of equations↙
numerically. It is possible to use 'solve', but we do not need that here, 'vpasolve'↙
```

```
is enough for our purpose and can be faster than finding exact solutions in some↙
cases.
    solutions_number = length(tSol); % Number of solutions.
    switch(solutions_number)
        case 1
            idx1 = idx1+1;
            L1(idx1, :) = [AA, BB];
        case 3
            idx3 = idx3+1;
            L3(idx3, :) = [AA, BB];
        case 2
            idx2 = idx2+1;
            L2(idx2, :) = [AA, BB];
        otherwise
            idx0 = idx0+1;
            L0(idx0, :) = [AA, BB];
    end
end
%
% Shrinking L_i lists to their real lengths.
%
L1 = L1(1:idx1, :);
L2 = L2(1:idx2, :);
L3 = L3(1:idx3, :);
L0 = L0(1:idx0, :);
%
% Here all the computations are completed, so we consider this location to
% stop the timing.
%
toc
%
% Writing L1 and L3 in two txt files.
%
folder = 'C:\Home\PSS\Codes\Section_5_2'; % replace this directory to the directory of↙
the folder you are using.
baseFileName = 'Section_5_2_SamplingRepresentation_L1_output.txt';
fullFileName = fullfile(folder, baseFileName);
L1_file = fopen(fullFileName, 'w');
fprintf(L1_file, '%d points of %d points have %d solutions. These %d points are listed↙
in below.\n\n', length(L1), NN, 1, length(L1));
for idx = 1:size(L1, 1)
    fprintf(L1_file, '%f,%f\n', L1(idx, 1), L1(idx, 2));
end
fclose(L1_file);
baseFileName = 'Section_5_2_SamplingRepresentation_L3_output.txt';
fullFileName = fullfile(folder, baseFileName);
L3_file = fopen(fullFileName, 'w');
fprintf(L3_file, '%d points of %d points have %d solutions. These %d points are listed↙
in below.\n\n', length(L3), NN, 3, length(L3));
for idx = 1:size(L3, 1)
    fprintf(L3_file, '%f,%f\n', L3(idx, 1), L3(idx, 2));
end
fclose(L3_file);
%
% Function to generate a random real number from a given interval.
```

```
%
function sampleo = sampleo(a, b)
    sampleo = a + (b-a) * rand;
end
%
% End of the file.
```