Table of Contents

E7 Lab 12 Solutions

Spring 2016

```
format compact
format short
clear all
clc
close all
```

Question 1.1

```
type myEulerApprox
```

```
function [T_euler,t_euler] = myEulerApprox(delta_t,T_0,R,C)
% Inputs:
% delta_t: scalar, time-step of the Euler method (hours)
% T_0: scalar, initial temperature in the building (°F)
% R: scalar, thermal resistance of 1 sq.ft of wall (°F.hr/BTU)
% C: scalar, thermal capacitance of 1 sq.ft of wall (BTU/°F)
%
% Outputs:
% T_euler: vector of size (24/delta_t+1)x1, temperature in the building as
% estimated by Euler's method (°F)
% t_euler: vector of size (24/delta_t+1)x1, time steps at which the
% temperature is evaluated (hours)
%
% The function evaluates the temperature inside a building using Euler's
% method to approximate the solution of the governing ODE. A Gaussian
% distribution of the external temperature is used.
t_euler=(0:delta_t:24)';

% define the external temperature distribution
T_min=60;
Amp=10;
```

```
sigma=1;
mu=12;
Ta=(Amp/(sigma*sqrt(2*pi)))*exp(-((t_euler-mu).^2)/(2*sigma^2))+T_min;

% implement Euler's method
T_euler=zeros(size(t_euler));
T_euler(1)=T_0;
for i=1:(length(T_euler)-1)
    T_euler(i+1)=T_euler(i)+(delta_t/(R*C))*(Ta(i)-T_euler(i));
end
end
```

Published Test Case

```
[T_euler, t_euler] = myEulerApprox(1, 70, 2, 10)
```

```
T_euler =
   70.0000
   69.5000
   69.0250
   68.5738
   68.1451
   67.7378
   67.3509
   66.9834
   66.6342
   66.3026
   65.9896
   65.7172
   65.5523
   65.4741
   65.3214
   65.0823
   64.8304
   64.5890
   64.3595
   64.1416
   63.9345
   63.7378
   63.5509
   63.3733
   63.2047
t_euler =
     0
     1
     2
     3
     4
     5
     6
     7
     8
     9
    10
    11
    12
    13
    14
    15
    16
    17
```

```
    18
    19
    20
    21
    22
    23
    24
```

Additional Test Case

```
[T_euler2, t_euler2] = myEulerApprox(2, 65, 2, 10)
```

```
T_euler2 =
   65.0000
   64.5000
   64.0500
   63.6450
   63.2805
   62.9526
   62.7113
   62.8391
   62.6092
   62.3484
   62.1136
   61.9022
   61.7120
t_euler2 =
     0
     2
     4
     6
     8
    10
    12
    14
    16
    18
    20
    22
    24
```

Question 1.2

```
type myODESolver
```

```
function [T,t,range] = myODESolver(delta_t,T_0,R,C)
% Inputs:
% delta_t: scalar, time-step for solving the ODE (hours)
% T_0: scalar, initial temperature in the building (°F)
% R: scalar, thermal resistance of 1 sq.ft of wall (°F.hr/BTU)
% C: scalar, thermal capacitance of 1 sq.ft of wall (BTU/°F)
%
% Outputs:
% T: vector of size (24/delta_t+1)x1, temperature in the building as
% estimated by ode45 (°F)
% t_euler: vector of size (24/delta_t+1)x1, time steps at which the
```

```
% temperature is evaluated (hours)
% range: vector of size 1x2, containing the minimum and maximum values
% reached by T during the time period t (°F)
%
% The function evaluates the temperature inside a building using MATLAB's
% ODE solver ode45. A Gaussian distribution of the external temperature
% is used.
[t,T]=ode45(@(t,y) (1/(R*C))*(myTa(t)-y),0:delta_t:24,T_0);
range=[min(T),max(T)];
end

function [Ta]=myTa(t)
% This sub-function calculates the external air temperature distribution,
% based on a Gaussian model
T_min=60;
Amp=10;
sigma=1;
mu=12;
Ta=(Amp/(sigma*sqrt(2*pi)))*exp(-((t-mu).^2)/(2*sigma^2))+T_min;
end
```

Published Test Case

```
[T, t, range] = myODESolver(2, 65, 2, 10);
T'
t'
range
```

```
ans =
  Columns 1 through 8
   65.0000   64.5242   64.0937   63.7041   63.3516   63.0426   62.9856   62.9252
  Columns 9 through 13
   62.6571   62.4044   62.1756   61.9686   61.7812
ans =
     0     2     4     6     8    10    12    14    16    18    20    22    24
range =
   61.7812   65.0000
```

Additional Test Case

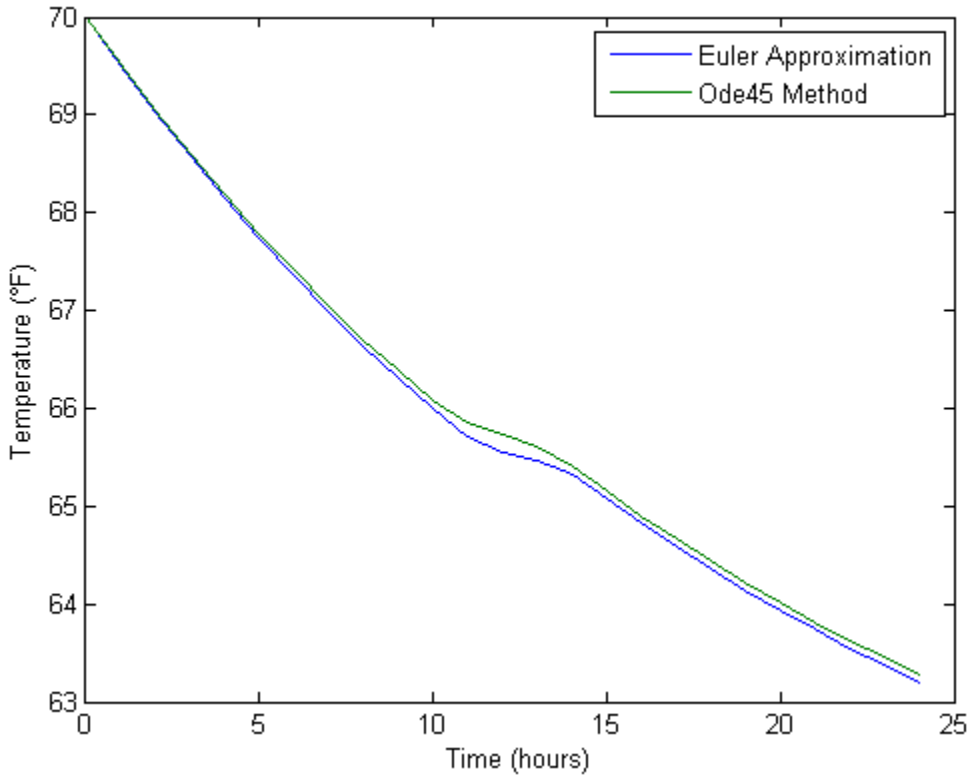
```
[T2, t2, range2] = myODESolver(1, 70, 2, 10);
T2'
t2'
range2
```

```
ans =
  Columns 1 through 8
   70.0000   69.5123   69.0484   68.6071   68.1873   67.7880   67.4082   67.0469
  Columns 9 through 16
   66.7032   66.3772   66.0765   65.8480   65.7265   65.6142   65.4055   65.1528
  Columns 17 through 24
   64.9021   64.6630   64.4356   64.2193   64.0135   63.8178   63.6316   63.4545
  Column 25
   63.2860
ans =
  Columns 1 through 14
     0     1     2     3     4     5     6     7     8     9    10    11    12
```

```
  Columns 15 through 25
    14    15    16    17    18    19    20    21    22    23    24
range2 =
   63.2860   70.0000
```

Comparison Figure:

```
figure;
plot(t_euler, T_euler, t2, T2);
legend('Euler Approximation', 'Ode45 Method');
ylabel(sprintf('Temperature (%cF)', char(176)));
xlabel('Time (hours)');
```



Question 2.1

```
type myAirQualityNoEmissions
```

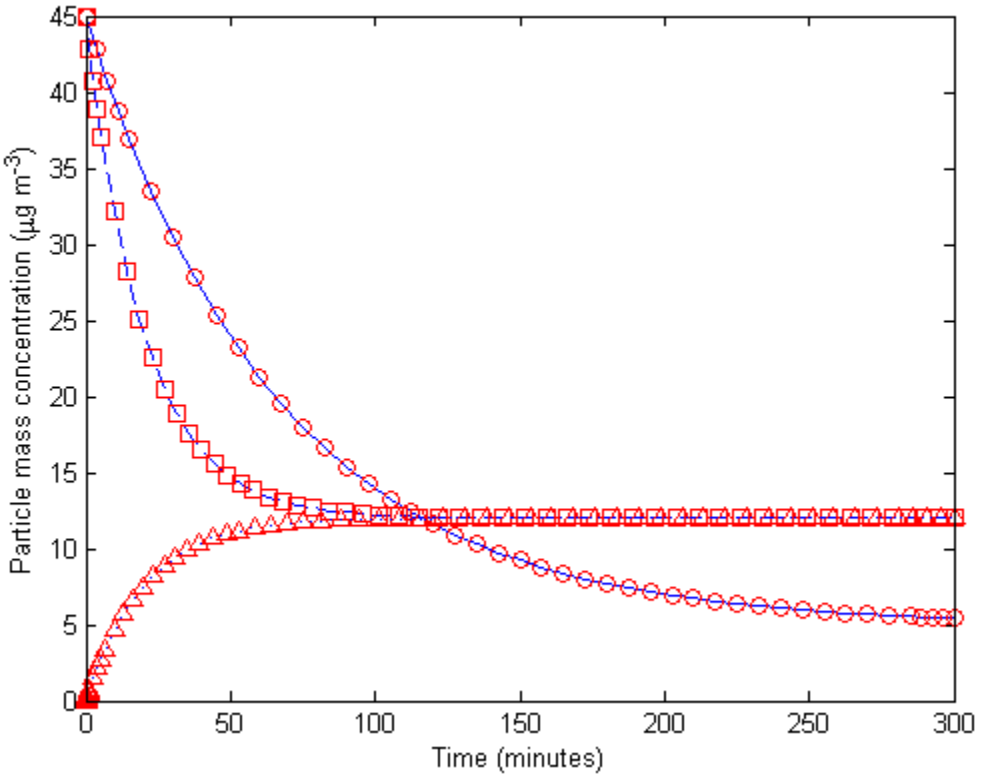
```
function [t_out,C_ode45,C_analytic,ME,RMSE] = myAirQualityNoEmissions(V,Q,beta,C_v
% Inputs:
% V: scalar, volume of the room (m3)
% Q: scalar, flow rate (m3/min)
% beta: scalar, surface deposition rate (/min)
% C_vent: scalar, concentration of particles brought by ventilation (µg/m3)
% t_span: either a 1x2 vector, initial and final times of the time period
% considered, or a 1xn vector containing all the time steps used
```

```
% to compute evaluate the final concentration (min)
% C0: scalar, initial concentration of particles in the room (µg/m3)
%
% Outputs:
% t_out: column vector, time steps used by ode45 (min)
% C_ode45: column vector having the same size as t_out, solution returned
% by ode45 (µg/m3)
% C_analytic: column vector having the same size as t_out, analytical
% solution (µg/m3)
% ME: scalar, mean error beteen the analytical solution and the numerical
% solution
% RMSE: scalar, root mean square error between the analytical solution and
% the numerical solution
%
% The function calculates the mass particle concentration in a room after a
% given period of time (without internal emissions), using MATLAB's ODE
% solver ode45 and an analytical expression of the solution
tau=V/(Q+beta*V);
[t_out,C_ode45]=ode45(@(t,y) (Q/V)*(C_vent-y)-beta*y,t_span,C0);
C_analytic=Q*C_vent*tau/V+(C0-Q*C_vent*tau/V)*exp(-(t_out-t_span(1))/tau);
ME=(1/length(t_out))*sum(C_ode45-C_analytic);
RMSE=sqrt((1/length(t_out))*sum((C_analytic-C_ode45).^2));
end
```

Published Test Cases

```
V = 50; Q = 0.25; C_vent = 15; beta = 0.01;
t_span = [0, 300]; C0 = 45;
[t_out, C_ode45, C_analytic, ME, RMSE] = ...
    myAirQualityNoEmissions(V,Q,beta,C_vent,t_span,C0);

% Check the numerical values in your outputs against these:
t_out_1_5 = t_out(1:5)'
C_ode45_1_5 = C_ode45(1:5)'
C_analytic_1_5 = C_analytic(1:5)'
ME
RMSE

plot(t_out, C_analytic, 'b-', t_out, C_ode45, 'ro')
xlabel('Time (minutes)')
ylabel('Particle mass concentration ({\mug} m^{-3})')

% Change Q and recalculate
Q = 2
[t_out, C_ode45, C_analytic, ME, RMSE] = ...
    myAirQualityNoEmissions(V,Q,beta,C_vent,t_span,C0);

% Check the numerical values in your outputs against these:
t_out_1_5 = t_out(1:5)'
C_ode45_1_5 = C_ode45(1:5)'
C_analytic_1_5 = C_analytic(1:5)'
ME
RMSE

hold on
plot(t_out, C_analytic, 'b--', t_out, C_ode45, 'rs')

% Set the initial particle concentration to 0
C0 = 0
[t_out, C_ode45, C_analytic, ME, RMSE] = ...
    myAirQualityNoEmissions(V,Q,beta,C_vent,t_span,C0);
```

```
% Check the numerical values in your outputs against these:
t_out_1_5 = t_out(1:5)'
C_ode45_1_5 = C_ode45(1:5)'
C_analytic_1_5 = C_analytic(1:5)'
ME
RMSE

hold on
plot(t_out, C_analytic, 'b:', t_out, C_ode45, 'r^')


t_out_1_5 =
         0    3.7678    7.5357   11.3035   15.0713
C_ode45_1_5 =
   45.0000   42.8020   40.7248   38.7617   36.9065
C_analytic_1_5 =
   45.0000   42.8020   40.7248   38.7617   36.9065
ME =
 -2.6665e-006
RMSE =
  9.9684e-005
Q =
     2
t_out_1_5 =
         0    1.3701    2.7402    4.1104    5.4805
C_ode45_1_5 =
   45.0000   42.8150   40.7746   38.8694   37.0904
C_analytic_1_5 =
   45.0000   42.8150   40.7747   38.8694   37.0904
ME =
 -6.5186e-005
RMSE =
    0.0023
C0 =
     0
t_out_1_5 =
  1.0e-003 *
         0    0.0837    0.1675    0.2512    0.3349
C_ode45_1_5 =
  1.0e-003 *
         0    0.0502    0.1005    0.1507    0.2009
C_analytic_1_5 =
  1.0e-003 *
         0    0.0502    0.1005    0.1507    0.2009
ME =
  2.6542e-005
RMSE =
  3.4706e-004
```



Additional Test Case

```
V = 50; Q = 1; beta = 0.0075; C_vent = 10; t_span = [10, 300]; C0 = 5;

[t_out, C_ode45, C_analytic, ME, RMSE] = ...
    myAirQualityNoEmissions(V,Q,beta,C_vent,t_span,C0);

t_out'
C_ode45'
C_analytic'
ME
RMSE


ans =
  Columns 1 through 8
   10.0000   14.0190   18.0380   22.0571   26.0761   33.3261   40.5761   47.8261
  Columns 9 through 16
   55.0761   62.3261   69.5761   76.8261   84.0761   91.3261   98.5761  105.8261
  Columns 17 through 24
  113.0761  120.3261  127.5761  134.8261  142.0761  149.3261  156.5761  163.8261
  Columns 25 through 32
  171.0761  178.3261  185.5761  192.8261  200.0761  207.3261  214.5761  221.8261
  Columns 33 through 40
  229.0761  236.3261  243.5761  250.8261  258.0761  265.3261  272.5761  279.8261
  Columns 41 through 45
  287.0761  290.3071  293.5380  296.7690  300.0000
ans =
  Columns 1 through 8
    5.0000    5.2378    5.4508    5.6414    5.8121    6.0764    6.2928    6.4697
```

```
  Columns 9 through 16
    6.6146    6.7337    6.8312    6.9109    6.9762    7.0299    7.0738    7.1097
  Columns 17 through 24
    7.1391    7.1633    7.1831    7.1993    7.2125    7.2234    7.2323    7.2396
  Columns 25 through 32
    7.2456    7.2505    7.2545    7.2578    7.2605    7.2627    7.2645    7.2660
  Columns 33 through 40
    7.2672    7.2682    7.2690    7.2697    7.2702    7.2707    7.2711    7.2714
  Columns 41 through 45
    7.2716    7.2717    7.2718    7.2719    7.2719
ans =
  Columns 1 through 8
    5.0000    5.2378    5.4507    5.6414    5.8121    6.0761    6.2924    6.4696
  Columns 9 through 16
    6.6148    6.7337    6.8311    6.9110    6.9763    7.0299    7.0738    7.1098
  Columns 17 through 24
    7.1392    7.1634    7.1831    7.1993    7.2126    7.2235    7.2324    7.2397
  Columns 25 through 32
    7.2456    7.2505    7.2545    7.2578    7.2605    7.2627    7.2645    7.2660
  Columns 33 through 40
    7.2672    7.2682    7.2690    7.2697    7.2703    7.2707    7.2711    7.2714
  Columns 41 through 45
    7.2716    7.2717    7.2718    7.2719    7.2719
ME =
  4.2548e-007
RMSE =
  9.2163e-005
```

Question 2.2

```
type myAirQuality
```

```
function [t_out,C_ode45] = myAirQuality(V,Q,beta,C_vent,t_span,C0,t_start,t_end,E_
% Inputs:
% V: scalar, volume of the room (m3)
% Q: scalar, flow rate (m3/min)
% beta: scalar, surface deposition rate (/min)
% C_vent: scalar, concentration of particles brought by ventilation (µg/m3)
% t_span: either a 1x2 vector, initial and final times of the time period
% considered, or a 1xn vector containing all the time steps used
% to compute evaluate the final concentration (min)
% C0: scalar, initial concentration of particles in the room (µg/m3)
% t_start: scalar, internal emission event starting time (min)
% t_end: scalar, internal emission event ending time (min)
% E_value: scalar, internal emission event magnitude (µg/min)
%
% Outputs:
% t_out: column vector, time steps used by ode45 (min)
% C_ode45: column vector having the same size as t_out, solution returned
% by ode45 (µg/m3)
%
% The function calculates the mass particle concentration in a room after a
% given period of time (witho internal emissions), using MATLAB's ODE
% solver ode45
[t_out,C_ode45]=ode45(@(t,y) (Q/V)*(C_vent-y)-beta*y+myE(t,t_start,t_end,E_value)/
end

function [E]=myE(t,t_start,t_end,E_value)
% This sub-function calculates the indoor emissions distribution,
% assumed to follow a top-hat behavior
```

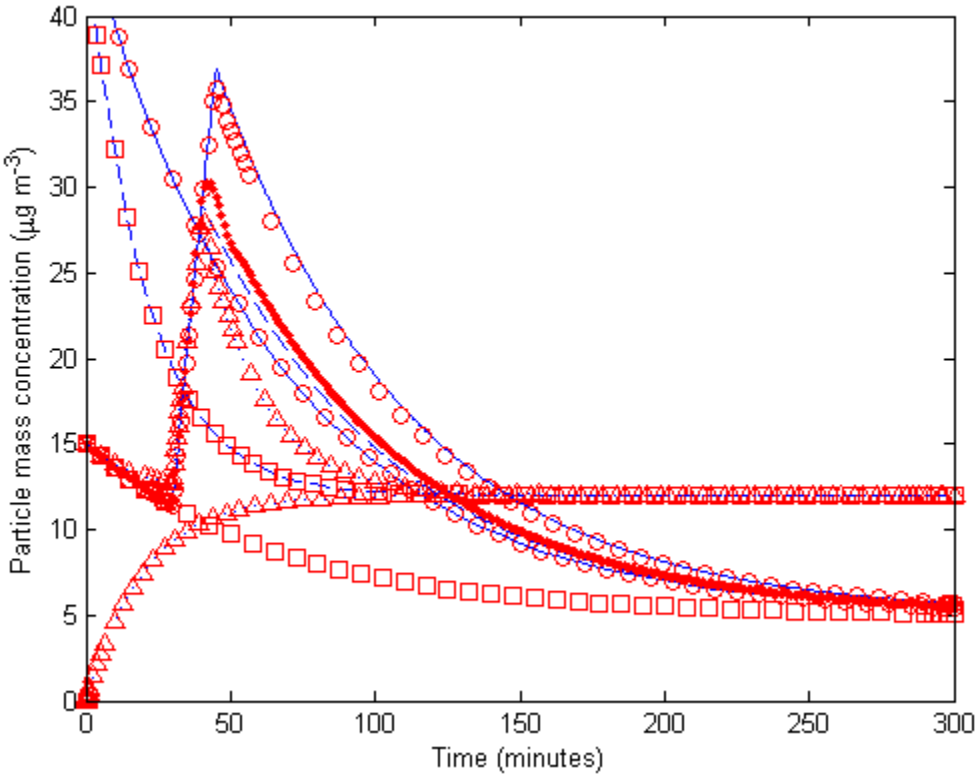
```
if t>t_start && t<=t_end
    E=E_value;
else E=0;
end
end
```

Published Test Cases

```
V = 50; Q = 0.25; C_vent = 15; beta = 0.01; t_span = [0, 300];
C0 = C_vent; E_start = 30; E_end = 45; E_value = 100;

[t_out, C_ode45] = myAirQuality(V, Q, beta, C_vent, t_span, ...
    C0, E_start, E_end, E_value);

t = linspace(t_span(1), t_span(2), 1000)';
C_analytic = myAirQualityAnalytic(V, Q, beta, C_vent, t_span(1), ...
    C0, E_start, E_end, E_value, t);

% Check the numerical values in your outputs against these:
t_out_10_15 = t_out(10:15)'
C_ode45_10_15 = C_ode45(10:15)'

plot(t, C_analytic, 'b-', t_out, C_ode45, 'ro')
ylim([0, 40])
xlabel('Time (minutes)')
ylabel('Particle mass concentration ({\mu}g m^{-3})')


% Decrease the length of emissions
E_end = 40
[t_out, C_ode45] = myAirQuality(V, Q, beta, C_vent, t_span, ...
    C0, E_start, E_end, E_value);
C_analytic = myAirQualityAnalytic(V, Q, beta, C_vent, t_span(1), ...
    C0, E_start, E_end, E_value, t);

% Check the numerical values in your outputs against these:
t_out_10_15 = t_out(10:15)'
C_ode45_10_15 = C_ode45(10:15)'

hold on
plot(t, C_analytic, 'b--', t_out, C_ode45, 'rs')

% Set the time values to evaluate at
t_ode45 = linspace(t_span(1), t_span(2), 301)';
[t_out, C_ode45] = myAirQuality(V, Q, beta, C_vent, t_ode45, ...
    C0, E_start, E_end, E_value);
plot(t_out, C_ode45, 'r.')

% Check the numerical values in your outputs against these:
t_out_spec_40_45 = t_out(40:45)'
C_ode45_spec_40_45 = C_ode45(40:45)'

% Change Q
Q = 2
[t_out, C_ode45] = myAirQuality(V, Q, beta, C_vent, t_span, ...
    C0, E_start, E_end, E_value);
C_analytic = myAirQualityAnalytic(V, Q, beta, C_vent, t_span(1), ...
    C0, E_start, E_end, E_value, t);

% Check the numerical values in your outputs against these:
t_out_10_15 = t_out(10:15)'
C_ode45_10_15 = C_ode45(10:15)'
```

```
plot(t, C_analytic, 'b:', t_out, C_ode45, 'r^')
```

```
t_out_10_15 =
   26.7506   27.7471   28.7437   29.7403   30.7369   31.7335
C_ode45_10_15 =
   11.6948   11.5954   11.4976   11.4012   12.3755   14.3667
E_end =
    40
t_out_10_15 =
   57.5951   65.0951   72.5951   80.0951   87.5951   95.0951
C_ode45_10_15 =
    9.2150    8.7665    8.3658    8.0077    7.6876    7.4016
t_out_spec_40_45 =
    39    40    41    42    43    44
C_ode45_spec_40_45 =
   27.7756   29.1628   29.9508   30.2649   30.2211   29.9219
Q =
     2
t_out_10_15 =
   28.7770   29.1623   29.5475   29.9327   30.3180   30.7032
C_ode45_10_15 =
   12.7125   12.6989   12.6856   12.6725   13.0723   13.8622
```



Additional Test Case

```
V = 50; Q = 1; beta = 0.0075; C_vent = 10; t_span = [10, 300]; C0 = 5;
E_start = 60; E_end = 90; E_value = 120;
```

```
[t_out, C_ode45] = myAirQuality(V, Q, beta, C_vent, t_span, ...
    C0, E_start, E_end, E_value);

t_out'
C_ode45'


ans =
  Columns 1 through 8
   10.0000   14.0190   18.0380   22.0571   26.0761   33.3261   40.5761   47.8261
  Columns 9 through 16
   55.0761   55.9090   56.7419   57.5748   58.4077   58.6950   58.9824   59.2697
  Columns 17 through 24
   59.5571   59.7866   60.0162   60.2457   60.4752   60.7048   60.9343   61.1638
  Columns 25 through 32
   61.3934   62.5411   63.6887   64.8364   65.9841   71.6091   77.2341   82.8591
  Columns 33 through 40
   88.4842   91.7664   95.0485   98.3307  101.6129  104.8951  108.1773  111.4595
  Columns 41 through 48
  114.7417  121.9917  129.2417  136.4917  143.7417  150.9917  158.2417  165.4917
  Columns 49 through 56
  172.7417  179.9917  187.2417  194.4917  201.7417  208.9917  216.2417  223.4917
  Columns 57 through 64
  230.7417  237.9917  245.2417  252.4917  259.7417  266.9917  274.2417  281.4917
  Columns 65 through 69
  288.7417  291.5563  294.3709  297.1854  300.0000
ans =
  Columns 1 through 8
    5.0000    5.2378    5.4508    5.6414    5.8121    6.0764    6.2928    6.4697
  Columns 9 through 16
    6.6146    6.6295    6.6441    6.6583    6.6722    6.6770    6.6816    6.6863
  Columns 17 through 24
    6.6909    6.6288    6.7144    7.1245    7.7130    8.2593    8.8023    9.3418
  Columns 25 through 32
    9.8780   12.5084   15.0572   17.5268   19.9196   30.6186   39.7829   47.6283
  Columns 33 through 40
   54.3491   53.7471   49.2609   45.1263   42.0753   39.0715   36.3270   33.8195
  Columns 41 through 48
   31.5284   27.1391   23.5447   20.6074   18.2014   16.2238   14.6043   13.2808
  Columns 49 through 56
   12.1968   11.3057   10.5760    9.9797    9.4913    9.0898    8.7611    8.4924
  Columns 57 through 64
    8.2723    8.0915    7.9433    7.8223    7.7231    7.6416    7.5749    7.5203
  Columns 65 through 69
    7.4757    7.4605    7.4466    7.4336    7.4216
```

Published with MATLAB® 7.10