Robert Heeter
BIOE 391 Numerical Methods – Due 24 January 2022

Problem Set 1 Solutions

1.

a. Output

Time (s), velocities (m/s) to t = 12s; step size = 1s

0	0
1.0000	9.8100
2.0000	19.2667
3.0000	27.7140
4.0000	34.7044
5.0000	40.0930
6.0000	44.0019
7.0000	46.7041
8.0000	48.5065
9.0000	49.6789
10.0000	50.4287
11.0000	50.9030
12.0000	51.2008

b. Output

Time (s), velocities (m/s) to t = 12s; step size = 0.5s

0	0
0.5000	4.9050
1.0000	9.7658
1.5000	14.4958
2.0000	19.0151
2.5000	23.2564
3.0000	27.1686
3.5000	30.7188
4.0000	33.8917
4.5000	36.6883
5.0000	39.1226
5.5000	41.2182
6.0000	43.0047
6.5000	44.5151
7.0000	45.7828
7.5000	46.8404
8.0000	47.7182
8.5000	48.4436
9.0000	49.0410
9.5000	49.5315
10.0000	49.9333
10.5000	50.2617
11.0000	50.5297
11.5000	50.7481
12.0000	50.9259

The errors of the calculation decreased for a smaller step size, using the analytical solution in Example 1.1. For t = 12s, the Euler approximation with a step size of 0.5s was 50.9259 m/s, the approximation with a step size of 1s was 51.2008, and the exact value was 50.6175, Interestingly, the rougher approximation approached the asymptote (terminal velocity) of 51.6938 m/s faster.

2.

a. $V = \frac{mg}{c'}\left(1 - e^{-c't/m}\right)$, given the initial condition of the jumper being at rest, where V is velocity in m/s; see my handwritten work at the end of the document.

b. Output

Time (s), velocities (m/s) to t = 40s; step size = 2s

0	0
2.0000	19.6200
4.0000	32.6136
6.0000	41.2187
8.0000	46.9176
10.0000	50.6917
12.0000	53.1911
14.0000	54.8464
16.0000	55.9427
18.0000	56.6687
20.0000	57.1495
22.0000	57.4679
24.0000	57.6788
26.0000	57.8184
28.0000	57.9109
30.0000	57.9722
32.0000	58.0127
34.0000	58.0396
36.0000	58.0574
38.0000	58.0692
40.0000	58.0770

3. Output

Time (days), depths (m) to t = 10d; step size = 0.5d

0	0
0.5000	-0.1800
1.0000	-0.2359
1.5000	-0.0335
2.0000	0.3238
2.5000	0.5903
3.0000	0.6037
3.5000	0.4344
4.0000	0.3209
4.5000	0.4502
5.0000	0.7862
5.5000	1.1027
6.0000	1.1915
6.5000	1.0537
7.0000	0.8987
7.5000	0.9517

8.0000	1.2469
8.5000	1.5954
9.0000	1.7597
9.5000	1.6714
10.0000	1.4945

4. **+1.3 liters of water** must be drunk per day to maintain a steady state in the body; see my handwritten work at the end of the document.
5. **Output**
   Time (min), temperature (°C) to t = 20m; step size = 2m

0	70.0000
2.0000	68.1000
4.0000	66.2722
6.0000	64.5139
8.0000	62.8223
10.0000	61.1951
12.0000	59.6297
14.0000	58.1237
16.0000	56.6750
18.0000	55.2814
20.0000	53.9407

6.
   a. **Output**
   Time (hrs), temperature (°C) to t = 5hr; step size = 0.5hr

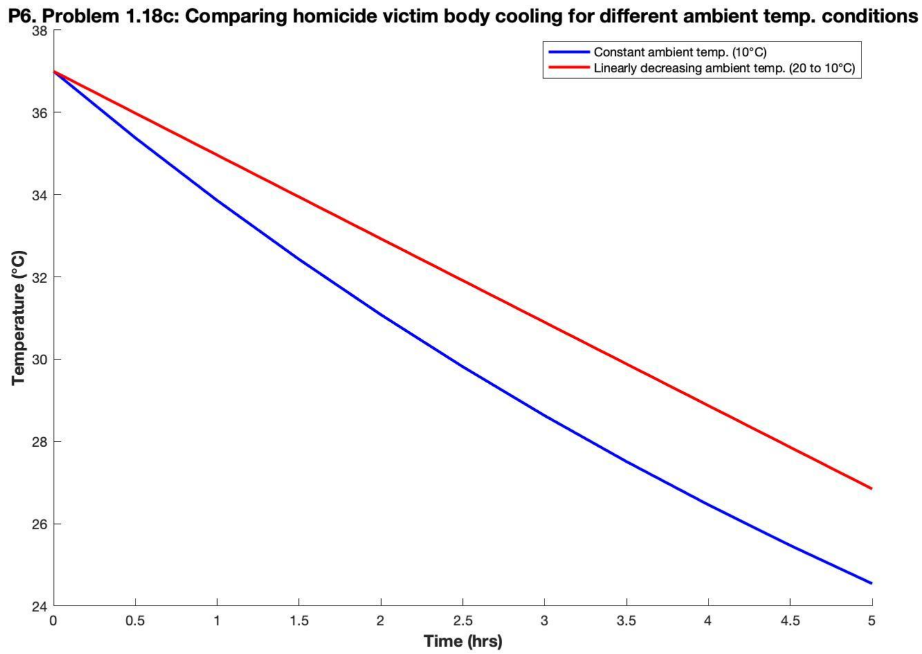
0	37.0000
0.5000	35.3800
1.0000	33.8572
1.5000	32.4258
2.0000	31.0802
2.5000	29.8154
3.0000	28.6265
3.5000	27.5089
4.0000	26.4584
4.5000	25.4709
5.0000	24.5426

   b. **Output**
   Time (hrs), temperature (°C) to t = 5hr; step size = 0.5hr

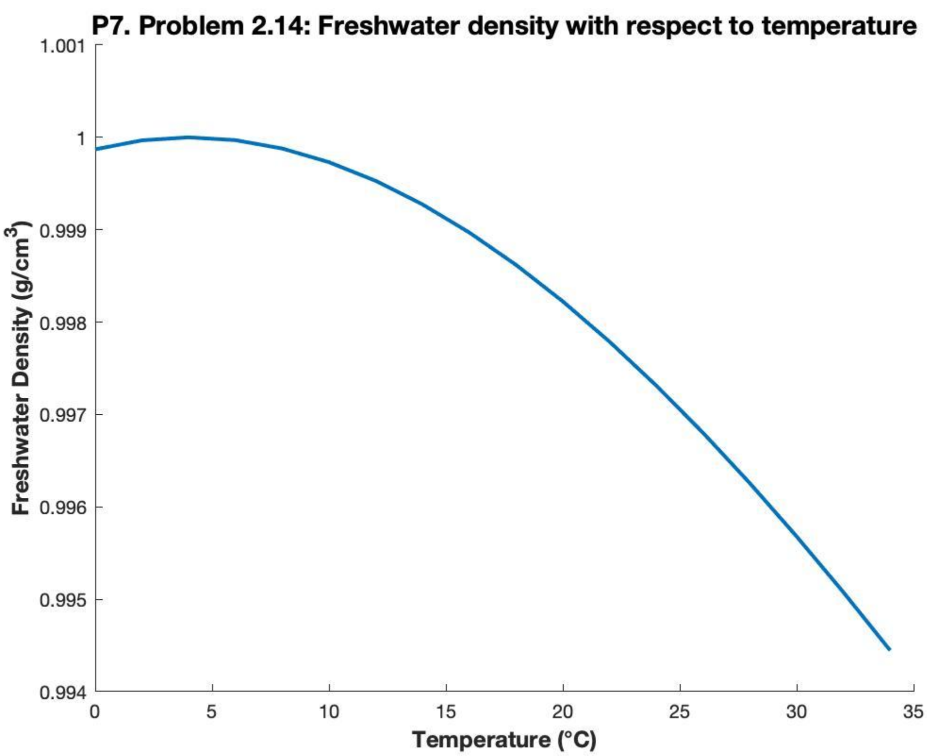
0	37.0000
0.5000	35.9800
1.0000	34.9612
1.5000	33.9435
2.0000	32.9269
2.5000	31.9113
3.0000	30.8966
3.5000	29.8828
4.0000	28.8699
4.5000	27.8577

5.0000   26.8462

c. **Figure**

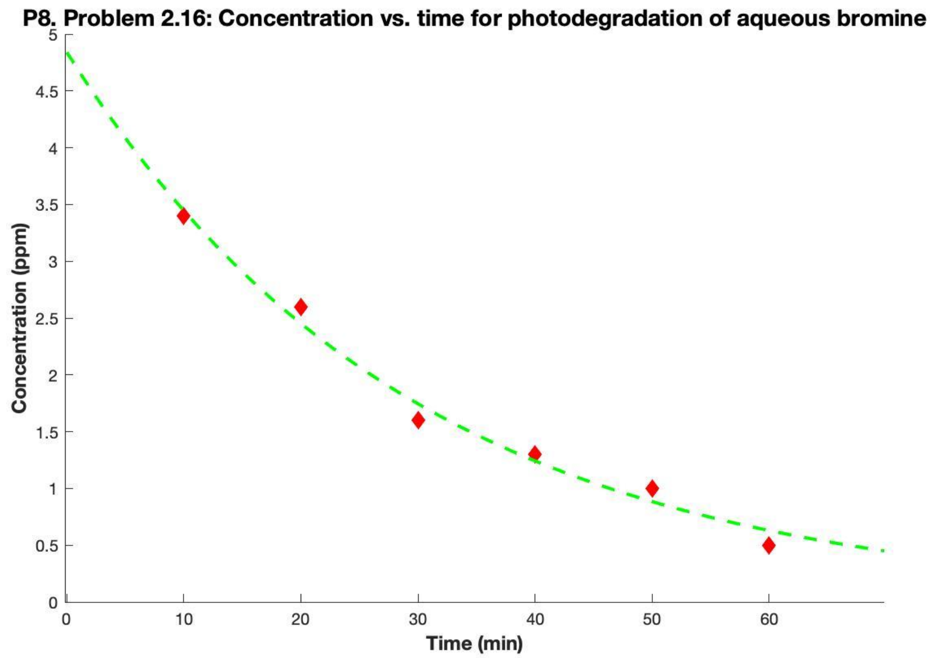


7. **Figure**

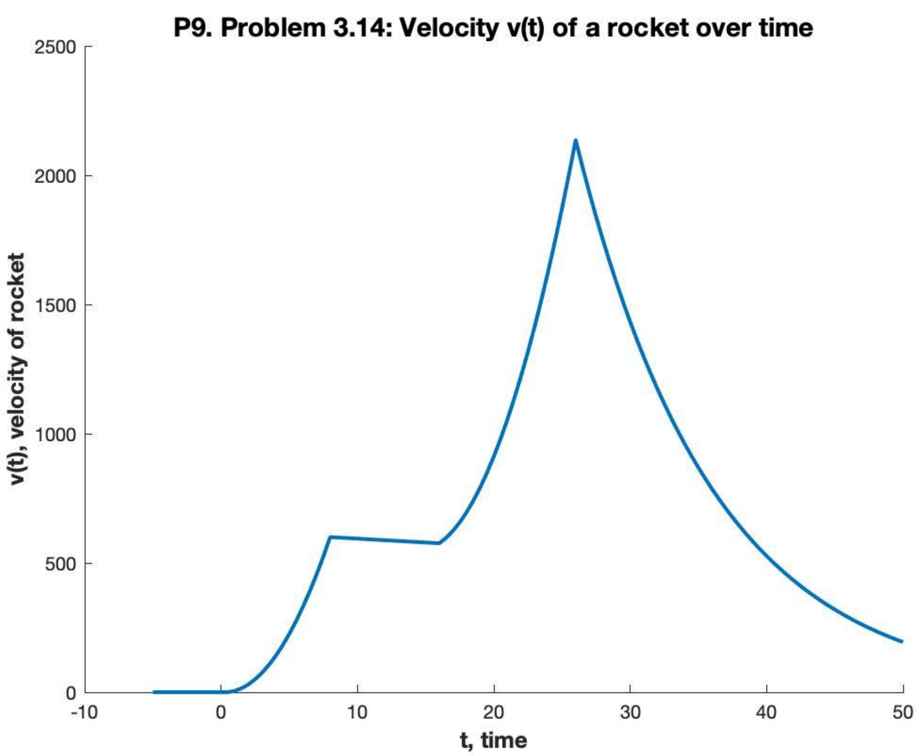


Robert Heeter
BIOE 391 Numerical Methods – Due 24 January 2022

8. Figure

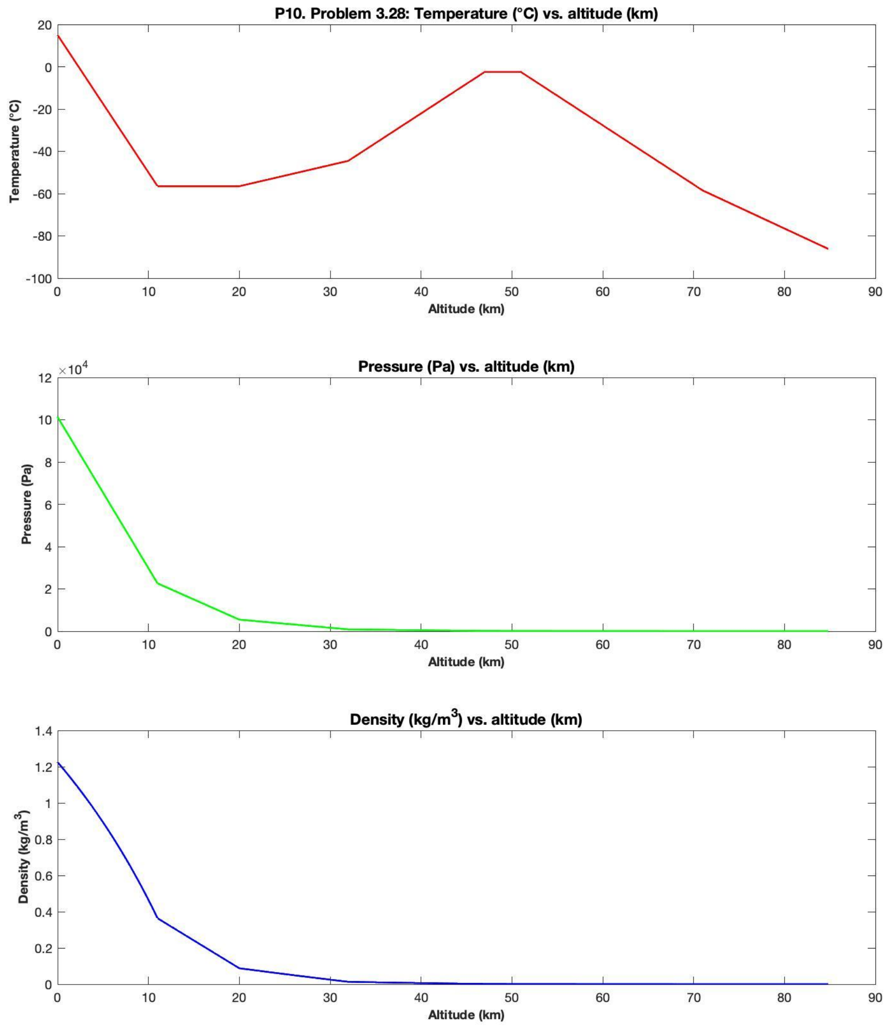


9. Figure



Robert Heeter
BIOE 391 Numerical Methods – Due 24 January 2022

10. Figure



Robert Heeter
BIOE 391 Numerical Methods – Due 24 January 2022

Complete MATLAB Code

```matlab
% Robert Heeter
% BIOE 391 Numerical Methods
% HOMEWORK 1 MATLAB SCRIPT

clc, clf, clear, close all

%% P1. PROBLEM 1.4
disp('P1. PROBLEM 1.4');

g = 9.81;
c_d = 0.25;
m = 68.1;
dvdt = @(t,v) g-(c_d./m).*v.^2; % differential equation

% 1.4a
h = 1; % step size = 1s
[t,v] = eulode(dvdt,[0,12],0,h); % use Euler's method function
disp('PART A: Time, velocities to t = 12s; step size = 1s');
disp([t,v]);
disp('');

% 1.4b
h = 0.5; % step size = 0.5s
[t,v] = eulode(dvdt,[0,12],0,h); % use Euler's method function
disp('PART B: Time, velocities to t = 12s; step size = 0.5s');
disp([t,v]);
disp('');


%% P2. PROBLEM 1.5
disp('P2. PROBLEM 1.5');

% 1.5b
g = 9.81;
c = 11.5;
m = 68.1;
dvdt = @(t,v) g-(c./m).*v; % differential equation using new drag eqn.
h = 2; % step size = 2s
[t,v] = eulode(dvdt,[0,40],0,h); % use Euler's method function
disp('Time, velocities to t = 40s; step size = 2s');
disp([t,v]);
disp('');


%% P3. PROBLEM 1.9
disp('P3. PROBLEM 1.9');

A = 1250;
Q = 450;
dydt = @(t,y) (3*(Q/A)*(sin(t)).^2)-(Q/A); % differential equation
h = 0.5; % step size = 0.5d
[t,y] = eulode(dydt,[0,10],0,h); % use Euler's method function
disp('Time, depths to t = 10d; step size = 0.5d');
disp([t,y]);
disp('');


%% P4. PROBLEM 1.13
disp('P4. PROBLEM 1.13');

% No MATLAB code for this problem
```

Robert Heeter
BIOE 391 Numerical Methods – Due 24 January 2022

```
%% P5. PROBLEM 1.17
disp('P5. PROBLEM 1.17');

T_a = 20;
k = 0.019;
dTdt = @(t,T) -k*(T-T_a); % differential equation
h = 2; % step size = 2m
[t,T] = eulode(dTdt,[0,20],70,h); % use Euler's method function
disp('Time, temperature to t = 20m; step size = 2m');
disp([t,T]);
disp('');


%% P6. PROBLEM 1.18
disp('P6. PROBLEM 1.18');

% 1.18a
T_a = 10;
k = 0.12;
dTdt = @(t,T) -k*(T-T_a); % differential equation
h = 0.5; % step size = 0.5hr
[t,T1] = eulode(dTdt,[0,5],37,h); % use Euler's method function
disp('PART A: Time, temperature to t = 5hr; step size = 0.5hr');
disp([t,T1]);
disp('');

% 1.18b
dTdt = @(t,T) -k*(T-(20-2*t)); % adjusted differential equation
[t,T2] = eulode(dTdt,[0,5],37,h); % use Euler's method function
disp('PART B: Time, temperature to t = 5hr; step size = 0.5hr');
disp([t,T2]);
disp('');

% 1.18c
figure % plot both temperature results together with time
hold on
plot(t,T1,'-b','LineWidth',2);
plot(t,T2,'-r','LineWidth',2);
xlabel('Time (hrs)','FontSize',12,'FontWeight','bold');
ylabel('Temperature (Â°C)','FontSize',12,'FontWeight','bold');
title('P6. Problem 1.18c: Comparing homicide victim body cooling for different ambient temp.
conditions','FontSize',14,'FontWeight','bold');
legend('Constant ambient temp. (10Â°C)','Linearly decreasing ambient temp. (20 to 10Â°C)');
hold off


%% P7. PROBLEM 2.14
disp('P7. PROBLEM 2.14');

T_f = (32:3.6:93.2)'; % vector of Â°F temperatures
T_c = (5/9).*(T_f-32); % conversion to Â°C
rho = (5.5289e-8)*(T_c.^3)-(8.5016e-6)*(T_c.^2)+(6.5622e-5)*(T_c)+0.99987; % solving for density

figure % plot density and temperature
hold on
plot(T_c,rho,'LineWidth',2);
xlabel('Temperature (Â°C)','FontSize',12,'FontWeight','bold');
ylabel('Freshwater Density (g/cm^3)','FontSize',12,'FontWeight','bold');
title('P7. Problem 2.14: Freshwater density with respect to
temperature','FontSize',14,'FontWeight','bold');
hold off


%% P8. PROBLEM 2.16
disp('P8. PROBLEM 2.16');
```

Robert Heeter

BIOE 391 Numerical Methods – Due 24 January 2022

```
c_t = @(t) 4.84*exp(-0.034*t); % concentration equation
t = [10 20 30 40 50 60]'; % time values
c = [3.4 2.6 1.6 1.3 1.0 0.5]'; % concentration values

figure % plot data and function together
hold on
plot(t,c,'rd','MarkerFaceColor','r','MarkerSize',8);
fplot(c_t,[0,70],'g--','LineWidth',2);
xlabel('Time (min)','FontSize',12,'FontWeight','bold');
ylabel('Concentration (ppm)','FontSize',12,'FontWeight','bold');
title('P8. Problem 2.16: Concentration vs. time for photodegradation of aqueous
bromine','FontSize',14,'FontWeight','bold');
hold off


%% P9. PROBLEM 3.14
disp('P9. PROBLEM 3.14');

% M-file function 'v_piecewise()' written below

t = [-5:0.5:50]'; % create vector of time values to use
[v] = v_piecewise(t); % find velocity vector from time vector with fxn
figure % plot time and velocity
hold on
plot(t,v,'LineWidth',2);
xlabel('t, time','FontSize',12,'FontWeight','bold');
ylabel('v(t), velocity of rocket','FontSize',12,'FontWeight','bold');
title('P9. Problem 3.14: Velocity v(t) of a rocket over time','FontSize',14,'FontWeight','bold');
hold off


%% P10. PROBLEM 3.28
disp('P10. PROBLEM 3.28');

% Function 'StdAtm()' written below

% Script to generate a plot of temperature, pressure and density for the
% International Standard Atmosphere
h = [0 11 20 32 47 51 71 84.852]; % base heights
g = [-6.5 0 1 2.8 0 -2.8 -2]; % gamma, lapse rates
T = [15 -56.5 -56.5 -44.5 -2.5 -2.5 -58.5 -86.28]; % base temperatures
p = [101325 22632 5474.9 868.02 110.91 66.939 3.9564 0.3734]; % base pressures

hint = [0:0.1:84.852]; % create vector of altitudes to use
Tint = zeros(size(hint)); % preallocate temperature, pressure, and density vectors
pint = zeros(size(hint));
rint = zeros(size(hint));

for i = 1:length(hint)
    [Tint(i),pint(i),rint(i)] = StdAtm(h,T,p,g,hint(i)); % use StdAtm() fxn
end

figure % plot altitude with temperature, pressure, and density together
hold on
subplot(3,1,1)
plot(hint,Tint,'-r','LineWidth',1.5);
xlabel('Altitude (km)','FontSize',10,'FontWeight','bold');
ylabel('Temperature (Â°C)','FontSize',10,'FontWeight','bold');
title('P10. Problem 3.28: Temperature (Â°C) vs. altitude (km)','FontSize',12,'FontWeight','bold');

subplot(3,1,2)
plot(hint,pint,'-g','LineWidth',1.5);
xlabel('Altitude (km)','FontSize',10,'FontWeight','bold');
ylabel('Pressure (Pa)','FontSize',10,'FontWeight','bold');
title('Pressure (Pa) vs. altitude (km)','FontSize',12,'FontWeight','bold');
```

Robert Heeter

BIOE 391 Numerical Methods – Due 24 January 2022

```
subplot(3,1,3)
plot(hint,rint,'-b','LineWidth',1.5);
xlabel('Altitude (km)','FontSize',10,'FontWeight','bold');
ylabel('Density (kg/m^3)','FontSize',10,'FontWeight','bold');
title('Density (kg/m^3) vs. altitude (km)','FontSize',12,'FontWeight','bold');

hold off


%% Additional Functions

function [v] = v_piecewise(t)
% ABOUT: For P9. Problem 3.14. Uses a for-loop with conditional statements
% to give the outputs of a piecewise function for an interval/vector of
% inputs.
% INPUTS: t, vector of values to plug into piecewise function
% OUTPUTS: v, vector of outputs from piecewise function

v = zeros(size(t));
for i = 1:length(t)
    if(0<=t(i) && t(i)<=8)
        v(i) = 10.*(t(i).^2)-(5.*t(i));
    elseif(8<t(i) && t(i)<=16)
        v(i) = 624-(3.*t(i));
    elseif(16<t(i) && t(i)<=26)
        v(i) = (36.*t(i))+12.*((t(i)-16).^2);
    elseif(t(i)>26)
        v(i) = 2136.*exp(-0.1.*(t(i)-26));
    end
end

end


function [T_ha,p_ha,r_ha] = StdAtm(h,T,p,g,h_a)
% ABOUT: For P10. Problem 3.28. Uses equations from the textbook to find
% the temperature, pressure, and density of air at a given altitude using
% tabulated data.
% INPUTS: h_a = height; T = base temp; p = base pressure; g = gamma = lapse
% rate; h = base height
% OUTPUTS: T_ha = temperature at h_a; p_ha = pressure at h_a; r_ha =
% density at h_a

% Check if h_a is in range of altitudes given by h
if h_a < h(1) || h_a > h(end)
    error('Altitude is outside of range.');
end

% Find layer number for given altitude h_a
i = 8; % start at top layer and work down
while h_a < h(i) % if given altitude is lower than base level of a layer
    i = i-1; % move down a layer
end

% Calcuate values for given altitude h_a
M = 0.0289644; % molar mass constant
R = 8.3144621; % universal gas constant

T_ha = T(i)+(g(i)*(h_a-h(i))); % solve for temperature
p_ha = p(i)+((p(i+1)-p(i))/(h(i+1)-h(i)))*(h_a-h(i)); % solve for pressure
r_ha = (p_ha*M)/(R*(T_ha+273.15)); % solve for density

end


function [t,y] = eulode(dydt,tspan,y0,h,varargin) % FROM TEXTBOOK
```

Robert Heeter

BIOE 391 Numerical Methods – Due 24 January 2022

```
% ABOUT: eulode - Euler ODE solver
%   [t,y] = eulode(dydt,tspan,y0,h,p1,p2,...):
%           uses Euler's method to integrate an ODE

% INPUTS:
%   dydt = name of the M-file that evaluates the ODE
%   tspan = [ti, tf] where ti and tf = initial and
%   final values of independent variable
%   y0 = initial value of dependent variable
%   h = step size
%   p1,p2,... = additional parameters used by dydt

% Outputs:
%   t = vector of independent variable
%   y = vector of solution for dependent variable

if nargin<4,error('at least 4 input arguments required'),end
ti = tspan(1);tf = tspan(2);
if ~(tf>ti),error('upper limit must be greater than lower'),end
t = (ti:h:tf)'; n = length(t);
% if necessary, add an additional value of t
% so that range goes from t = ti to tf
if t(n)<tf
    t(n+1) = tf;
    n = n+1;
end
y = y0*ones(n,1); % preallocate y to improve efficiency
for i = 1:n-1 % implement Euler's method
    y(i+1) = y(i) + dydt(t(i),y(i),varargin{:})*(t(i+1)-t(i));
end
end
```

Robert Heeter

BIOE 391

1/24/22

Problem Set #1

2.a. $F_u = -c'v$

$F_N = F_g - F_u = mg - c'v = m\frac{dv}{dt} \rightarrow \frac{dv}{dt} = g - \frac{c'v}{m}$ (differential equation)

$\frac{1}{g - \frac{c'v}{m}}\,dv = dt$ (separate)

$\int \frac{1}{g - \frac{c'v}{m}}\,dv = \int dt$ (integrate)

$-\ln\left(g - \frac{c'v}{m}\right)\left(\frac{m}{c'}\right) + C_1 = t$

$\ln\left(g - \frac{c'v}{m}\right) = -\frac{tc'}{m} + C_2$

$g - \frac{c'v}{m} = C_3 e^{-\frac{c't}{m}}$ → use initial condition $v = 0$ at $t = 0$

$\therefore C_3 = g$

$\frac{c'v}{m} = g\left(1 - e^{-\frac{c't}{m}}\right)$

$v = \frac{mg}{c'}\left(1 - e^{-\frac{c't}{m}}\right) \longrightarrow \boxed{v = \frac{mg}{c}\left(1 - e^{-\frac{c't}{m}}\right)}$

4. Input − Output + Generation − Consumption = Accumulation

$\underbrace{(1L}_{\text{food}} + \underbrace{0.05L}_{\text{inhale}} + \underbrace{D)}_{\text{drinking}} - (\underbrace{0.4L}_{\text{exhale}} + \underbrace{0.3}_{\text{sweat}} + \underbrace{1.4}_{\text{urine}} + \underbrace{0.2}_{\text{feces}} + \underbrace{0.35)}_{\text{skin}} + \underbrace{(0.3)}_{\text{metabolism}} = 0$ (steady state)

$D = 1.3L$ → $\boxed{\text{1.3 Liters must be drunk per day}}$