

UWTSD

Module

Task 3

Module code: ACHA7002

Anthony Yin Wai Hau

Student Number:1550559

5-12-2025

Contents

1	Task 3a	1
1.1	Table of own functions results compared with given function.....	1
2	Task 3b	1
2.1	Table of research of MATLAB/Online functions	1
2.1.1	Additional researched functions findings	1
2.2	Table of tried MATLAB/Online functions results	2
3	Referencing.....	2
4	Appendices.....	3
4.1	Own code and functions task 3a	3
4.2	Own code task 3b	12

1 Task 3a

1.1 Table of own functions results compared with given function

Method	n5/nt	Err(f)	Time(s)	Iterations	Δn_5
Original	0.051501	6.81E-13	0.0201	4432	
Bisection	0.051501	2.11E-15	0.0072	36	1.53E-12
Secant	0.051501	0	0.0019	6	1.53E-12
Newton	0.051501	0	0.0017	4	1.53E-12

2 Task 3b

2.1 Table of research of MATLAB/Online functions

Method & MATLAB Tooling	Technique Type	Ease of Use (1–5)	Efficiency (1–5)	Accuracy (1–5)	Notes
Fmincon (MathWorks, 2024a)	Nonlinear constrained optimization	3	5	5	Best for general Gibbs minimization with constraints, but requires Optimization Toolbox and some setup
Fsolve (MathWorks, 2024b)	Nonlinear root-finding	4	4	4	good double-precision accuracy but can converge to wrong root if initial guess is poor or system is stiff.
solve / vpasolve (MathWorks, 2024c)	Symbolic algebra + numeric solving	3	3	5	High precision; good for analytical manipulation. but must manage assumptions, variable ranges, and initial guesses

\(MathWorks, 2024d) / linsolve (MathWorks, 2024e)	Direct linear algebra	5	5	4	Very strong if Gibbs model reduces to a linear system , Not suitable for fully nonlinear G(n) without approximation.
Chemistry toolboxes (e.g., getEqmComp) (Julian, 2024)	Domain-specific Gibbs minimization	4	4	4	ideal for quick, validated thermochemical equilibrium calculations

2.1.1 Additional researched functions findings

2.2 Table of tried MATLAB/Online functions results

Compare original functions TIME, ITERATIONS, RESULT DIFFERENCE, LINES of CODE with your results for researched code functions

Running code (Moodle zip) with all associated files must be submitted to receive marks for your results table

3 Referencing

MathWorks, 2024a. fmincon – Constrained Nonlinear Optimization. Available at: <https://uk.mathworks.com/help/optim/ug/fmincon.html> [Accessed 4 Dec. 2024].

MathWorks, 2024b. fsolve – Solve Nonlinear Equations. Available at: <https://uk.mathworks.com/help/optim/ug/fsolve.html> [Accessed 4 Dec. 2024].

MathWorks, 2024c. solve and vpasolve – Symbolic Math Toolbox. Available at: <https://uk.mathworks.com/help/symbolic/solve.html> [Accessed 4 Dec. 2024].

MathWorks, 2024d. mldivide (“\”) – Linear System Solver. Available at: <https://uk.mathworks.com/help/matlab/ref/mldivide.html> [Accessed 4 Dec. 2024].

MathWorks, 2024e. linsolve – Solve Linear Systems. Available at: <https://uk.mathworks.com/help/matlab/ref/linsolve.html> [Accessed 4 Dec. 2024].

Julian, 2024. *Chemical Equilibrium* (Version 1.0.0). MATLAB Central File Exchange. Available at: <https://uk.mathworks.com/matlabcentral/fileexchange/160431-chemical-equilibrium> [Accessed 4 Dec. 2025].

4 Appendices

4.1 Own code and functions task 3a

4.1.1 Task_3

```
clc; clear;

c = 8;
h = 18;
o = 0;
airs = 12.5;

T = 2000;
P = 40;
phi = 1.0;

% ----- Original -----
tic
[n5_orig, err_orig, it_orig] = n5_original(c,h,o,T,phi,airs,P);
time_orig = toc;

% ----- Bisection -----
tic
[n5_bis, err_bis, it_bis] = n5_bisection(c,h,o,T,phi,airs,P);
time_bis = toc;

% ----- Secant -----
tic
[n5_sec, err_sec, it_sec] = n5_secant(c,h,o,T,phi,airs,P);
time_sec = toc;

% ----- Newton -----
tic
[n5_new, err_new, it_new] = n5_newton(c,h,o,T,phi,airs,P);
time_new = toc;

% ----- Comparison -----
fprintf("\nMethod      n5/nt      Err(f)      Time(s)      Iter      |\Delta n5|\n");
fprintf("-----\n");
fprintf("Original      %12.5e      %10.3e      %8.4f      %4d      ---\n", n5_orig, err_orig,
time_orig, it_orig);
fprintf("Bisection      %12.5e      %10.3e      %8.4f      %4d      %10.3e\n", n5_bis, err_bis,
time_bis, it_bis, abs(n5_bis-n5_orig));
fprintf("Secant        %12.5e      %10.3e      %8.4f      %4d      %10.3e\n", n5_sec, err_sec,
time_sec, it_sec, abs(n5_sec-n5_orig));
fprintf("Newton        %12.5e      %10.3e      %8.4f      %4d      %10.3e\n", n5_new, err_new,
time_new, it_new, abs(n5_new-n5_orig));
```

4.1.2 n5_equation

```
function val =
n5_equation(x,carbon,hydrogen,oxygen,temperature,fuelequiv,molar_air_stoich,pressu
re)
```

```
% Prevent nonphysical values
```

```
if x <= 0
    val = 1e6;
    return
end
```

```
% ----- Gibbs free energy table -----
```

```
gibbseng=[0      -393.151      -238.921      -113.805
100      -393.683      -236.584      -120.239
200      -394.085      -232.766      -128.526
298.15    -394.389      -228.582      -137.163
300      -394.394      -228.5      -137.328
400      -394.675      -223.901      -146.338
500      -394.939      -219.051      -155.414
600      -395.182      -214.007      -164.486
700      -395.398      -208.812      -173.518
800      -395.586      -203.496      -182.497
900      -395.748      -198.083      -191.416
1000      -395.886      -192.59      -200.275
1100      -396.001      -187.033      -209.075
1200      -396.098      -181.425      -217.819
1300      -396.177      -175.774      -226.509
1400      -396.24      -170.089      -235.149
1500      -396.288      -164.376      -243.74
1600      -396.323      -158.639      -252.284
1700      -396.344      -152.883      -260.784
1800      -396.353      -147.111      -269.242
1900      -396.349      -141.325      -277.658
2000      -396.333      -135.528      -286.034
2100      -396.304      -129.721      -294.372
2200      -396.262      -123.905      -302.672
2300      -396.209      -118.082      -310.936
2400      -396.142      -112.252      -319.165
2500      -396.062      -106.416      -327.358
2600      -395.969      -100.575      -335.517
2700      -395.862      -94.729      -343.643
2800      -395.742      -88.878      -351.736
2900      -395.609      -83.023      -359.797
3000      -395.461      -77.163      -367.826
3100      -395.298      -71.298      -375.824
3200      -395.122      -65.43      -383.792
3300      -394.932      -59.558      -391.73
3400      -394.726      -53.681      -399.638
3500      -394.506      -47.801      -407.517
3600      -394.271      -41.916      -415.366
3700      -394.022      -36.027      -423.188
3800      -393.756      -30.133      -430.981
3900      -393.477      -24.236      -438.746
4000      -393.183      -18.334      -446.485
4100      -392.874      -12.427      -454.196
4200      -392.55      -6.516      -461.881
4300      -392.21      -0.6      -469.539
4400      -391.857      5.32      -477.171
```

4500	-391.488	11.245	-484.778
4600	-391.105	17.175	-492.359
4700	-390.706	23.111	-499.915
4800	-390.292	29.052	-507.446
4900	-389.862	34.998	-514.951
5000	-389.419	40.949	-522.433
5100	-388.959	46.906	-529.89
5200	-388.486	52.869	-537.323
5300	-387.996	58.838	-544.732
5400	-387.493	64.811	-552.118
5500	-386.974	70.791	-559.48
5600	-386.439	76.777	-566.818
5700	-385.89	82.769	-574.133
5800	-385.324	88.767	-581.424
5900	-384.745	94.77	-588.694
6000	-384.148	100.78	-595.939];

```
R = 8.31446e3;
```

```
gco2 = interp1(gibbseng(:,1),gibbseng(:,2),temperature)*1e3;
gh2o = interp1(gibbseng(:,1),gibbseng(:,3),temperature)*1e3;
gco = interp1(gibbseng(:,1),gibbseng(:,4),temperature)*1e3;
```

```
kco2 = exp(-(gco2 - gco)/(R*temperature));
kh2o = exp(-(gh2o)/(R*temperature));
```

```
% ----- Species Mole Fractions -----
a = sqrt(x * pressure);
```

```
n3 = (carbon * fuelequiv)/(kco2*a + 1);
n4 = ((hydrogen/2)* fuelequiv)/(kh2o*a + 1);
n1 = carbon*fuelequiv - n3;
n2 = (hydrogen/2)*fuelequiv - n4;
n5 = (molar_air_stoich*2 + oxygen*fuelequiv - 2*n1 - n2 - n3)/2;
n6 = molar_air_stoich*3.76;
```

```
ntsum = n1 + n2 + n3 + n4 + n5 + n6;
```

```
% ----- Root Function -----
val = x - n5/ntsum;
```

```
end
```

4.1.3 n5_original

```
function [out,error,iterations] =
n5_original(carbon,hydrogen,oxygen,temperature,fuelequiv,molar_air_stoich,pressure)
n5ntvalue = 0.1;
error = 1;
iterations = 0;
```

%TEMP	CO2	H2O	CO
gibbseng=[0	-393.151	-238.921	-113.805
100	-393.683	-236.584	-120.239
200	-394.085	-232.766	-128.526

298.15	-394.389	-228.582	-137.163
300	-394.394	-228.5	-137.328
400	-394.675	-223.901	-146.338
500	-394.939	-219.051	-155.414
600	-395.182	-214.007	-164.486
700	-395.398	-208.812	-173.518
800	-395.586	-203.496	-182.497
900	-395.748	-198.083	-191.416
1000	-395.886	-192.59	-200.275
1100	-396.001	-187.033	-209.075
1200	-396.098	-181.425	-217.819
1300	-396.177	-175.774	-226.509
1400	-396.24	-170.089	-235.149
1500	-396.288	-164.376	-243.74
1600	-396.323	-158.639	-252.284
1700	-396.344	-152.883	-260.784
1800	-396.353	-147.111	-269.242
1900	-396.349	-141.325	-277.658
2000	-396.333	-135.528	-286.034
2100	-396.304	-129.721	-294.372
2200	-396.262	-123.905	-302.672
2300	-396.209	-118.082	-310.936
2400	-396.142	-112.252	-319.165
2500	-396.062	-106.416	-327.358
2600	-395.969	-100.575	-335.517
2700	-395.862	-94.729	-343.643
2800	-395.742	-88.878	-351.736
2900	-395.609	-83.023	-359.797
3000	-395.461	-77.163	-367.826
3100	-395.298	-71.298	-375.824
3200	-395.122	-65.43	-383.792
3300	-394.932	-59.558	-391.73
3400	-394.726	-53.681	-399.638
3500	-394.506	-47.801	-407.517
3600	-394.271	-41.916	-415.366
3700	-394.022	-36.027	-423.188
3800	-393.756	-30.133	-430.981
3900	-393.477	-24.236	-438.746
4000	-393.183	-18.334	-446.485
4100	-392.874	-12.427	-454.196
4200	-392.55	-6.516	-461.881
4300	-392.21	-0.6	-469.539
4400	-391.857	5.32	-477.171
4500	-391.488	11.245	-484.778
4600	-391.105	17.175	-492.359
4700	-390.706	23.111	-499.915
4800	-390.292	29.052	-507.446
4900	-389.862	34.998	-514.951
5000	-389.419	40.949	-522.433
5100	-388.959	46.906	-529.89
5200	-388.486	52.869	-537.323
5300	-387.996	58.838	-544.732
5400	-387.493	64.811	-552.118
5500	-386.974	70.791	-559.48
5600	-386.439	76.777	-566.818
5700	-385.89	82.769	-574.133
5800	-385.324	88.767	-581.424
5900	-384.745	94.77	-588.694
6000	-384.148	100.78	-595.939];

```

%Gibbs free energy
gco2=interp1(gibbseng(:,1),gibbseng(:,2),temperature)*1e3;
gh2o=interp1(gibbseng(:,1),gibbseng(:,3),temperature)*1e3;
gco =interp1(gibbseng(:,1),gibbseng(:,4),temperature)*1e3;

%Equilibrium constants
R = 8.31446e3;
kco2=exp(-(gco2-gco)/(R*temperature));
kh2o=exp(-(gh2o)/(R*temperature));

% ----- Goodger method (original search) -----

% 1) Coarse step: 1e-3
while error > 0
    a = sqrt(n5ntvalue*pressure);
    n3 = (carbon*fuelequiv)/(kco2*a+1);
    n4 = ((hydrogen/2)*fuelequiv)/(kh2o*a+1);
    n1 = carbon*fuelequiv-n3;
    n2 = (hydrogen/2)*fuelequiv-n4;
    n5 = (molar_air_stoich*2+oxygen*fuelequiv-2*n1-n2-n3)/2;
    n6 = molar_air_stoich*3.76;
    ntsum = n1+n2+n3+n4+n5+n6;

    error = n5ntvalue-n5/ntsum;
    iterations = iterations + 1;

    n5ntvalue = n5ntvalue-0.001;
    if n5ntvalue < 0
        break
    end
end

% 2) Medium step: 1e-6 (first branch: went below 0)
if n5ntvalue < 0
    n5ntvalue = 0.001;
    while error > 0
        a = sqrt(n5ntvalue*pressure);
        n3 = (carbon*fuelequiv)/(kco2*a+1);
        n4 = ((hydrogen/2)*fuelequiv)/(kh2o*a+1);
        n1 = carbon*fuelequiv-n3;
        n2 = (hydrogen/2)*fuelequiv-n4;
        n5 = (molar_air_stoich*2+oxygen*fuelequiv-2*n1-n2-n3)/2;
        n6 = molar_air_stoich*3.76;
        ntsum = n1+n2+n3+n4+n5+n6;

        error = n5ntvalue-n5/ntsum;
        iterations = iterations + 1;

        n5ntvalue = n5ntvalue-0.000001;
        if n5ntvalue < 0
            break
        end
    end
end

% 2b) Medium step: refine if error went negative
else
    while error < 0
        a = sqrt(n5ntvalue*pressure);
        n3 = (carbon*fuelequiv)/(kco2*a+1);

```

```

n4 = ((hydrogen/2)*fuelequiv)/(kh2o*a+1);
n1 = carbon*fuelequiv-n3;
n2 = (hydrogen/2)*fuelequiv-n4;
n5 = (molar_air_stoich*2+oxygen*fuelequiv-2*n1-n2-n3)/2;
n6 = molar_air_stoich*3.76;
ntsum = n1+n2+n3+n4+n5+n6;

error = n5ntvalue-n5/ntsum;
iterations = iterations + 1;

n5ntvalue = n5ntvalue+0.000001;
end
end

% 3) Finer step: 1e-9
if n5ntvalue < 0
    n5ntvalue = 0.000001;
    while error > 0
        a = sqrt(n5ntvalue*pressure);
        n3 = (carbon*fuelequiv)/(kco2*a+1);
        n4 = ((hydrogen/2)*fuelequiv)/(kh2o*a+1);
        n1 = carbon*fuelequiv-n3;
        n2 = (hydrogen/2)*fuelequiv-n4;
        n5 = (molar_air_stoich*2+oxygen*fuelequiv-2*n1-n2-n3)/2;
        n6 = molar_air_stoich*3.76;
        ntsum = n1+n2+n3+n4+n5+n6;

        error = n5ntvalue-n5/ntsum;
        iterations = iterations + 1;

        n5ntvalue = n5ntvalue-0.000000001;
        if n5ntvalue < 0
            break
        end
    end
else
    while error > 0
        a = sqrt(n5ntvalue*pressure);
        n3 = (carbon*fuelequiv)/(kco2*a+1);
        n4 = ((hydrogen/2)*fuelequiv)/(kh2o*a+1);
        n1 = carbon*fuelequiv-n3;
        n2 = (hydrogen/2)*fuelequiv-n4;
        n5 = (molar_air_stoich*2+oxygen*fuelequiv-2*n1-n2-n3)/2;
        n6 = molar_air_stoich*3.76;
        ntsum = n1+n2+n3+n4+n5+n6;

        error = n5ntvalue-n5/ntsum;
        iterations = iterations + 1;

        n5ntvalue = n5ntvalue-0.000000001;
    end
end

% 4) Finest step: 1e-12
if n5ntvalue < 0
    n5ntvalue = 0.000000001;
    while error > 0
        a = sqrt(n5ntvalue*pressure);
        n3 = (carbon*fuelequiv)/(kco2*a+1);

```

```

n4 = ((hydrogen/2)*fuelequiv)/(kh2o*a+1);
n1 = carbon*fuelequiv-n3;
n2 = (hydrogen/2)*fuelequiv-n4;
n5 = (molar_air_stoich*2+oxygen*fuelequiv-2*n1-n2-n3)/2;
n6 = molar_air_stoich*3.76;
ntsum = n1+n2+n3+n4+n5+n6;

error = n5ntvalue-n5/ntsum;
iterations = iterations + 1;

n5ntvalue = n5ntvalue-0.000000000001;
if n5ntvalue < 0
    break
end
end
else
    while error < 0
        a = sqrt(n5ntvalue*pressure);
        n3 = (carbon*fuelequiv)/(kco2*a+1);
        n4 = ((hydrogen/2)*fuelequiv)/(kh2o*a+1);
        n1 = carbon*fuelequiv-n3;
        n2 = (hydrogen/2)*fuelequiv-n4;
        n5 = (molar_air_stoich*2+oxygen*fuelequiv-2*n1-n2-n3)/2;
        n6 = molar_air_stoich*3.76;
        ntsum = n1+n2+n3+n4+n5+n6;

        error = n5ntvalue-n5/ntsum;
        iterations = iterations + 1;

        n5ntvalue = n5ntvalue+0.000000000001;
    end
end

out = n5ntvalue;
end

```

4.1.4 n5_bisection

```

function [root, err, iterations] =
n5_bisection(carbon,hydrogen,oxygen,temperature,fuelequiv,molar_air_stoich,pressur
e)

a = 1e-8;
b = 0.1;

fa =
n5_equation(a,carbon,hydrogen,oxygen,temperature,fuelequiv,molar_air_stoich,pressu
re);
fb =
n5_equation(b,carbon,hydrogen,oxygen,temperature,fuelequiv,molar_air_stoich,pressu
re);

if fa*fb > 0
    error('Bisection: f(a) and f(b) do not bracket a root.');
```

```

end

tol = 1e-12;
iterations = 0;

```

```

while (b - a)/2 > tol
    iterations = iterations + 1;

    c = 0.5*(a + b);
    fc =
n5_equation(c,carbon,hydrogen,oxygen,temperature,fuelequiv,molar_air_stoich,pressure);

    if fa*fc < 0
        b = c; fb = fc;
    else
        a = c; fa = fc;
    end
end

root = 0.5*(a + b);
err =
abs(n5_equation(root,carbon,hydrogen,oxygen,temperature,fuelequiv,molar_air_stoich,pressure));
end

```

4.1.5 n5_secant

```

function [root, err, iterations] =
n5_secant(carbon,hydrogen,oxygen,temperature,fuelequiv,molar_air_stoich,pressure)

    x0 = 1e-8;
    x1 = 0.1;

    f0 =
n5_equation(x0,carbon,hydrogen,oxygen,temperature,fuelequiv,molar_air_stoich,pressure);
    f1 =
n5_equation(x1,carbon,hydrogen,oxygen,temperature,fuelequiv,molar_air_stoich,pressure);

    tol = 1e-12;
    iterations = 0;

    while abs(x1 - x0) > tol
        iterations = iterations + 1;

        if abs(f1 - f0) < eps
            break
        end

        x2 = x1 - f1*(x1 - x0)/(f1 - f0);
        if x2 <= 0, x2 = x1/2; end

        f2 =
n5_equation(x2,carbon,hydrogen,oxygen,temperature,fuelequiv,molar_air_stoich,pressure);

        x0 = x1; f0 = f1;
        x1 = x2; f1 = f2;

        if abs(f2) < tol
            break
        end
    end

```

```

    end

    root = x1;
    err =
abs(n5_equation(root, carbon, hydrogen, oxygen, temperature, fuelequiv, molar_air_stoich
, pressure));
end

```

4.1.6 n5_newton

```

function [root, err, iterations] =
n5_newton(carbon, hydrogen, oxygen, temperature, fuelequiv, molar_air_stoich, pressure)

    x = 0.05;
    tol = 1e-12;
    iterations = 0;

    while true
        iterations = iterations + 1;

        fx =
n5_equation(x, carbon, hydrogen, oxygen, temperature, fuelequiv, molar_air_stoich, pressu
re);
        if abs(fx) < tol
            break
        end

        h = 1e-6*max(1,x);
        f_plus =
n5_equation(x+h, carbon, hydrogen, oxygen, temperature, fuelequiv, molar_air_stoich, pres
sure);
        f_minus = n5_equation(x-
h, carbon, hydrogen, oxygen, temperature, fuelequiv, molar_air_stoich, pressure);

        dfdx = (f_plus - f_minus)/(2*h);
        if abs(dfdx) < eps, break, end

        x_new = x - fx/dfdx;
        if x_new <= 0, x_new = x/2; end
        if abs(x_new - x) < tol
            x = x_new;
            break
        end

        x = x_new;
    end

    root = x;
    err =
abs(n5_equation(root, carbon, hydrogen, oxygen, temperature, fuelequiv, molar_air_stoich
, pressure));
end

```

Module code

Task 3

Name

4.2 Own code task 3b