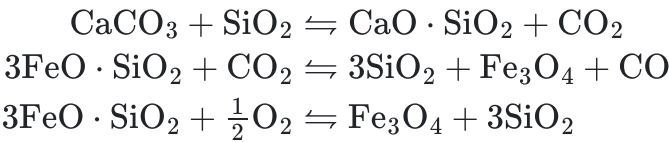


cheg325 homework7 SIS 13.20

AUTHOR  
kyle wodehouse

PUBLISHED  
April 19, 2025

we need to find allll the  $\Delta H$  and  $\Delta G$  for the reactions



from an earlier homework i have the appendix with all the formation energies as a `csv` file and will import it as a dataframe to make menial math easy

```
import pandas as pd
import numpy as np
from scipy.constants import R

a4 = pd.read_csv('appendix_a4.csv', index_col=1)
a4.head(3)
```

	chemical_name	state	delta_h_form	delta_g_form
chemical_formula				
CH4	Methane	g	-74.5	-50.5
C2H6	Ethane	g	-83.8	-31.9
C3H8	Propane	g	-104.7	-24.3

thankfully sandler says ‘go look in the chemical engineering handbook’ to find the ones not in his table, so after some digging i was able to get the numbers needed from the 8th edition of the chemical engineering handbook. I also took the SiO<sub>2</sub> heat and free energy of formation from the handbook as it was giving me wildly unphysical results in calculations.

values i found from handbook and also the SiO2 replacement value (note that 1 kcal = 4.184 kJ)

Compound	$\Delta H_f$ (kcal/mol)	$\Delta G_f$ (kcal/mol)	$\Delta H_f$ (kJ/mol)	$\Delta G_f$ (kJ/mol)
Fe <sub>3</sub> O <sub>4</sub>	-266.9	-242.3	-1116.7096	-1013.7832
CaO·SiO <sub>2</sub>	-377.9	-357.5	-1581.1336	-1495.78
CaCO <sub>3</sub>	-289.5	-270.8	-1211.268	-1133.0272
SiO <sub>2</sub>	-203.35	-190.4	-850.8164	-796.6336

now we (i) can easily calculate all the  $K_a$  values at standard state

$$K_a^\circ = \exp\left(\frac{-\Delta_{\text{rxn}}G^\circ}{RT}\right)$$

```
reaction1_components = ['CaCO3', 'SiO2', 'CaO*SiO2', 'CO2']
r1_coeffs = np.array([-1, -1, 1, 1])
temp = a4.loc[reaction1_components]
G1 = (temp['delta_g_form'].astype(float) * r1_coeffs).sum() * 1000

reaction2_components = ['Fe0*SiO2', 'CO2', 'SiO2', 'Fe3O4', 'CO']
r2_coeffs = np.array([-3, -1, 3, 1, 1])
temp = a4.loc[reaction2_components]
G2 = (temp['delta_g_form'].astype(float) * r2_coeffs).sum() * 1000

reaction3_components = ['Fe0*SiO2', 'O2', 'Fe3O4', 'SiO2']
r3_coeffs = np.array([-3.0, -0.5, 1.0, 3.0])
temp = a4.loc[reaction3_components]
G3 = (temp['delta_g_form'].astype(float) * r3_coeffs).sum() * 1000

print(f'G1: {G1:.1f} J/mol\nG2: {G2:.1f} J/mol\nG3: {G3:.1f} J/mol')

Gs = np.array([G1, G2, G3])
K = np.exp(-Gs / (R*298.15))
print(f'Ko: {K}')
```

G1: 35253.6 J/mol  
G2: 33516.0 J/mol  
G3: -223684.0 J/mol  
Ko: [6.66555768e-07 1.34354074e-06 1.54078254e+39]

very quickly we can guess that the first two reactions should be fairly reactant favored and the 3rd reaction should be heavily product favored.

this approach is very similar to (13.1-22b) (and for some reason there are two equations tagged 13.1-22b). here’s the outline:

- using free energies of formations and reaction stoichiometry to get  $\Delta_{\text{rxn}}G$
- using  $\Delta_{\text{rxn}}G$  to find  $K$  at 298.15 K
- using enthalpies of formations, reaction stoichiometry, and  $C_P$  polynomials to correct K for the difference in temperature between 298.15 and the ~750K

now let’s get down to the mathy business. we need to get  $\Delta_{\text{rxn}}H^\circ(T)$

$$\Delta_{\text{rxn}}H(T) = \Delta_{\text{rxn}}H^\circ(298.15) + \int_{293.15}^T \Delta_{\text{rxn}}C_p \, dT$$

if the  $C_p$  is in the form of

$$C_p = a + bT + e/T^2$$

then

$$\int_{T_1}^{T_2} C_p = a[T_2 - T_1] + \frac{1}{2}b[T_2 - T_1]^2 + e\left[\frac{1}{T_1} - \frac{1}{T_2}\right]$$

and then

$$\ln \frac{K_a(747)}{K_a(298.15)} = \int_{298.15}^{747} \frac{\Delta_{\text{rxn}}H(T)}{RT^2} \, dT$$

```
heat_capacity = pd.read_csv('heat_capacity.csv', index_col=0).fillna(0) # table from ques

def calculate_K_at_T(T, species, coeffs):
    cp_data = np.array(heat_capacity.loc[species])
    coeffs = np.array(coeffs)

    delta_a, delta_b, delta_c, delta_d, delta_e = (cp_data * coeffs[:, np.newaxis]).sum(axis=1)

    thermo_data = a4.loc[species]
    delta_H = (thermo_data['delta_h_form'].astype(float) * coeffs).sum() * 1000 # J/mol
    delta_G = (thermo_data['delta_g_form'].astype(float) * coeffs).sum() * 1000 # J/mol
    T1 = 298.15
    K_298 = np.exp(-delta_G / (R * T1))

    term1 = (delta_a/R) * np.log(T/T1)
    term2 = (delta_b/(2*R)) * (T - T1)
    term3 = (delta_c/(6*R)) * (T**2 - T1**2)
    term4 = (delta_d/(12*R)) * (T**3 - T1**3)
    term5 = (delta_e/(2*R)) * (1/T**2 - 1/T1**2)

    bracket_term = (-delta_H + delta_a*T1 + (delta_b/2)*T1**2 +
                    (delta_c/3)*T1**3 + (delta_d/4)*T1**4 - delta_e/T1)

    term6 = (1/R) * bracket_term * (1/T - 1/T1)

    ln_ratio = term1 + term2 + term3 + term4 + term5 + term6

    K_T = K_298 * np.exp(ln_ratio)

    return K_T

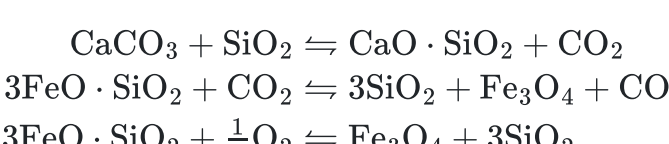
K1, K2, K3 = [calculate_K_at_T(747, components, coeffs) for components, coeffs
               in zip([reaction1_components,
                       reaction2_components,
                       reaction3_components],
                       [r1_coeffs, r2_coeffs, r3_coeffs])]

print(f'K1: {K1:.3f}')
print(f'K2: {K2:.3f}')
print(f'K3: {K3:.3e}')
```

K1: 340.891  
K2: 0.254  
K3: 4.141e+14

now we/i can use the K values and compare them to the atomsphere

Species	Mole Percent
CO <sub>2</sub>	99.9 –
H <sub>2</sub> O	0.01
CO	0.01
HCl	$1 \times 10^{-1}$
HF	$1 \times 10^{-6}$
O <sub>2</sub>	trace



right off the bat it makes sense that there’s basically no oxygen in the atmosphere since the only reaction including oxygen gas is so heavily product favored.

now for the ratio of CO<sub>2</sub> we can look at the second constant.

$$K_{\alpha,2} = \frac{\alpha_{CO}}{\alpha_{CO_2}} = 0.254$$

which doesn’t quite line up with the measurements, but at least the measurements show that there should be significantly more CO<sub>2</sub> than CO. if we look at the first reaction (say the pressure is 100 bar or so)

$$K_{\alpha,1} = \alpha_{\text{CO}_2} = 341 = y_{\text{CO}_2} \left(\frac{P}{1 \text{ bar}}\right) = y_{\text{CO}_2} \left(\frac{100 \text{ bar}}{1 \text{ bar}}\right)$$
$$y_{\text{CO}_2} = 3.41$$

obviously this is inconsistent with reality (the CO<sub>2</sub> is likely not an ideal gas at 100 bar), but it is generally consistent with the idea that the atmosphere should be mostly CO<sub>2</sub>

```
# filler
```