Converting_between_reaction_and_relative_enthalpies

July 30, 2024

Supplementary notebook to the manuscript "Unifying thermochemistry concepts in computational heterogeneous catalysis", by Dr. Bjarne Kreitz (Brown University), Dr. Gabriel S. Gusmão (Georgia Tech), Dingqi Nai (Georgia Tech), Sushree Jagriti Sahoo (Georgia Tech), Prof. Andrew A. Peterson (Brown University), Dr. David H. Bross (Argonne National Laboratory), Prof. C. Franklin Goldsmith (Brown University), Prof. Andrew J. Medford (Georgia Tech).

Correspondence to Bjarne Kreitz (bjarne_kreitz@brown.edu) and Andrew J. Medford (ajm@gatech.edu)

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec
from sklearn.linear_model import LinearRegression
from scipy import linalg

eV_to_kJmol=96.485
```

1 Converting between reaction enthalpies and formation enthalpies

The reaction enthalpies calculated from the total DFT energies are conserved when using an atomic reference basis set, regardless of the choice of anchor/reference species. We can use this fact and calculate EOFs if only reaction enthalpies are reported for a mechanism. This problem cannot be formulated for a single species, and it directly becomes a linear algebra problem. For a chemical kinetics network involving n chemical species, a total of $n_a + n_s$ reference species must be defined, where n_a is the number of unique atomic elements of which all species consist and n_s is the number of surface sites (typically one). Let $\underline{\mathbf{E}}$ be the zero-point corrected energies of chemical species, $\underline{\mathbf{H}}_{\mathrm{A}}$ be the EOFs of the same species given defined references A and $\underline{\mathbf{H}}_{\mathrm{r}}$ be the reaction energies according to the stoichiometry matrix $\underline{\underline{\mathbf{M}}}$. The relationship between $\underline{\mathbf{H}}_{\mathrm{r}}$ and $\underline{\mathbf{E}}$ is a simple linear combination,

$$\underline{\mathbf{H}}_{\mathrm{r}} = \underline{\underline{\mathbf{M}}}^{\top} \underline{\mathbf{H}}_{\mathrm{A}} = \underline{\underline{\mathbf{M}}}^{\top} \underline{\mathbf{E}} \tag{1}$$

Since the number of elementary reactions is typically greater than or equal to the number of chemical species, i.e., $m \geq n$, there exists a direct mapping between $\underline{\mathbf{E}}$ and $\underline{\mathbf{H}}_{\mathrm{r}}$. Yet, no obvious mapping exists in the opposite direction since this is an under-constrained linear system of equations. The Moore-Penrose pseudo-inverse of the stoichiometry matrix $\underline{\underline{\mathbf{M}}}^{+}$ can be used to construct the inverse mapping. To solve this inverse mapping, it is first necessary to select anchor species for each element

from the mechanism, e.g., $A = C_2H_6, CO, H_2O, Pt(111)$. We can separate the stoichiometry matrix $\underline{\underline{\mathbf{M}}}$ into the anchor species $\underline{\underline{\mathbf{M}}}_{\mathrm{A}}$ and non-anchor species $\hat{\underline{\underline{\mathbf{M}}}}$, leading to

$$\underline{\mathbf{H}}_{\mathrm{r}} = \underline{\underline{\mathbf{M}}}^{\top} \underline{\mathbf{H}}_{\mathrm{A}} = \hat{\underline{\underline{\mathbf{M}}}}^{\top} \hat{\underline{\mathbf{H}}}_{\mathrm{A}} + \underline{\underline{\mathbf{M}}}_{\mathrm{A}}^{\top} \tilde{\underline{\mathbf{H}}}_{\mathrm{A}} \tag{2}$$

In this equation, we separate $\underline{\mathbf{H}}_{\mathrm{A}}$ into the anchor species $\tilde{\underline{\mathbf{H}}}_{\mathrm{A}}$ and the unknown EOFs $\hat{\underline{\mathbf{H}}}_{\mathrm{A}}$. All EOFs are referenced to the DFT energies of the anchor species. Thus, the EOFs of the anchor species $\underline{\mathbf{H}}_{\mathrm{A}}$ are 0.
We can now rearrange the equation to compute the unknown EOFs from the reaction enthalpies referenced to the set of anchor species.

$$\underline{\mathbf{H}}_{\mathrm{r}} = \hat{\underline{\underline{\mathbf{M}}}}^{\top} \hat{\underline{\mathbf{H}}}_{\mathrm{A}} + \underline{\underline{\mathbf{M}}}_{\mathrm{A}}^{\top} \tilde{\underline{\mathbf{H}}}_{\mathrm{A}} \tag{3}$$

$$\hat{\underline{\mathbf{H}}}_{\mathrm{A}} = \left(\hat{\underline{\underline{\mathbf{M}}}}^{\top}\right)^{+} \left(\underline{\mathbf{H}}_{\mathrm{r}} - \underline{\underline{\mathbf{M}}}_{\mathrm{A}}^{\top} \tilde{\underline{\mathbf{H}}}_{\mathrm{A}}\right) \tag{4}$$

$$\hat{\underline{\mathbf{H}}}_{\mathrm{A}} = \left(\hat{\underline{\underline{\mathbf{M}}}}^{\top}\right)^{+} \left(\underline{\underline{\mathbf{M}}}^{\top} \underline{\mathbf{E}} - \underline{\underline{\mathbf{M}}}_{\mathrm{A}}^{\top} \tilde{\underline{\mathbf{H}}}_{\mathrm{A}}\right) \tag{5}$$

We start by constructing the matrix of stoichiometric matrix $\underline{\underline{\mathbf{M}}}$.

[2]:
```
# Stoichiometry of each elementary reaction in the mechanism
reactions = [
    {"X": -1, "C2H6": -1, "C2H6X": 1},
    {"X": -1, "CO2": -1, "CO2X": 1},
    {"X": -1, "C2H6X": -1, "XH": 1, "XCH2CH3": 1},
    {"X": -1, "XCH2CH3": -1, "XH": 1, "XCH2XCH2": 1},
    {"X": -1, "CO2X": -1, "XCO": 1, "XO":1},
    {"XO": -1, "XH": -1, "X": 1, "XOH": 1},
    {"XOH": -1, "XH": -1, "H2OX": 1, "X": 1},
    {"H2OX": -1, "H2O": 1, "X": 1},
    {"XCO": -1, "CO": 1, "X": 1},
    {"XCH2XCH2": -1, "C2H4": 1, "X":1},
]

#Species in the reaction mechanism
species = ['CO2', 'C2H4', 'C2H6X', 'XCH2CH3', 'XO', 'XCH2XCH2', 'XCO',
           'CO2X', 'XOH', 'H2OX', 'XH', 'X', 'C2H6','H2O', 'CO']
#the last for species will be our reference species

# Creating the stoichiometry matrix
M = np.zeros((len(species), len(reactions)))

# Filling the matrix
for j, reaction in enumerate(reactions):
    for reactant, stoich_coeff in reaction.items():
        i = species.index(reactant)
        M[i, j] = stoich_coeff
```

```
M
```

[2]:
```
array([[ 0., -1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.],
       [ 1.,  0., -1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  1., -1.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  1., -1.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0., -1.],
       [ 0.,  0.,  0.,  0.,  1.,  0.,  0.,  0., -1.,  0.],
       [ 0.,  1.,  0.,  0., -1.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  1., -1.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  1., -1.,  0.,  0.],
       [ 0.,  0.,  1.,  1.,  0., -1., -1.,  0.,  0.,  0.],
       [-1., -1., -1., -1., -1.,  1.,  1.,  1.,  1.,  1.],
       [-1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.,  0.]])
```

We can then calculate the enthalpies of reaction from the BEEF-vdW DFT energies

$$\underline{\mathbf{H}}_{\mathrm{r}} = \underline{\underline{\mathbf{M}}}^{\top}\underline{\mathbf{E}} \tag{6}$$

[3]:
```
#We use the BEEF-vdW DFT energies to calculate the reaction enthalpies for the␣
 ↪reactions above
species_energies = {
    "X": -377616.072,
    "C2H6": -615.3019939,
    "H2O": -611.0186083,
    "CO2": -1414.682864,
    "C2H6X" : -378231.5655,
    "XCH2CH3": -378214.9168,
    "XO": -378193.2832,
    "XCH2XCH2": -378198.4442,
    "XCO": -378453.0261,
    "CO2X": -379030.8164,
    "XOH": -378209.8045,
    "H2OX": -378227.2801,
    "XH": -377632.6636,
    "CO": -835.5389907,
    "C2H4":-581.44565067567,
}

#Sort the species energies according to the defined species list
E = np.array([species_energies[_]*eV_to_kJmol for _ in species])

#Calculate the enthalpies of formation
```

```
Hr=M.T.dot(E)
Hr
```

[3]:
```
array([-18.47746606,  -5.93730097,   5.5092935 , -11.48171501,
        55.87446348,   6.78289551, -85.29273999,  18.28310668,
       136.53682082,  89.39811156])
```

We select anchor species for each element from the mechanism, in this case $A = C_2H_6, CO, H_2O, Pt(111)$. Since we are creating a local thermochemical network in the DFT energy reference frame, we set the enthalpies of formation of these anchor species to 0.

[4]:
```
# Enthalpies of formation of the reference species are assumed to be zero

H_A_dict = {
    "X": 0,
    "C2H6": 0,
    "H2O": 0,
    "CO":0,
}

ref_species=H_A_dict.keys()

#Remove the reference species from the row space
M_A=M[-4:] # these are the reference species
M_hat=M[0:-4] # this is the rest of the stoichiometry matrix

#Array of the enthalpies of formation of the anchor species
tilde_H_A =  np.array( [H_A_dict[_] for _ in H_A_dict.keys()])
tilde_H_A
```

[4]:
```
array([0, 0, 0, 0])
```

We can then determine the enthalpies of formation of the target species via

$$\underline{\hat{\mathbf{H}}}_{\mathrm{A}} = \left(\underline{\underline{\hat{\mathbf{M}}}}^{\top}\right)^{+} \left(\underline{\underline{\mathbf{M}}}^{\top}\,\underline{\mathbf{E}} - \underline{\underline{\mathbf{M}}}_{\mathrm{A}}^{\top}\,\underline{\tilde{\mathbf{H}}}_{\mathrm{A}}\right) \tag{7}$$

[5]:
```
#Calculate the enthalpies of formation of the target species using the␣
 ↪pseudo-inverse
H_A= np.linalg.pinv(M_hat.T).dot(Hr-M_A.T.dot(tilde_H_A))

targets=species-ref_species

#Create a dictionary with the results
enthalpies_relative_to_anchors = {species[i]: H_A[i] for i in␣
 ↪range(len(targets))}
```

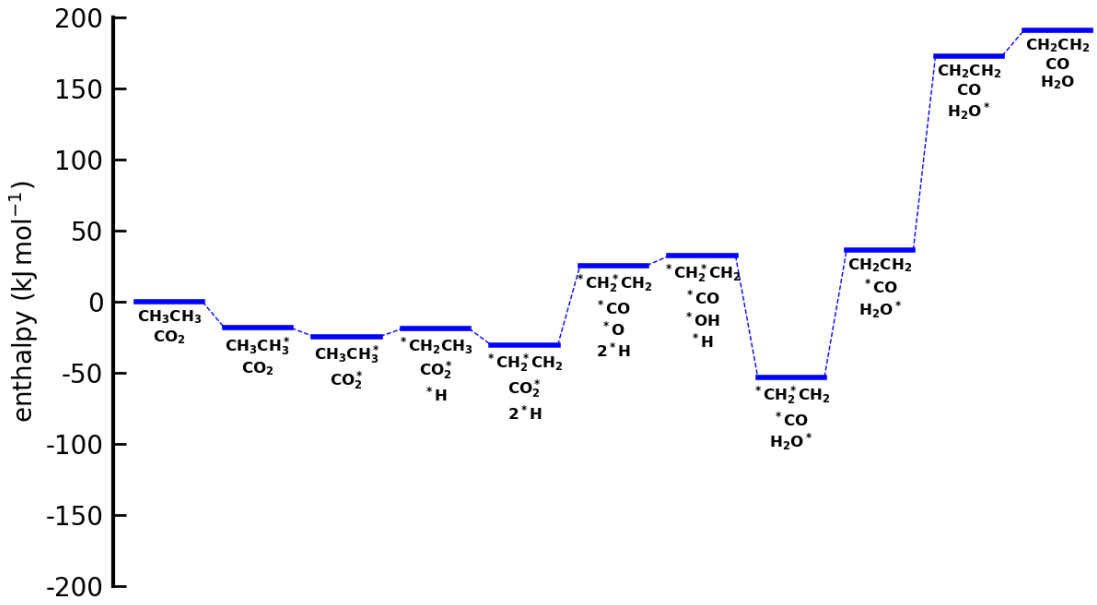
```
#Append the enthalpies of formation of the reference species to the enthalpies␣
 ↪of formation dictionary
enthalpies_relative_to_anchors.update(H_A_dict)
enthalpies_relative_to_anchors
```

[5]:
```
{'CO2': -103.51924771423779,
 'C2H4': 87.67622179997443,
 'C2H6X': -18.477466061631237,
 'XCH2CH3': -1.6041736579410335,
 'XO': 82.95473561961586,
 'XCH2XCH2': -1.7218897594761842,
 'XCO': -136.53682081535223,
 'CO2X': -109.45654867942778,
 'XOH': 78.37363221796271,
 'H2OX': -18.28310667649201,
 'XH': -11.363998907081388,
 'X': 0,
 'C2H6': 0,
 'H2O': 0,
 'CO': 0}
```

[6]:
```
plt.rcParams['figure.figsize'] = (14, 8)
plt.rcParams['axes.linewidth'] = 3
plt.rc('xtick', labelsize=20)
plt.rc('ytick', labelsize=20)
plt.rc('axes', labelsize=20)
plt.rc('legend', fontsize=20)
plt.rcParams['lines.markersize'] = 10
plt.rcParams['lines.linewidth'] = 4
plt.rcParams['xtick.direction'] = 'in'
plt.rcParams['ytick.direction'] = 'in'
plt.rcParams['xtick.major.size'] = 10
plt.rcParams['xtick.major.width'] = 2
plt.rcParams['ytick.major.size'] = 10
plt.rcParams['ytick.major.width'] = 2
plt.rcParams['legend.edgecolor'] = 'k'
plt.rcParams['axes.unicode_minus'] = False
plt.rcParams["legend.framealpha"] = 1
plt.rcParams['xtick.major.pad'] = 8
plt.rcParams['ytick.major.pad'] = 8
plt.rcParams['legend.handletextpad'] = 0.2
plt.rcParams['legend.columnspacing'] = 0.1
plt.rcParams['legend.labelspacing'] = 0.1
plt.rcParams['legend.title_fontsize'] = 14
plt.rcParams['axes.formatter.limits'] = (-3, 6)
```

```
gs = gridspec.GridSpec(nrows=1, ncols=1)
gs.update(wspace=0.5, hspace=0.5)
ax0 = plt.subplot(gs[0, 0])

ax0.set_ylim([-200,200])
ax0.set_xlim([-1,44])
ax0.get_xaxis().set_visible(False)
ax0.spines['right'].set_visible(False)
ax0.spines['top'].set_visible(False)
ax0.spines['bottom'].set_visible(False)
ax0.set_ylabel('$\mathrm{enthalpy\ (kJ\,mol^{-1})}$')


mechanism=({'C2H6':1,'CO2':1},
    {'C2H6X':1,'CO2':1},
    {'C2H6X':1,'CO2X':1},
    {'XCH2CH3':1,'CO2X':1,'XH':1},
    {'XCH2XCH2':1,'CO2X':1,'XH':2},
    {'XCH2XCH2':1,'XCO':1,'XO':1,'XH':2},
    {'XCH2XCH2':1,'XCO':1,'XOH':1,'XH':1},
    {'XCH2XCH2':1,'XCO':1,'H2OX':1},
    {'C2H4':1,'XCO':1,'H2OX':1},
    {'C2H4':1,'CO':1,'H2OX':1},
    {'C2H4':1,'CO':1,'H2O':1},
    )

tags=('$\mathbf{CH_3CH_3}$\n$\mathbf{CO_2}$',
      '$\mathbf{CH_3CH_3^*}$\n$\mathbf{CO_2}$',
      '$\mathbf{CH_3CH_3^*}$\n$\mathbf{CO_2^*}$',
      '$\mathbf{^*CH_2CH_3}$\n$\mathbf{CO_2^*}$\n$\mathbf{^*H}$',
      '$\mathbf{^*CH_2^*CH_2}$\n$\mathbf{CO_2^*}$\n$\mathbf{2 ^*H}$',
      '$\mathbf{^*CH_2^*CH_2}$\n$\mathbf{^*CO}$\n$\mathbf{^*O}$\n$\mathbf{2␣
 ↪^*H}$',
    ␣
 ↪'$\mathbf{^*CH_2^*CH_2}$\n$\mathbf{^*CO}$\n$\mathbf{^*OH}$\n$\mathbf{^*H}$',
      '$\mathbf{^*CH_2^*CH_2}$\n$\mathbf{^*CO}$\n$\mathbf{H_2O^*}$',
      '$\mathbf{CH_2CH_2}$\n$\mathbf{^*CO}$\n$\mathbf{H_2O^*}$',
      '$\mathbf{CH_2CH_2}$\n$\mathbf{CO}$\n$\mathbf{H_2O^*}$',
      '$\mathbf{CH_2CH_2}$\n$\mathbf{CO}$\n$\mathbf{H_2O}$',
)

# Function to compute the sum for each step in the mechanism
def compute_sum(vec,dictionary):
    return sum(vec[key] * value for key, value in dictionary.items())

def ediagram(ref_system):
    system = [compute_sum(ref_system,entry) for entry in mechanism]
```

```
    rel_enthalpies=np.zeros(len(system))
    for i, enthalpies in enumerate(system):
        start = i * 4
        end = start + 3
        ax0.plot((start, end), (enthalpies-system[0], enthalpies-system[0]),␣
 ↪linestyle='solid', color='b')
        if i>0:
            ax0.
 ↪plot((start-1,start),(system[i-1]-system[0],enthalpies-system[0]),linestyle='dashed',color=
 ↪linewidth=1)
        rel_enthalpies[i]=enthalpies-system[0]
    return rel_enthalpies

values=ediagram(enthalpies_relative_to_anchors)


for i in range(len(np.array(values).T)):
    start = i * 4
    ax0.text(start+1.5,np.array(values).
 ↪T[i]-5,tags[i],va='top',ha='center',size=12)
```



[ ]: