Thermo-Calc Workflow Guide

Extracting CALPHAD Data for Python Plotting

Anthony DiMascio

MSE 4381 | Honda CALPHAD Project | Spring 2026

---

1 File Organization

Create this folder structure **before** starting:

```
honda-calphad/
  data/
    thermocalc/
      raw/           <- Direct exports from TC
      processed/     <- Cleaned CSVs for Python
    literature/      <- Reference values from papers
  simulations/
    pycalphad/       <- Your existing Marimo notebook
```

1.1 File Naming Convention

```
[system]_[property]_[conditions].txt

Examples:
  cu2o_dGf_1273-1873K.txt      Gibbs energy of Cu2O formation
  al2o3_dGf_1273-1873K.txt     Gibbs energy of Al2O3 formation
  fe-cu_activity_1873K.txt     Cu activity in Fe at 1873K
  cu-al-o_isotherm_1873K.txt   Cu-Al-O phase diagram section
```

2 Calculation 1: Oxide Gibbs Energies (Ellingham Data)

Goal: Get accurate ΔG_f° vs T for each oxide to replace our approximations.

2.1 Setup

1. Open Thermo-Calc → `File` → `New Project`
2. Select database: **SSUB** (SGTE Substance Database)
3. Define system: For Cu_2O, select elements `Cu, O`

2.2 For Each Oxide

2.3 Running the Calculation

1. `Equilibrium Calculator` → `Property Diagram`
2. Axis variable: **T** from 1273 to 1873 K, step 25 K
3. Set composition to stoichiometric oxide (e.g., 66.67% Cu, 33.33% O for Cu_2O)
4. Add to plot: `GM(*)` (Gibbs energy of system)
5. `Perform` → `Calculate`

Oxide	Elements	Phase to check	Output file
Cu_2O	Cu, O	CU2O_S	`cu2o_dGf_1273-1873K.txt`
CuO	Cu, O	CUO_S	`cuo_dGf_1273-1873K.txt`
Al_2O_3	Al, O	CORUNDUM	`al2o3_dGf_1273-1873K.txt`
MgO	Mg, O	PERICLASE	`mgo_dGf_1273-1873K.txt`
SiO_2	Si, O	QUARTZ / CRISTOBALITE	`sio2_dGf_1273-1873K.txt`
TiO_2	Ti, O	RUTILE	`tio2_dGf_1273-1873K.txt`
FeO	Fe, O	WUSTITE	`feo_dGf_1273-1873K.txt`

2.4 Exporting

1. `Results` → `Table Renderer`
2. Select columns: `T`, `GM`
3. `File` → `Export Table` → `Tab-separated`
4. Save to: `data/thermocalc/raw/cu2o_dGf_1273-1873K.txt`

Repeat for each oxide. Total: 7 files.

3 Calculation 2: Cu Activity in Liquid Fe

Goal: How "active" is 0.3% Cu in molten steel?

3.1 Setup

1. Database: **TCFE** (Steels/Fe-alloys)
2. System: `Fe, Cu`
3. Phase: `LIQUID`

3.2 Calculation A: Activity vs Composition at 1873K

1. Property Diagram, axis: `X(Cu)` from 0 to 0.05 (0–5%)
2. Fixed: $T = 1873$ K
3. Plot: `AC(Cu)` (activity of Cu)
4. Export: `fe-cu_activity-vs-xcu_1873K.txt`

3.3 Calculation B: Activity vs Temperature at 0.3% Cu

1. Property Diagram, axis: `T` from 1773 to 1973 K
2. Fixed: `X(Cu)` = 0.003
3. Plot: `AC(Cu)`
4. Export: `fe-cu_activity-vs-T_xcu003.txt`

4 Calculation 3: Cu-Al-O Phase Diagram

Goal: What phases exist when Cu contacts Al_2O_3?

4.1 Setup

1. Database: **TCOX** (Oxides) — *check if available*
2. System: `Cu, Al, O`

4.2 Isothermal Section at 1873K

1. `Phase Diagram` → `Ternary`
2. Fix $T = 1873$ K, $P = 1$ atm
3. Generate diagram
4. Export tie-lines/phase boundaries as table
5. Save: `cu-al-o_isotherm_1873K.txt`

Note: If TCOX unavailable, check SSUB for limited oxide data.

5 Calculation 4: Cu Solubility in Al_2O_3

Goal: Maximum Cu that dissolves in alumina vs temperature.
This requires checking if the database models Cu solubility in the CORUNDUM phase.

1. System: Cu-Al-O, database TCOX
2. Fix Al_2O_3 composition (Al:O = 2:3)
3. Step temperature 1273–1873 K
4. Check: `X(CORUNDUM,Cu)` if available
5. Export: `al2o3_cu-solubility_1273-1873K.txt`

If not modeled: Note this as a limitation; the database may treat Al_2O_3 as a stoichiometric compound with no Cu solubility.

6 Data Processing

6.1 Raw Export Format

Thermo-Calc exports look like:

```
T       GM
1273    -245678.3
1298    -246012.1
...
```

6.2 Processing Script

Save as `data/process_tc_data.py`:

```
import numpy as np
import pandas as pd
from pathlib import Path

RAW = Path("thermocalc/raw")
OUT = Path("thermocalc/processed")
OUT.mkdir(exist_ok=True)
```

```
def process_oxide(filename, o2_factor, molar_mass_oxide):
    """Convert GM to dGf per mol O2 for Ellingham."""
    df = pd.read_csv(RAW/filename, sep='\t')
    df.columns = ['T_K', 'GM_J']

    # GM is in J/mol, convert to kJ/mol O2
    df['dGf_kJ_per_molO2'] = (df['GM_J'] / 1000) / o2_factor
    df['T_C'] = df['T_K'] - 273.15

    out_name = filename.replace('.txt', '_processed.csv')
    df.to_csv(OUT/out_name, index=False)
    print(f"Saved: {out_name}")

# Process each oxide
process_oxide('cu2o_dGf_1273-1873K.txt', 0.5, 143.09)
process_oxide('al2o3_dGf_1273-1873K.txt', 1.5, 101.96)
process_oxide('mgo_dGf_1273-1873K.txt', 0.5, 40.30)
process_oxide('sio2_dGf_1273-1873K.txt', 1.0, 60.08)
process_oxide('tio2_dGf_1273-1873K.txt', 1.0, 79.87)
process_oxide('feo_dGf_1273-1873K.txt', 0.5, 71.84)
```

7 Plotting in Python

Add to your Marimo notebook or create new script:

```
import pandas as pd
import matplotlib.pyplot as plt
from pathlib import Path

DATA = Path("data/thermocalc/processed")

# Load all oxides
oxides = {
    'Cu2O':  ('cu2o_dGf_1273-1873K_processed.csv',  '#0077BB', '-'),
    'Al2O3': ('al2o3_dGf_1273-1873K_processed.csv', '#AA3377', ':'),
    'MgO':   ('mgo_dGf_1273-1873K_processed.csv',   '#009988', '-'),
    'SiO2':  ('sio2_dGf_1273-1873K_processed.csv',  '#CC3311', '--'),
    'TiO2':  ('tio2_dGf_1273-1873K_processed.csv',  '#E69F00', ':'),
    'FeO':   ('feo_dGf_1273-1873K_processed.csv',   '#EE7733', '--'),
}

fig, ax = plt.subplots(figsize=(10, 7))

for name, (file, color, ls) in oxides.items():
    df = pd.read_csv(DATA/file)
    ax.plot(df['T_C'], df['dGf_kJ_per_molO2'],
            color=color, ls=ls, lw=2.5, label=name)

ax.set_xlabel('Temperature (C)')
ax.set_ylabel('dG (kJ/mol O2)')
ax.set_title('Ellingham Diagram (Thermo-Calc Data)')
ax.legend()
```

```
ax.grid(True, alpha=0.3)
plt.tight_layout()
plt.savefig('ellingham_thermocalc.png', dpi=150)
```

8 Checklist

Task	Done
Create folder structure	☐
Export Cu_2O ΔG_f vs T	☐
Export Al_2O_3 ΔG_f vs T	☐
Export MgO ΔG_f vs T	☐
Export SiO_2 ΔG_f vs T	☐
Export TiO_2 ΔG_f vs T	☐
Export FeO ΔG_f vs T	☐
Export Cu activity in Fe vs X(Cu)	☐
Export Cu activity in Fe vs T	☐
Cu-Al-O isothermal section (if TCOX available)	☐
Run processing script	☐
Generate new Ellingham plot	☐
Update Marimo notebook with TC data	☐

After completing, commit data to GitHub: `git add data/ && git commit -m "Add TC data" && git push`