

# pycalphad and ESPEI Workshop

ESPEI

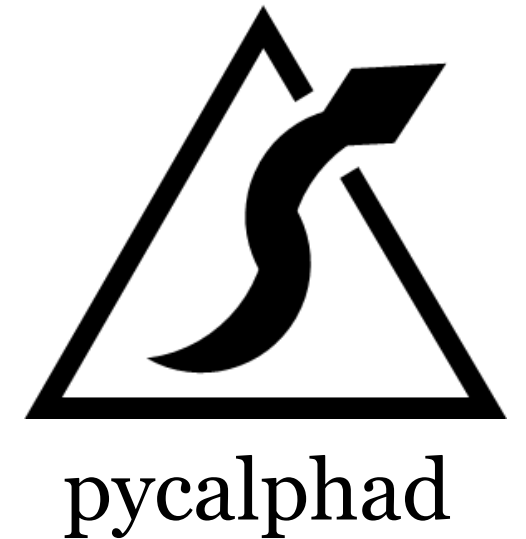
CALPHAD XLVII



**PennState**

# Why develop CALPHAD software tools?

- Better science
  - Develop new thermodynamic models
  - Improve and share methods as a community
  - Scripting: get more work done and be able to reproduce it
- Integration
  - Use standard tools in a huge Python ecosystem
  - Create calculation and data pipelines
  - Traverse length and time scales
  - Hierarchical models, machine learning
- Free
  - Freedom: share your work (or don't)
  - Cost: lower the barrier for entry



PennState

Why develop CALPHAD software tools?

# Calculation software is NOT the whole picture!

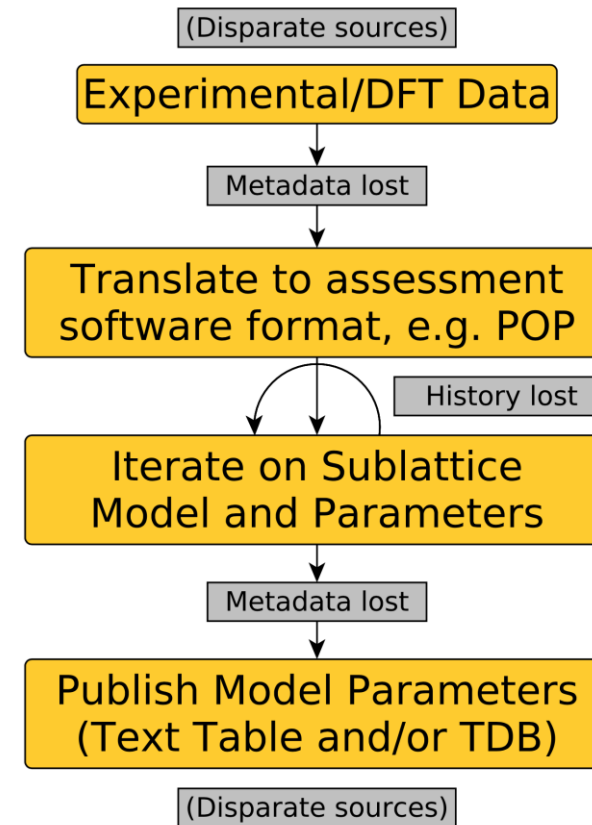
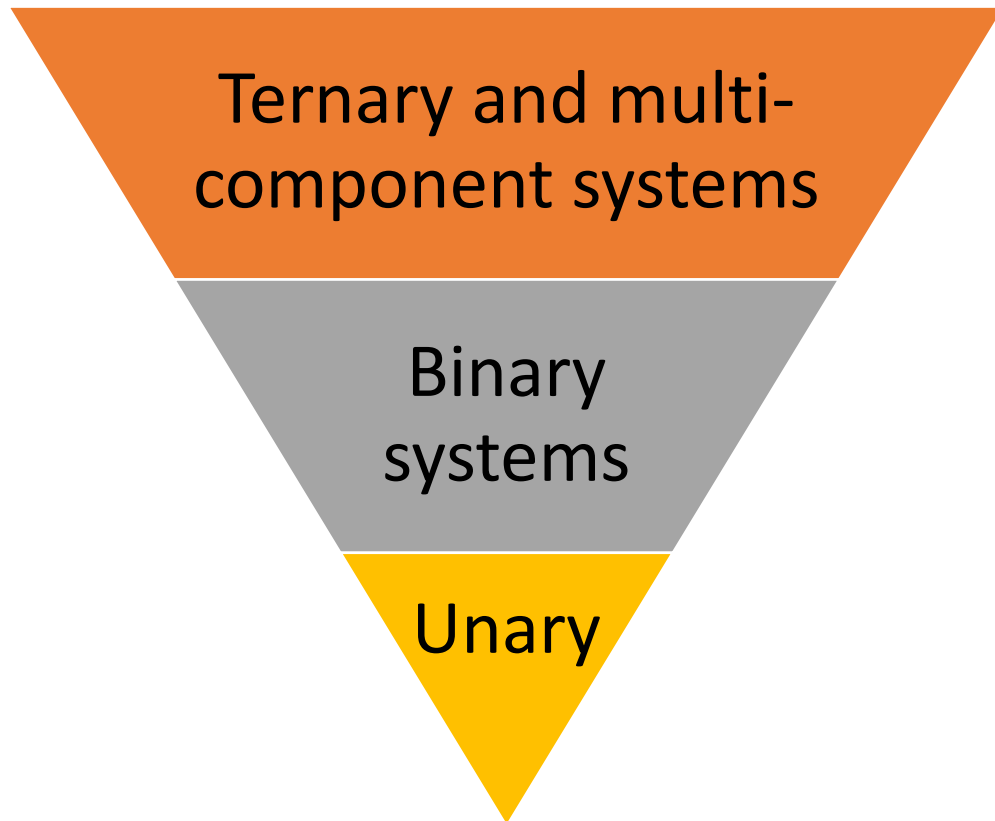
Databases and custom models come from somewhere

- Free
  - Freedom: share your work (or don't)
  - Cost: lower the barrier for entry



PennState

# Model for traditional database development



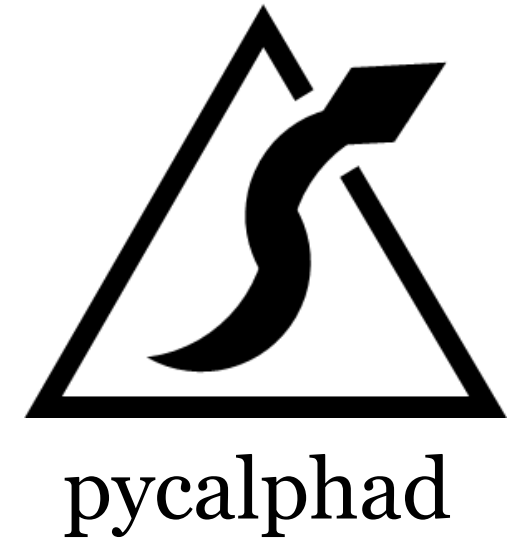
Otis (2016).



PennState

# Established methods are effective, but have pain points

- Data formats can lead to ambiguity
  - “*Should the experiment be `T` or `X`?*”
  - How long does it take to understand a file someone else created?
- Critically assessed data is closely guarded
- Maintenance and re-optimization prevents new models from being tested and adopted =>  $n^{th}$  generation of databases



# We need better modeling tools too

- DRY principle: Don't Repeat Yourself
- We need better tools for rapid (re)assessment of new models
- Our tools should tell us
  - Where our models might make bad predictions
  - How we should spend our time doing experiments and calculations



# Solution: ESPEI

Extensible Self-optimizing Phase Equilibria Infrastructure

## 2 step approach:

1. Rapidly generate model parameters
2. Automatically optimize parameters simultaneously

➔ Quantify uncertainty to make us better modelers



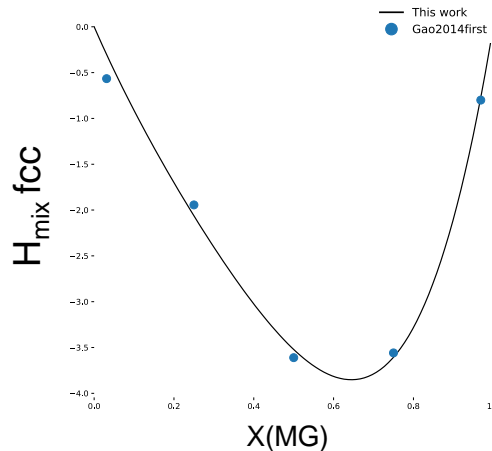
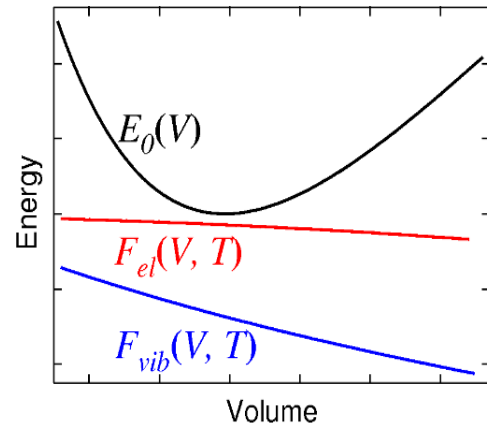
# Parameter selection

- Goal: Maximize model predictiveness, minimize model complexity
- Given a model:  $a + bT + cT \ln T + dT^{-1} + \sum_i e_i T^i$
- Question:  
*How many parameters from this model should be used for each CALPHAD-type parameter?*
- Traditional solution
  - Introduce parameters one at a time
  - Use domain knowledge
  - Rely on experimental information with increasing amounts from DFT
- ESPEI approach:
  - Rely heavily on DFT data
  - Use derivatives of Gibbs energy to simplify the problem
  - Apply corrected Akaike information criterion





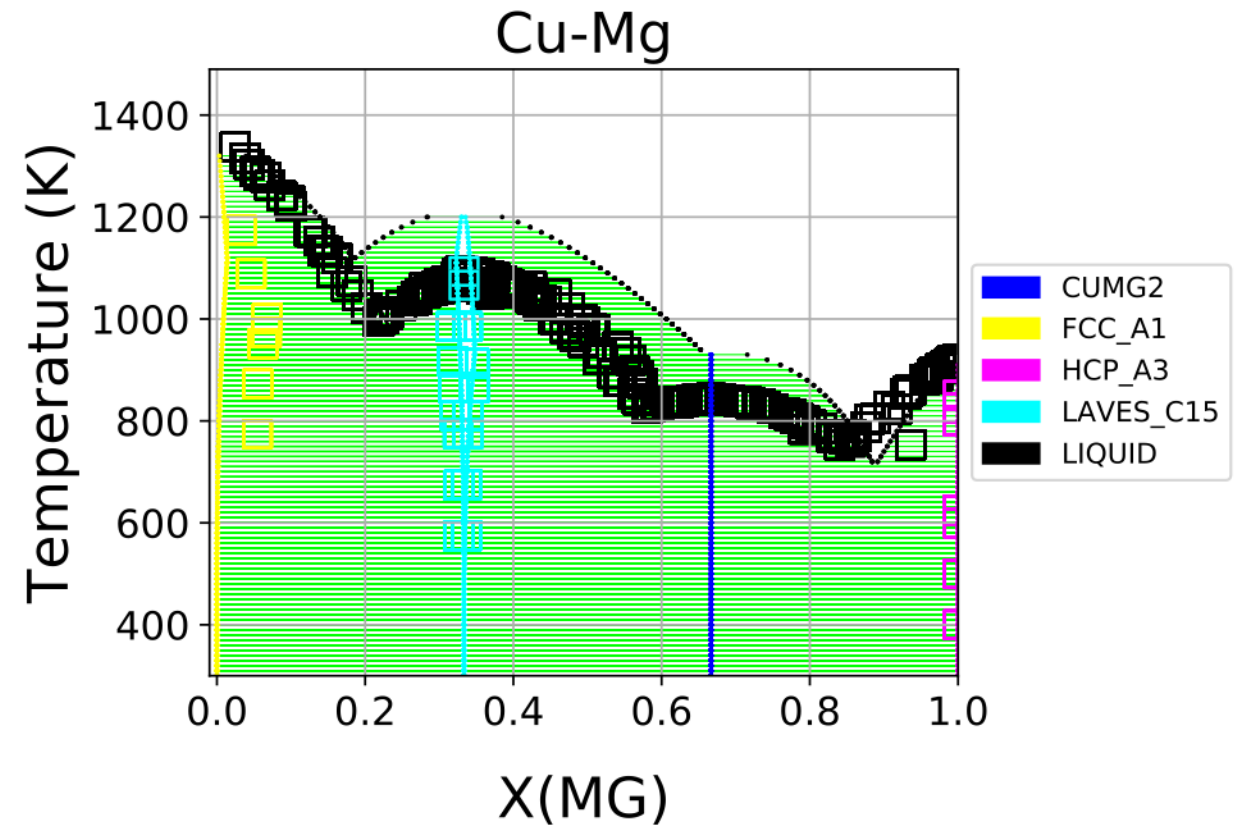
# Parameter selection example: Cu-Mg phase diagram from single phase data



**ESPEI**  
Parameter Selection



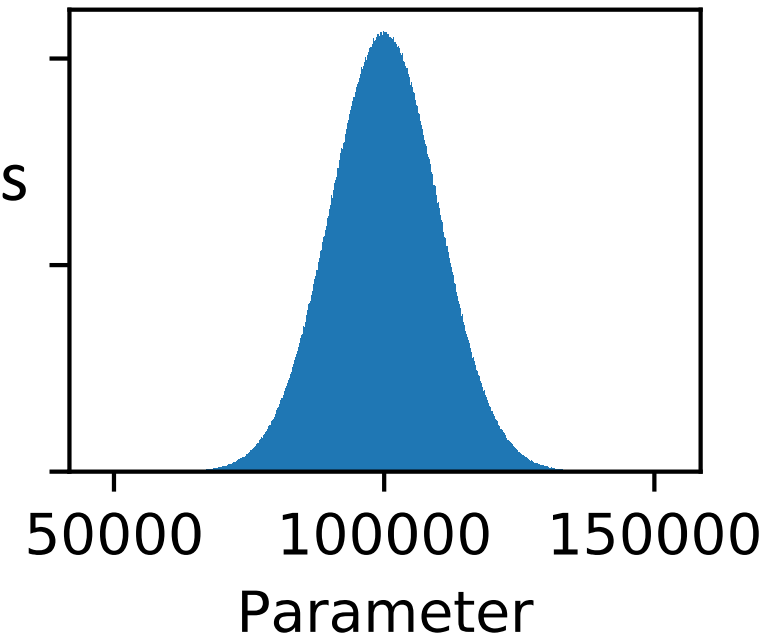
Run time:  $\approx 3$  seconds



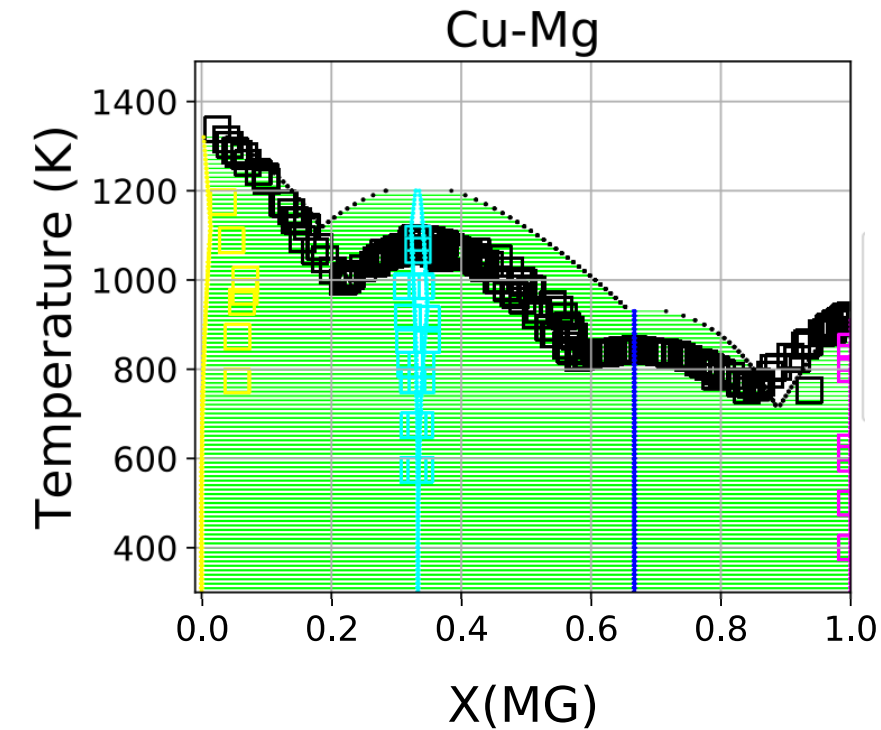
PennState

# Ensemble Markov Chain Monte Carlo

- Goal of MCMC: find probability distribution for parameters
- Bayesian statistics method
- Start from a prior that is based on the current TDB
- Construct a Markov chain of possible parameter values
- For more on this come to our demo or Brandon's talk

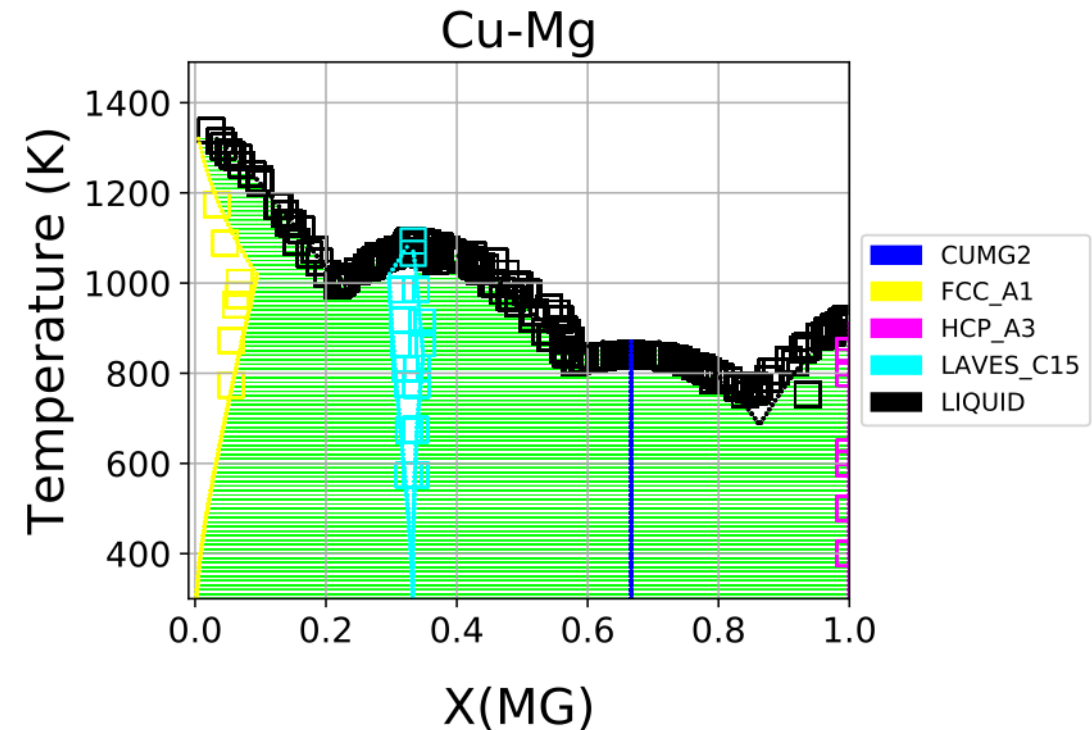


# MCMC example: Cu-Mg phase diagram



**ESPEI**  
**MCMC**

Run time:  $\approx 8$  hours  
on a laptop



PennState

# ESPEI datasets

- Critically assessed data is key to obtaining good results
- Data is digitized in JSON, a machine and human readable format

```
1 {
2   "components": ["CU", "MG", "VA"],
3   "phases": ["LAVES_C15"],
4   "solver": {
5     "mode": "manual",
6     "sublattice_site_ratios": [2, 1],
7     "sublattice_configurations": [
8       ["CU", "MG"],
9       ["MG", "CU"],
10      ["MG", "MG"],
11      ["CU", "CU"]
12    ],
13   "conditions": {
14     "P": 101325,
15     "T": 298.15
16   },
17   "output": "HM_FORM",
18   "values": [[[-15720, 34720, 7000, 15500]]],
19   "reference": "Zhou2007",
20   "comment": "DFT reported in S. H. Zhou et al., J. Phase Equilib.
```

Single phase data

```
1 {
2   "components": ["CU", "MG", "VA"],
3   "phases": ["LIQUID", "FCC_A1", "LAVES_C15"],
4   "conditions": {
5     "P": 101325,
6     "T": [773.85, 872.75, 951.15, 973.15, 1001.35, 1088.45, 1174.25]
7   },
8   "broadcast_conditions": false,
9   "output": "ZPF",
10  "values": [
11    [
12      ["FCC_A1", ["MG"], [0.0545706]], ["LAVES_C15", ["MG"], [null]],
13      ["FCC_A1", ["MG"], [0.0551807]], ["LAVES_C15", ["MG"], [null]],
14      ["FCC_A1", ["MG"], [0.0621264]], ["LAVES_C15", ["MG"], [null]],
15      ["FCC_A1", ["MG"], [0.0666104]], ["LAVES_C15", ["MG"], [null]],
16      ["FCC_A1", ["MG"], [0.0679314]], ["LIQUID", ["MG"], [null]],
17      ["FCC_A1", ["MG"], [0.0459052]], ["LIQUID", ["MG"], [null]],
18      ["FCC_A1", ["MG"], [0.0329389]], ["LIQUID", ["MG"], [null]]
19    ],
20    "reference": "Jones1931",
21    "comment": "solubility lines from Fig A, converted to atomic percent"
22  ]
23 }
```

ZPF data

# ESPEI datasets

- Some digitized data freely available at
  - <https://github.com/PhasesResearchLab/ESPEI-datasets>
  - Contributions, fixes and suggestions are welcome
- Specification and guide for writing ESPEI datasets
  - [http://espei.org/en/latest/input\\_data.html](http://espei.org/en/latest/input_data.html)
- Prototype digitization of POP files to Python dictionaries
  - Used at Citrine Informatics
  - Conversion to ESPEI JSON coming soon
  - <https://github.com/PhasesResearchLab/popparsing>
  - <http://popparsing.readthedocs.io/>
  - We'd love suggestions and for you to report failures



# Description of the phase models

```
1  {
2    "components": ["CU", "MG", "VA"],
3    "refdata": "SGTE91",
4    "phases": {
5      "LIQUID" : {
6        "sublattice_model": [["CU", "MG"]],
7        "sublattice_site_ratios": [1]
8      },
9      "CUMG2": {
10       "sublattice_model": [["CU"], ["MG"]],
11       "sublattice_site_ratios": [1, 2]
12     },
13     "FCC_A1": {
14       "sublattice_model": [["CU", "MG"], ["VA"]],
15       "sublattice_site_ratios": [1, 1]
16     },
17     "HCP_A3": {
18       "sublattice_site_ratios": [1, 0.5],
19       "sublattice_model": [["CU", "MG"], ["VA"]]
20     },
21     "LAVES_C15": {
22       "sublattice_site_ratios": [2, 1],
23       "sublattice_model": [["CU", "MG"], ["CU", "MG"]]
24     }
25   }
26 }
```



# Running ESPEI: File based inputs

- Can do parameter generation, mcmc, or both
- Parameter generation
  - SGTE91 reference state provided
  - Easy to add other unary reference states
  - Choose an excess model (currently linear only)
- MCMC
  - Restart from previous run
  - Use MPI on a cluster
  - Many more customization options...

Example of minimal input for a full run

system:

```
phase_models: phases.json
```

```
datasets: my-input-data
```

```
generate_parameters:
```

```
excess_model: linear
```

```
ref_state: SGTE91
```

```
mcmc:
```

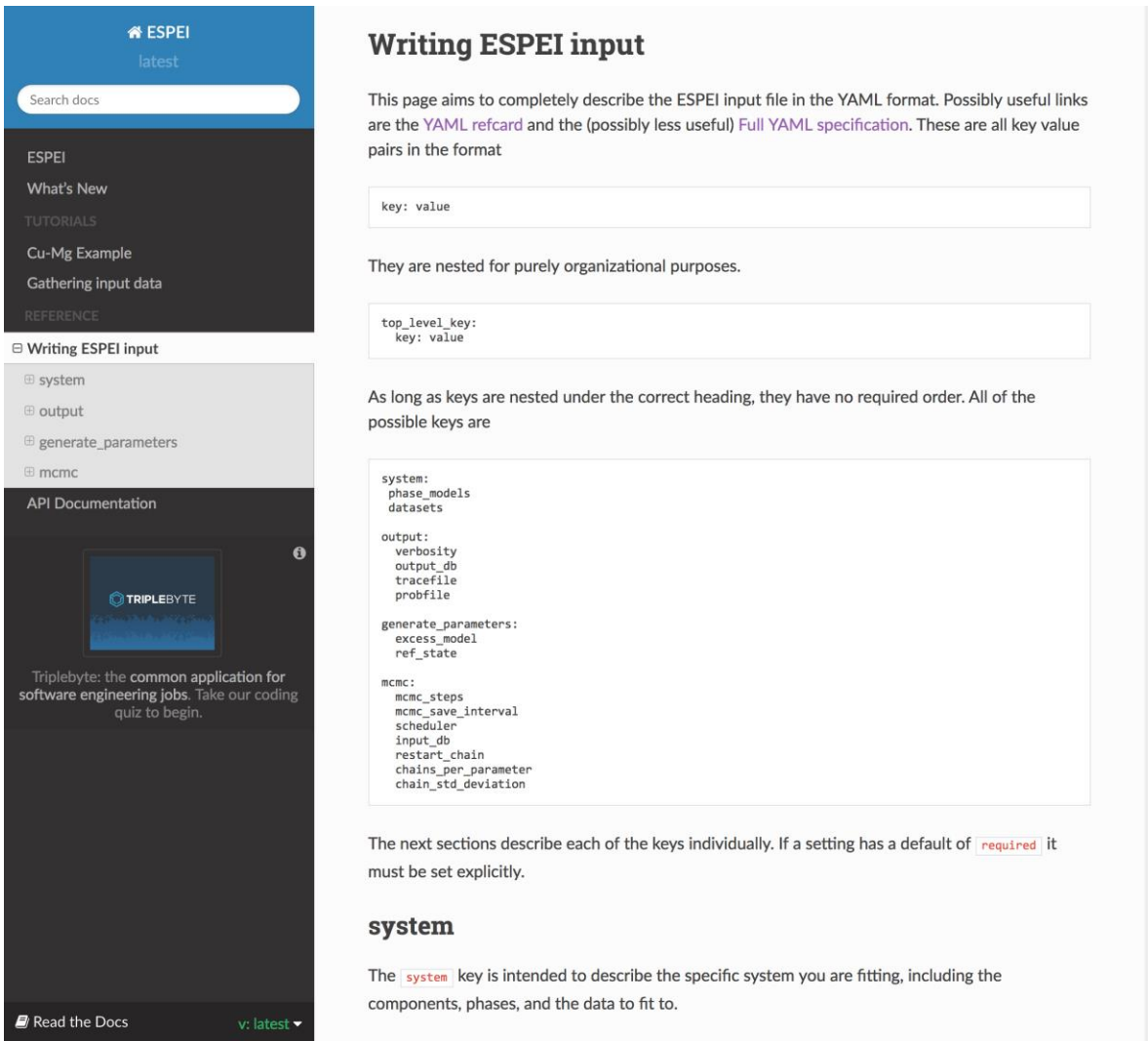
```
mcmc_steps: 1000
```



PennState

# Running ESPEI: File based inputs

## Fully documented online



The screenshot shows the ESPEI documentation website. The left sidebar contains a navigation menu with links to 'ESPEI', 'What's New', 'TUTORIALS', 'Cu-Mg Example', 'Gathering Input data', and 'REFERENCE'. Below this is a section titled 'Writing ESPEI input' with sub-links for 'system', 'output', 'generate\_parameters', and 'mcmc'. The main content area is titled 'Writing ESPEI input' and explains that the page describes the ESPEI input file in the YAML format. It provides examples of key-value pairs in YAML format, such as 'key: value' and 'top\_level\_key: key: value'. It also lists possible keys for the 'system' section, including 'phase\_models', 'datasets', 'output', 'verbosity', 'output\_db', 'tracefile', 'probfile', 'generate\_parameters', 'excess\_model', 'ref\_state', 'mcmc', 'mcmc\_steps', 'mcmc\_save\_interval', 'scheduler', 'input\_db', 'restart\_chain', 'chains\_per\_parameter', and 'chain\_std\_deviation'. The bottom of the page features a 'Triplebyte' logo and a call to action to 'Read the Docs'.

**Writing ESPEI input**

This page aims to completely describe the ESPEI input file in the YAML format. Possibly useful links are the [YAML refcard](#) and the (possibly less useful) [Full YAML specification](#). These are all key value pairs in the format

```
key: value
```

They are nested for purely organizational purposes.

```
top_level_key:
  key: value
```

As long as keys are nested under the correct heading, they have no required order. All of the possible keys are

```
system:
  phase_models
  datasets

output:
  verbosity
  output_db
  tracefile
  probfile

generate_parameters:
  excess_model
  ref_state

mcmc:
  mcmc_steps
  mcmc_save_interval
  scheduler
  input_db
  restart_chain
  chains_per_parameter
  chain_std_deviation
```

The next sections describe each of the keys individually. If a setting has a default of `required` it must be set explicitly.

**system**

The `system` key is intended to describe the specific system you are fitting, including the components, phases, and the data to fit to.

<https://espei.org>



PennState