```python
import numpy as np
import matplotlib.pyplot as plt
import os
import h5py, hickle
```

0.0.1 Calculating canonical ensemble averages

```python
class CanonicalEnsemble:
    def __init__(self, Es, gs, name):
        self.Es = Es
        self.gs = gs
        self.name = name
    def Z(self, β):
        return np.sum(self.gs * np.exp(-β * self.Es))
    def average(self, f, β):
        return np.sum(f(self) * self.gs * np.exp(-β * self.Es)) / self.Z(β)
    def energy(self, β):
        return self.average(lambda ens: ens.Es, β)
    def energy2(self, β):
        return self.average(lambda ens: ens.Es**2, β)
    def heat_capacity(self, β):
        return self.energy2(β) - self.energy(β)**2
    def free_energy(self, β):
        return -np.log(self.Z(β)) / β
    def entropy(self, β):
        return β * self.energy(β) + np.log(self.Z(β))
```

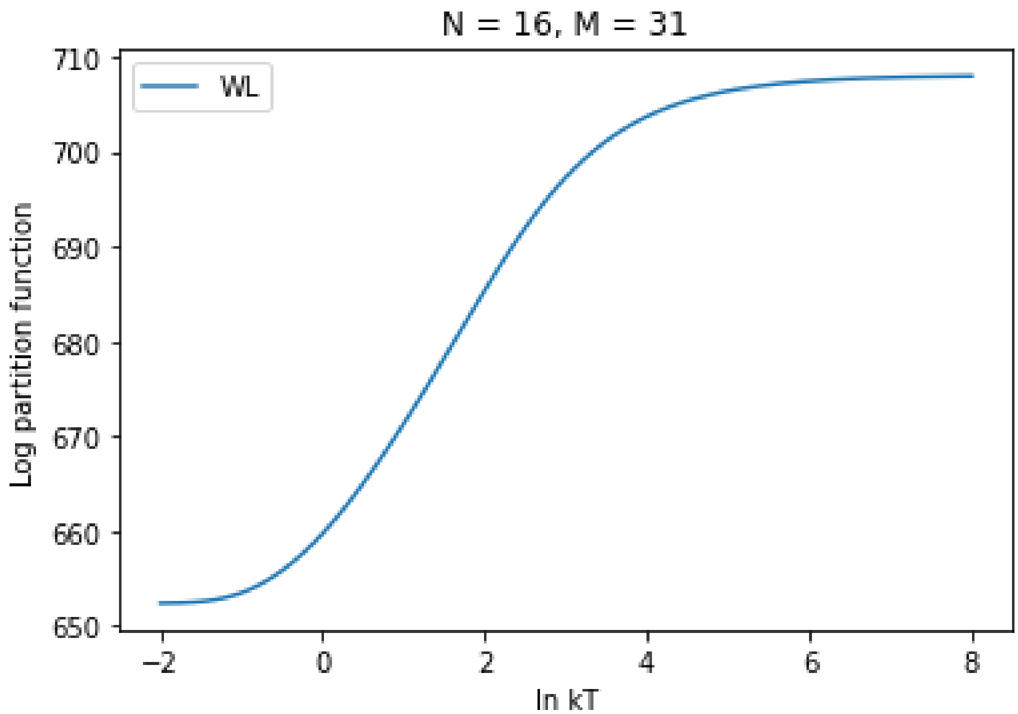
```python
with h5py.File('data/wlresults-image-6_4jpbol.hdf5', 'r') as f:
    results = hickle.load(f)
```

```python
N, M = results['N'], results['M']
wlEs, wlgs = results['sEs'][:-1], np.exp(results['sS'])
```

```python
βs = [np.exp(k) for k in np.linspace(-8, 2, 500)]
wlens = CanonicalEnsemble(wlEs, wlgs, 'WL') # Wang-Landau results
# xens = CanonicalEnsemble(Es, gs, 'Exact') # Exact
# ensembles = [wlens, xens]
ensembles = [wlens]
```

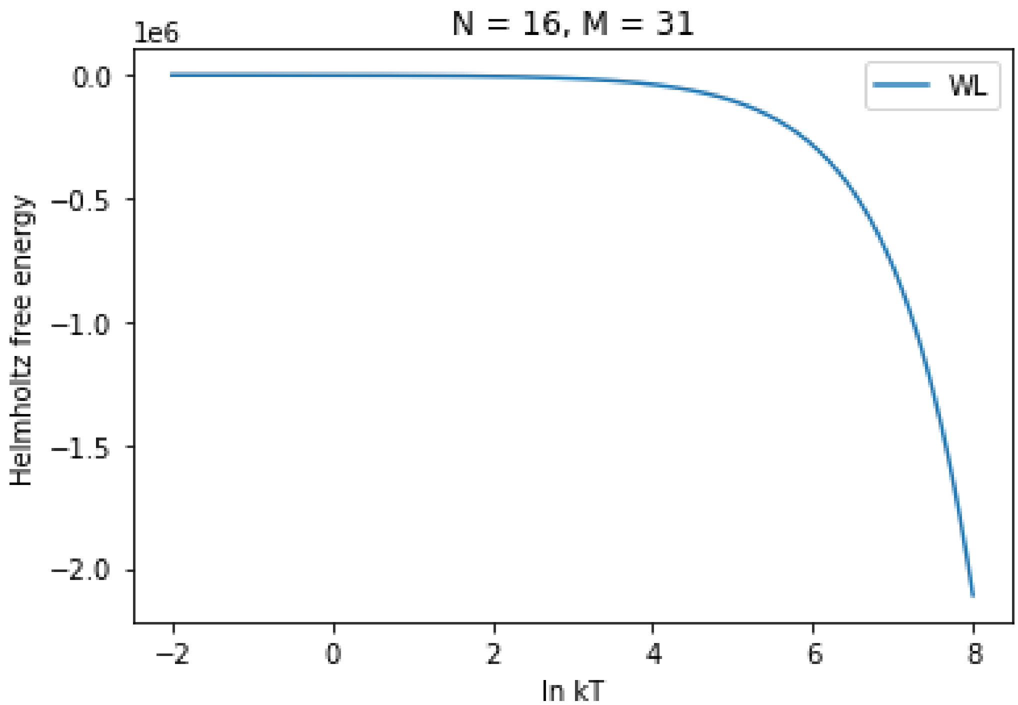
Partition function

```python
for ens in ensembles:
    plt.plot(-np.log(βs), np.log(np.vectorize(ens.Z)(βs)), label=ens.name)
plt.xlabel("ln kT")
plt.ylabel("Log partition function")
plt.title('N = {}, M = {}'.format(N, M))
plt.legend();
```



Helmholtz free energy

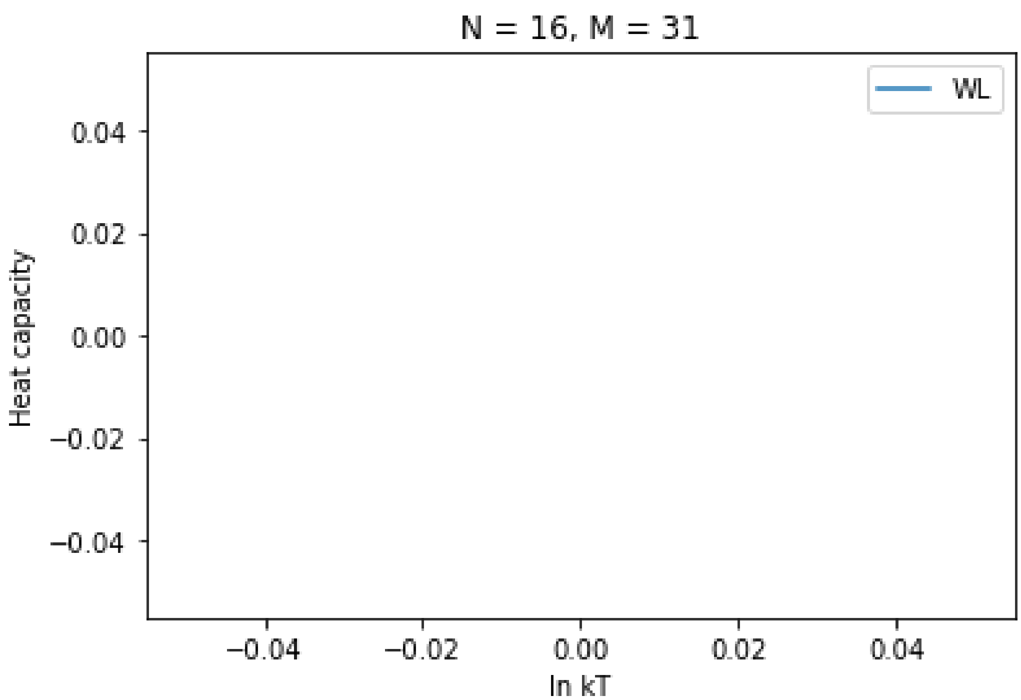
```
for ens in ensembles:
    plt.plot(-np.log(βs), np.vectorize(ens.free_energy)(βs), label=ens.name)
plt.xlabel("ln kT")
plt.ylabel("Helmholtz free energy")
plt.title('N = {}, M = {}'.format(N, M))
plt.legend();
```



Heat capacity

```
for ens in ensembles:
    plt.plot(-np.log(βs), np.vectorize(ens.heat_capacity)(βs), label=ens.name)
plt.xlabel("ln kT")
plt.ylabel("Heat capacity")
plt.title('N = {}, M = {}'.format(N, M))
plt.legend();
```

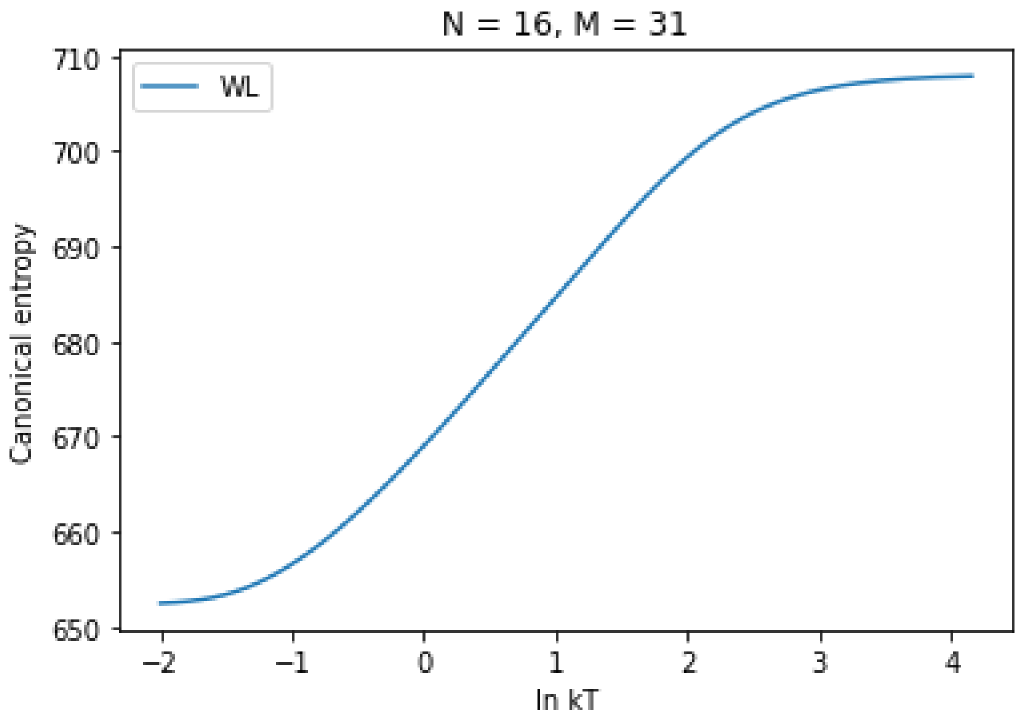
```
<ipython-input-2-8c224f54cf97>:9: RuntimeWarning: overflow encountered in multiply
  return np.sum(f(self) * self.gs * np.exp(-β * self.Es)) / self.Z(β)
/usr/lib/python3.8/site-packages/numpy/core/fromnumeric.py:90: RuntimeWarning: overflow encountered in r
  return ufunc.reduce(obj, axis, dtype, out, **passkwargs)
<ipython-input-2-8c224f54cf97>:15: RuntimeWarning: invalid value encountered in double_scalars
  return self.energy2(β) - self.energy(β)**2
<ipython-input-2-8c224f54cf97>:9: RuntimeWarning: invalid value encountered in multiply
  return np.sum(f(self) * self.gs * np.exp(-β * self.Es)) / self.Z(β)
```



Entropy

```python
for ens in ensembles:
    plt.plot(-np.log(βs), np.vectorize(ens.entropy)(βs), label=ens.name)
plt.xlabel("ln kT")
plt.ylabel("Canonical entropy")
plt.title('N = {}, M = {}'.format(N, M))
plt.legend();
```

```
/usr/lib/python3.8/site-packages/numpy/core/fromnumeric.py:90: RuntimeWarning: overflow encountered in
  return ufunc.reduce(obj, axis, dtype, out, **passkwargs)
```


N = 16, M = 31
WL
Canonical entropy
ln kT
710
700
690
680
670
660
650
−2
−1
0
1
2
3
4