Conclusion: Can increase default `EDIFFG` to `-0.025` for ~30-40% speed boost

In [ ]:

```

```

Wrote this script to pull the energies and forces from VASP calculations:

In [18]:

```
!cat E_and_F_vs_step_from_vr.py
```

```
#!/usr/bin/env python

import sys
import numpy as np
import pandas as pd
from pymatgen.io.vasp.outputs import Vasprun

if len(sys.argv) > 1:
    vr_path = sys.argv[1]
else:
    vr_path = "vasprun.xml"

vr = Vasprun(vr_path)
# energy(sigma->0) is e_0_energy
# max norm force in eV/A, breaks when less than EDIFFG (=-0.01 here)

df_dict = {"Energy": [], "Max_Norm_Force": []}
for step_dict in vr.ionic_steps:
    df_dict["Energy"].append(step_dict["e_0_energy"])
    forces_array = step_dict["forces"]
    norms_array = np.apply_along_axis(np.linalg.norm, 1, forces_array)
    df_dict["Max_Norm_Force"].append(np.max(norms_array))

df = pd.DataFrame(df_dict)
df.to_json("E_and_F_vs_step_df.json")
```

V_{Cd}^0 ("The Classic")

In [63]:

```
import os
import matplotlib.pyplot as plt
import pandas as pd
plt.style.use("../Packages/publication_style.mplstyle")

json_dir = "Rattling_Tests/vac_1_Cd_0/BDM/"
e_ground = -206.478
```

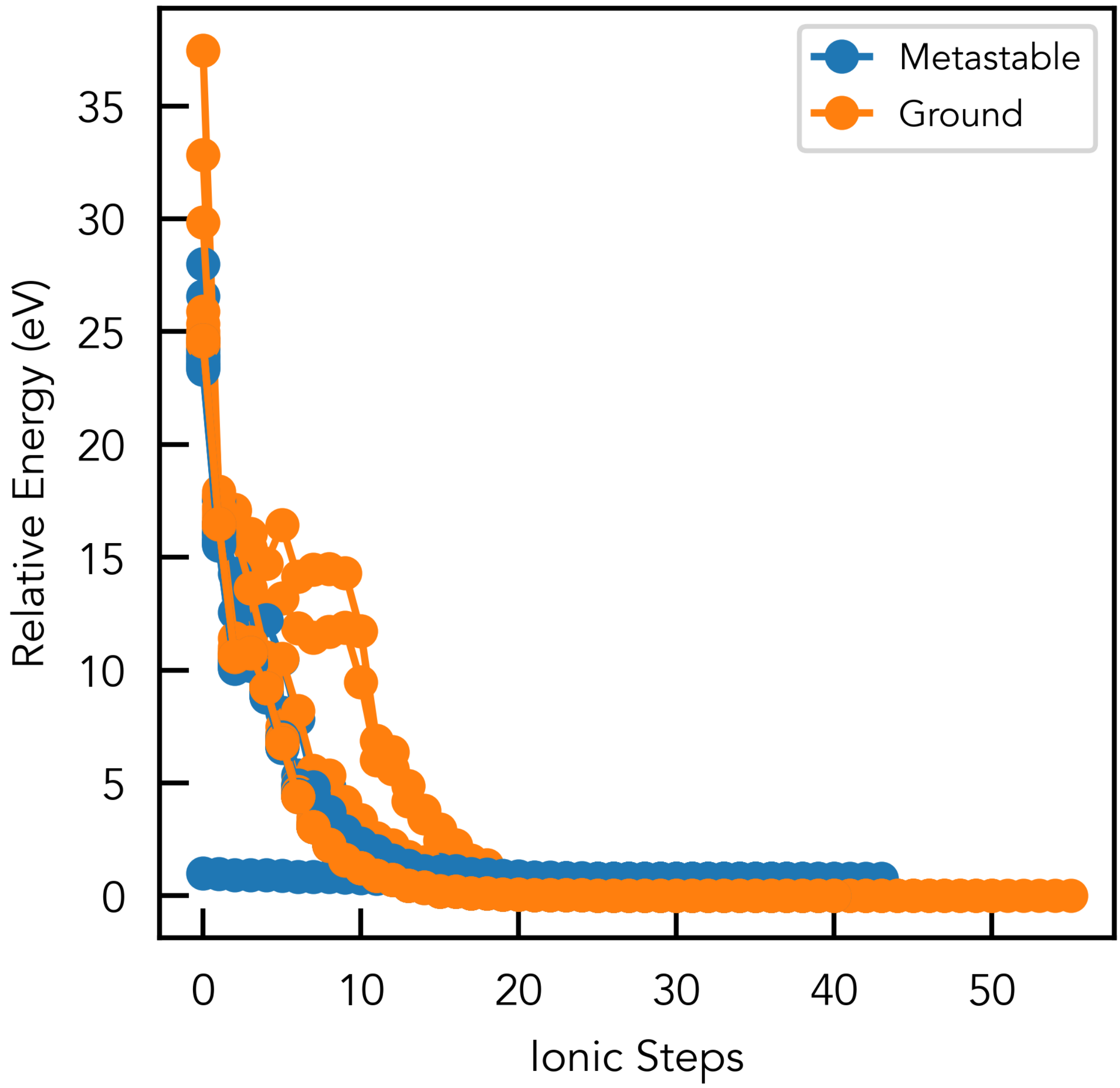
In [64]:

```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Energy - e_ground, color=c, marker="o")  # groundstate en
ax.plot(df.Energy - e_ground, label="Metastable", color="C0", marker="o")
ax.plot(df.Energy - e_ground, label="Ground", color="C1", marker="o")
ax.plot(df.Energy - e_ground, color=c, marker="o")
#ax.set_xlim(0,30)
ax.legend()
```

```
ax.set_xlabel("Ionic Steps")
ax.set_ylabel("Relative Energy (eV)")
```

Out[64]: Text(0, 0.5, 'Relative Energy (eV)')

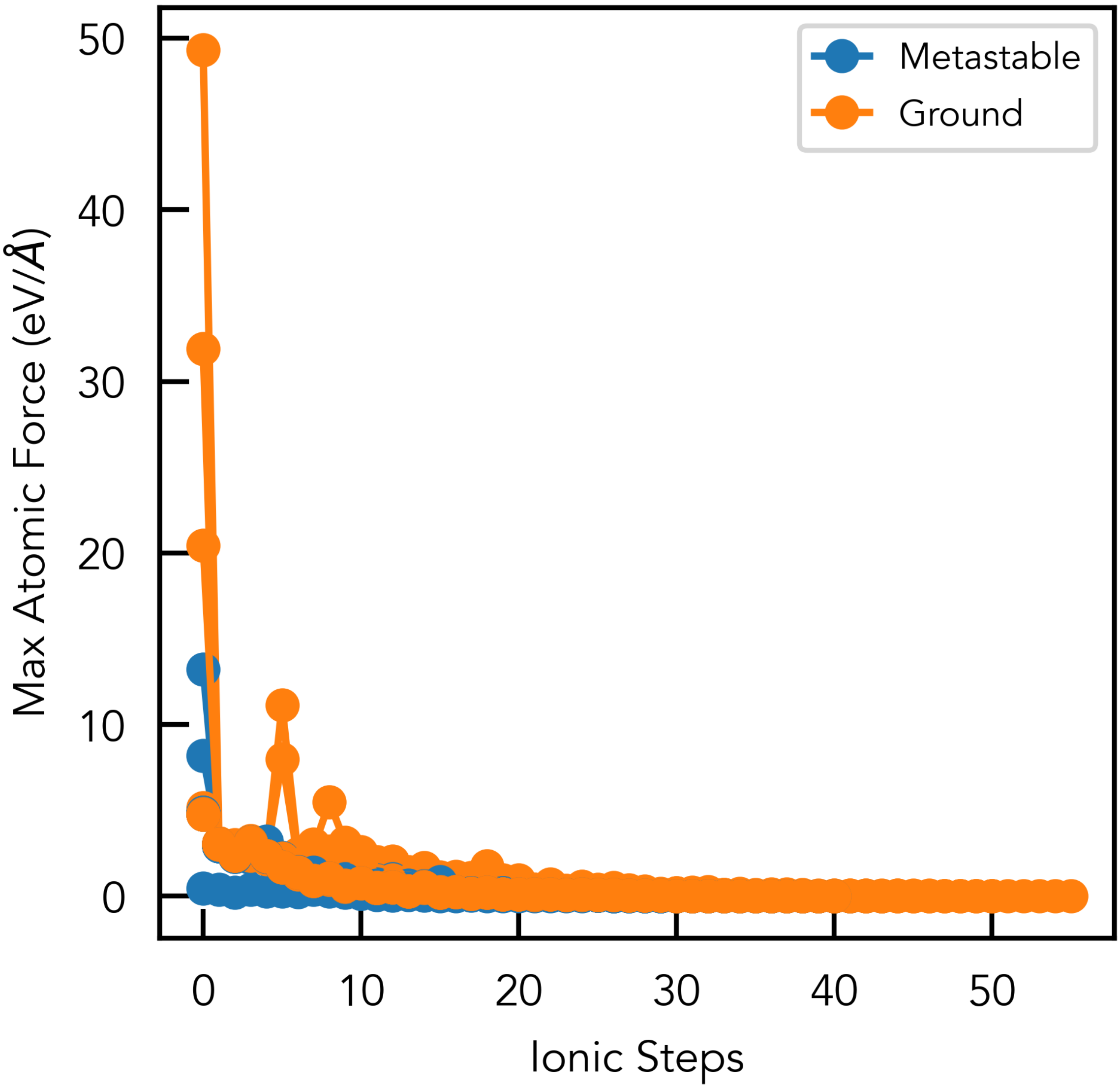


Ok so no clear correlation between ground/metastable and energy vs step. Quasi-flat line is unperturbed as you'd imagine

In [65]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.plot(df.Max_Norm_Force, label="Metastable", color="C0", marker="o")
ax.plot(df.Max_Norm_Force, label="Ground", color="C1", marker="o")
ax.plot(df.Max_Norm_Force, color=c, marker="o")
#ax.set_xlim(0,10)
ax.legend()
ax.set_xlabel("Ionic Steps")
ax.set_ylabel("Max Atomic Force (eV/$\AA$)")
```

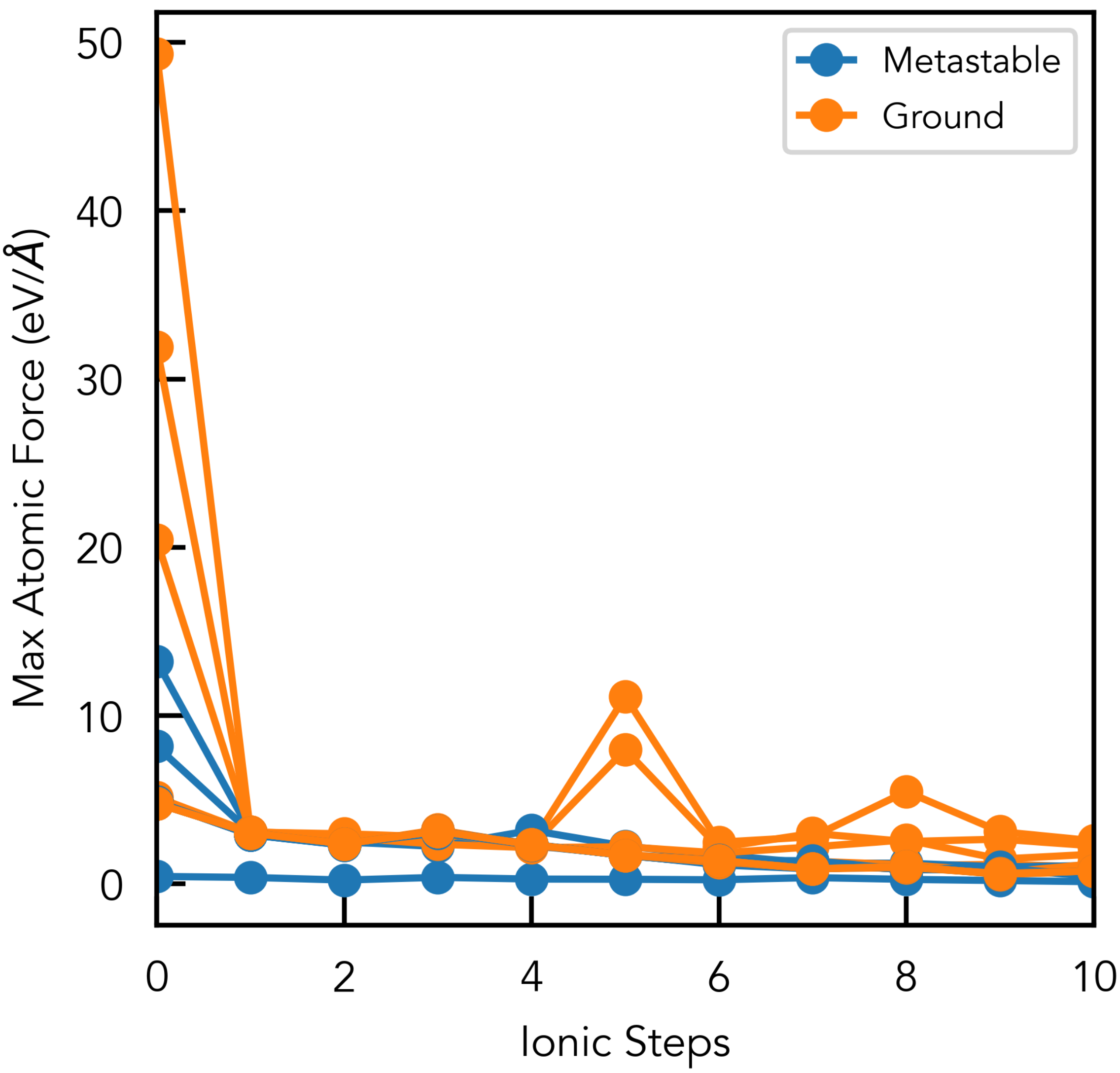
Out[65]: Text(0, 0.5, 'Max Atomic Force (eV/$\\AA$)')



In [66]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.plot(df.Max_Norm_Force, label="Metastable", color="C0", marker="o")
ax.plot(df.Max_Norm_Force, label="Ground", color="C1", marker="o")
ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.set_xlim(0,10)
ax.legend()
ax.set_xlabel("Ionic Steps")
ax.set_ylabel("Max Atomic Force (eV/$\AA$)")
```

Out[66]: Text(0, 0.5, 'Max Atomic Force (eV/$\\AA$)')



Interesting, many of the ground-state ones actually have the highest initial force... But again no real clear correlation

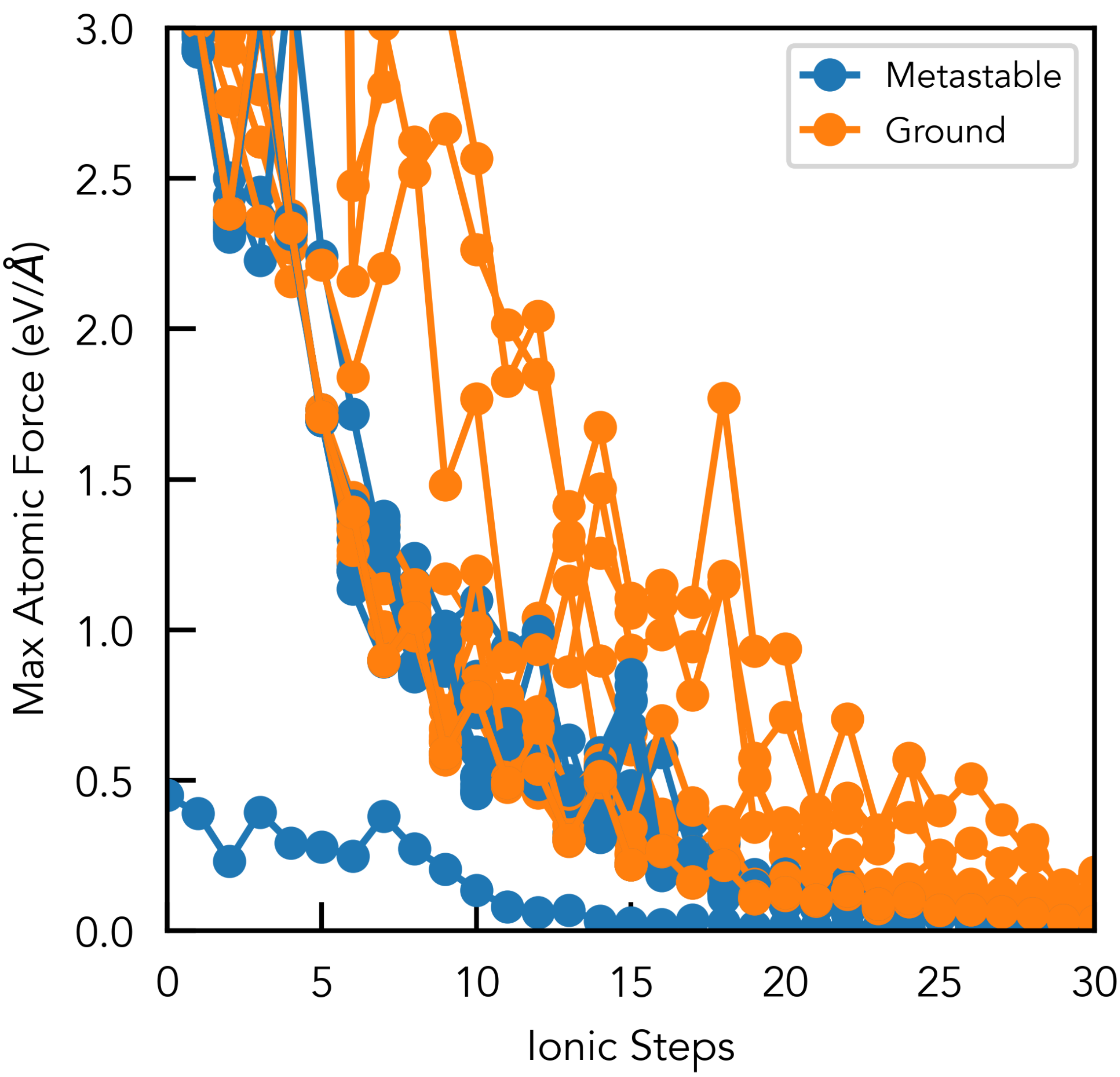
In [67]:

```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.plot(df.Max_Norm_Force, label="Metastable", color="C0", marker="o")
ax.plot(df.Max_Norm_Force, label="Ground", color="C1", marker="o")
ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.set_xlim(0,30)
ax.set_ylim(0,3)
ax.legend()
ax.set_xlabel("Ionic Steps")
ax.set_ylabel("Max Atomic Force (eV/$\AA$)")
```

Out[67]:

```
Text(0, 0.5, 'Max Atomic Force (eV/$\\AA$)')
```



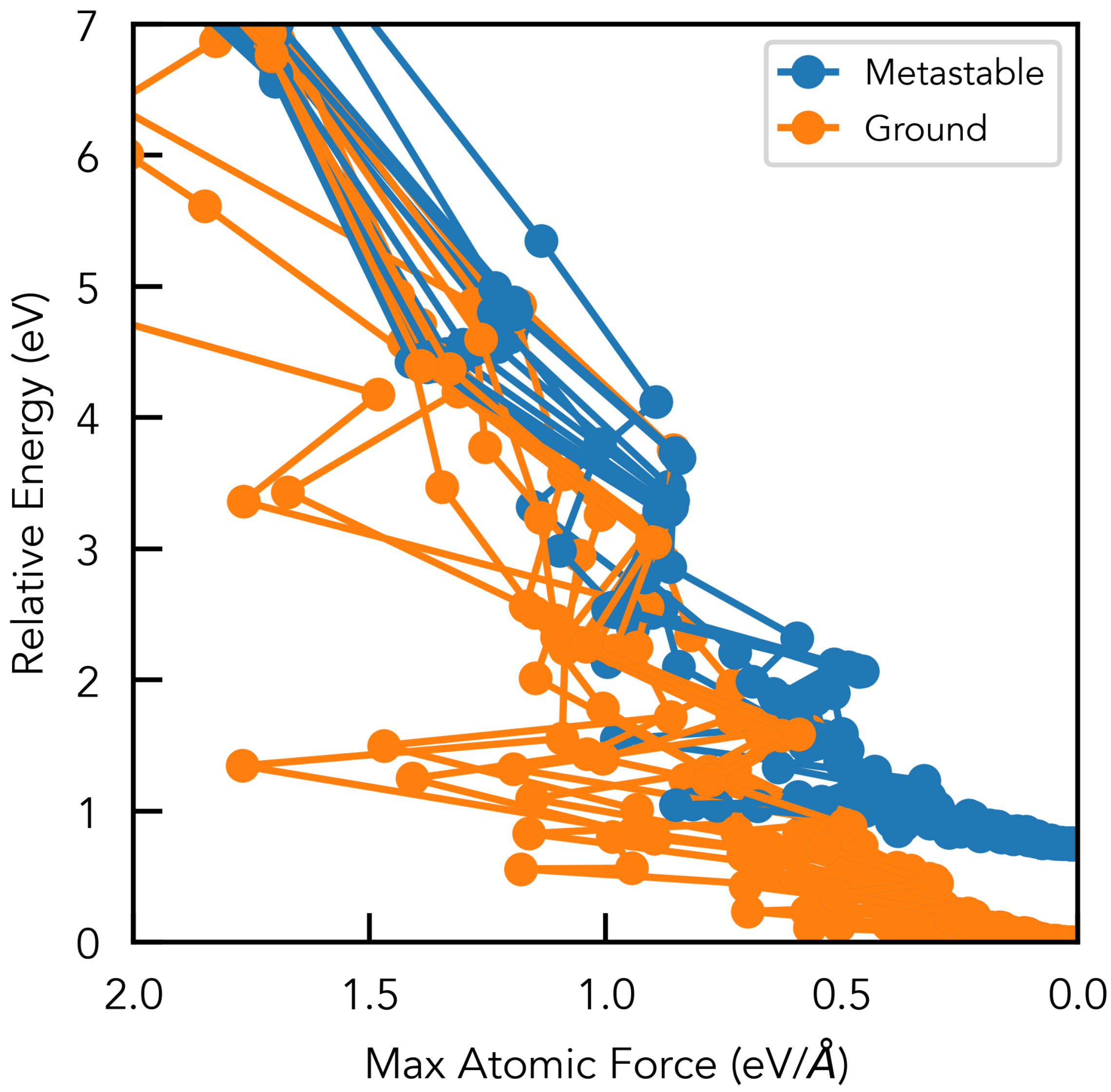
In [68]:

```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o"
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Metastable", color="
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Ground", color="C1",
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o")
ax.invert_xaxis()
ax.set_xlim(2,0)
ax.set_ylim(0, 7)
ax.legend()
ax.set_xlabel("Max Atomic Force (eV/$\AA$)")
ax.set_ylabel("Relative Energy (eV)")
```

Out[68]:

```
Text(0, 0.5, 'Relative Energy (eV)')
```



Ok so clear separation at `EDIFFG = -0.4` ... but this could be very system-dependent, depending on (1) the softness of the PES involved, and (2) the difference in energy between the two states (i.e. smaller energy difference likely means tighter force tolerance required to distinguish...)

In [69]:
```
for i in os.listdir("Rattling_Tests/vac_1_Cd_0/BDM/"):
    if "json" in i:
        df = pd.read_json(f"Rattling_Tests/vac_1_Cd_0/BDM/{i}")
        idx_0pt4_force = min(df.index[df.Max_Norm_Force < 0.4].tolist())
        if df.Energy.iloc[-1] < e_ground+0.05:
            print(f"Groundstate force less than 0.4 eV/A at step: {idx_0pt4_
        else:
            print(f"Metastable force less than 0.4 eV/A at step: {idx_0pt4_f
```

```
Metastable force less than 0.4 eV/A at step: 13, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.4 eV/A at step: 23, out of 51 steps for 0.01 e
V/A
Metastable force less than 0.4 eV/A at step: 14, out of 41 steps for 0.01 e
V/A
Metastable force less than 0.4 eV/A at step: 14, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.4 eV/A at step: 13, out of 41 steps for 0.01 e
V/A
Groundstate force less than 0.4 eV/A at step: 13, out of 41 steps for 0.01 e
V/A
Groundstate force less than 0.4 eV/A at step: 16, out of 45 steps for 0.01 e
V/A
Groundstate force less than 0.4 eV/A at step: 16, out of 44 steps for 0.01 e
V/A
Metastable force less than 0.4 eV/A at step: 14, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.4 eV/A at step: 13, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.4 eV/A at step: 13, out of 42 steps for 0.01 e
V/A
Metastable force less than 0.4 eV/A at step: 1, out of 25 steps for 0.01 eV/
A
Groundstate force less than 0.4 eV/A at step: 21, out of 51 steps for 0.01 e
V/A
Metastable force less than 0.4 eV/A at step: 17, out of 44 steps for 0.01 e
V/A
Groundstate force less than 0.4 eV/A at step: 19, out of 56 steps for 0.01 e
V/A
Groundstate force less than 0.4 eV/A at step: 13, out of 41 steps for 0.01 e
V/A
Groundstate force less than 0.4 eV/A at step: 13, out of 47 steps for 0.01 e
V/A
Metastable force less than 0.4 eV/A at step: 14, out of 35 steps for 0.01 e
V/A
Metastable force less than 0.4 eV/A at step: 14, out of 35 steps for 0.01 e
V/A
Metastable force less than 0.4 eV/A at step: 16, out of 39 steps for 0.01 e
V/A
Metastable force less than 0.4 eV/A at step: 16, out of 37 steps for 0.01 e
V/A
Metastable force less than 0.4 eV/A at step: 14, out of 37 steps for 0.01 e
V/A
Groundstate force less than 0.4 eV/A at step: 15, out of 44 steps for 0.01 e
V/A
Metastable force less than 0.4 eV/A at step: 16, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.4 eV/A at step: 13, out of 40 steps for 0.01 e
V/A
Groundstate force less than 0.4 eV/A at step: 13, out of 41 steps for 0.01 e
V/A
```

In [70]:
```
for i in os.listdir("Rattling_Tests/vac_1_Cd_0/BDM/"):
    if "json" in i:
        df = pd.read_json(f"Rattling_Tests/vac_1_Cd_0/BDM/{i}")
        idx_0pt4_force = min(df.index[df.Max_Norm_Force < 0.2].tolist())
        if df.Energy.iloc[-1] < e_ground+0.05:
            print(f"Groundstate force less than 0.2 eV/A at step: {idx_0pt4_
        else:
            print(f"Metastable force less than 0.2 eV/A at step: {idx_0pt4_f
```

```
Metastable force less than 0.2 eV/A at step: 19, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 29, out of 51 steps for 0.01 e
V/A
Metastable force less than 0.2 eV/A at step: 18, out of 41 steps for 0.01 e
V/A
Metastable force less than 0.2 eV/A at step: 17, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 17, out of 41 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 17, out of 41 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 22, out of 45 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 22, out of 44 steps for 0.01 e
V/A
Metastable force less than 0.2 eV/A at step: 17, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 17, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 17, out of 42 steps for 0.01 e
V/A
Metastable force less than 0.2 eV/A at step: 10, out of 25 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 29, out of 51 steps for 0.01 e
V/A
Metastable force less than 0.2 eV/A at step: 19, out of 44 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 25, out of 56 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 17, out of 41 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 19, out of 47 steps for 0.01 e
V/A
Metastable force less than 0.2 eV/A at step: 18, out of 35 steps for 0.01 e
V/A
Metastable force less than 0.2 eV/A at step: 17, out of 35 steps for 0.01 e
V/A
Metastable force less than 0.2 eV/A at step: 16, out of 39 steps for 0.01 e
V/A
Metastable force less than 0.2 eV/A at step: 16, out of 37 steps for 0.01 e
V/A
Metastable force less than 0.2 eV/A at step: 18, out of 37 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 19, out of 44 steps for 0.01 e
V/A
Metastable force less than 0.2 eV/A at step: 17, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 17, out of 40 steps for 0.01 e
V/A
Groundstate force less than 0.2 eV/A at step: 17, out of 41 steps for 0.01 e
V/A
```

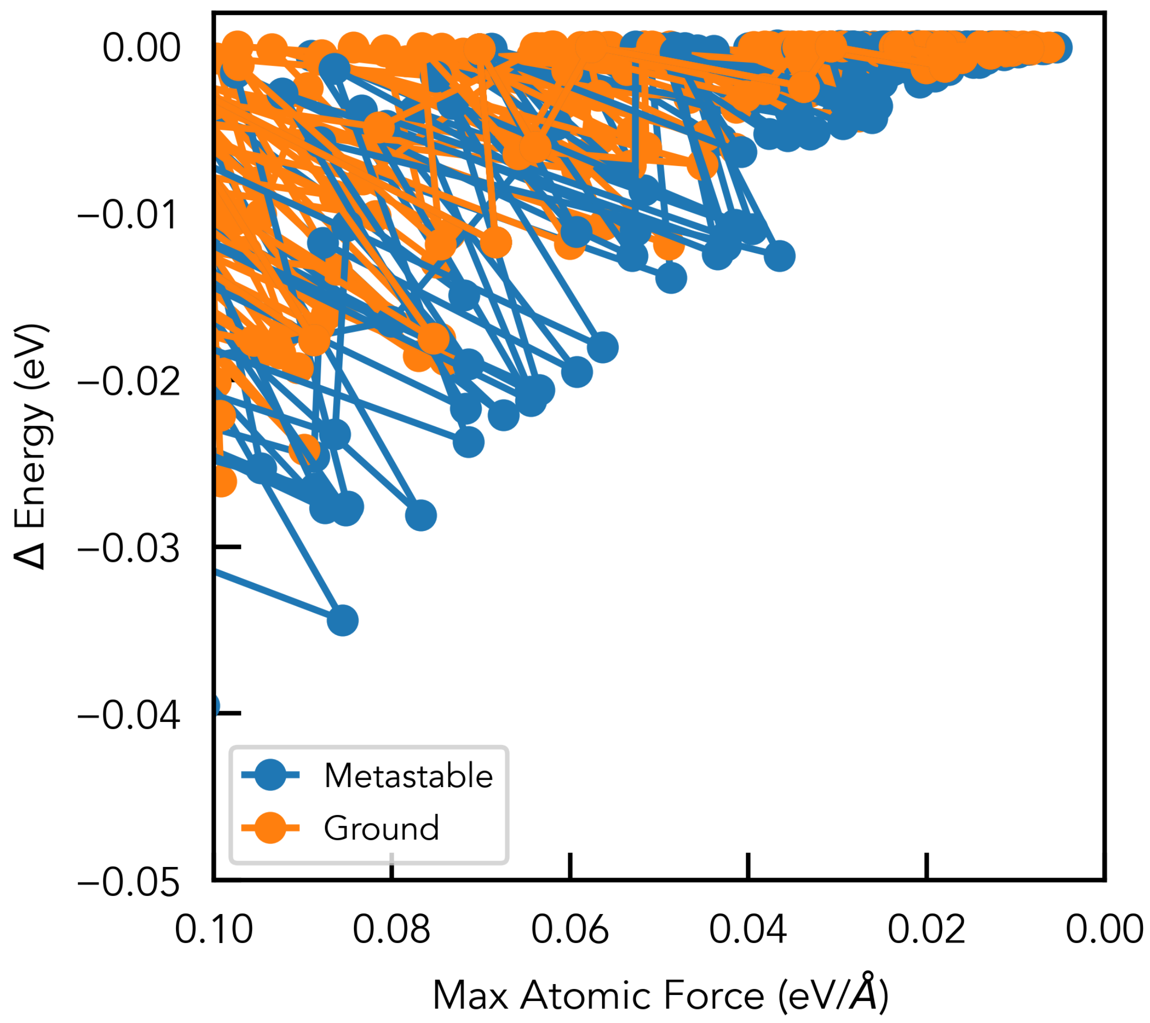
Ok, can we think of a worst-case scenario for soft PES and small energy difference (say ~0.1 eV)? Let's look at our other example cases.

In [71]:
```python
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
```

```
        else:
            c = "C0"
        df.loc[0, "Delta_E"] = df.loc[0, "Energy"]
        for i in range(1, len(df)):
            df.loc[i, "Delta_E"] = df.loc[i, "Energy"] - df.loc[i-1, "Energy
        ax.plot(df.Max_Norm_Force, df.Delta_E, color=c, marker="o")
ax.plot(df.Max_Norm_Force, df.Delta_E, label="Metastable", color="C0", marke
ax.plot(df.Max_Norm_Force, df.Delta_E, label="Ground", color="C1", marker="c
ax.plot(df.Max_Norm_Force, df.Delta_E, color=c, marker="o")
ax.invert_xaxis()
ax.set_xlim(0.1,0)
ax.set_ylim(-0.05, 0.002)
ax.legend()
ax.set_xlabel("Max Atomic Force (eV/$\AA$)")
ax.set_ylabel("$\Delta$ Energy (eV)")
```

Out[71]:
```
Text(0, 0.5, '$\\Delta$ Energy (eV)')
```



Hmm ok, so really we'd want to be looking at energy differences less than 0.01 eV or so I think, before truncating (thinking about worst case scenarios with soft PESs where you likely need accuracy on this level...). Suggests can change default `EDIFFG` to -0.02 eV/A but not really much more... Gives ~30% efficiency boost as shown below which is still nice

In [72]:
```
for i in os.listdir("Rattling_Tests/vac_1_Cd_0/BDM/"):
    if "json" in i:
        df = pd.read_json(f"Rattling_Tests/vac_1_Cd_0/BDM/{i}")
        idx_0pt4_force = min(df.index[df.Max_Norm_Force < 0.02].tolist())
```

```
        if df.Energy.iloc[-1] < e_ground+0.05:
            print(f"Groundstate force less than 0.02 eV/A at step: {idx_0pt4
        else:
            print(f"Metastable force less than 0.02 eV/A at step: {idx_0pt4_
```

```
Metastable force less than 0.02 eV/A at step: 31, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.02 eV/A at step: 49, out of 51 steps for 0.01
eV/A
Metastable force less than 0.02 eV/A at step: 32, out of 41 steps for 0.01 e
V/A
Metastable force less than 0.02 eV/A at step: 28, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.02 eV/A at step: 31, out of 41 steps for 0.01
eV/A
Groundstate force less than 0.02 eV/A at step: 33, out of 41 steps for 0.01
eV/A
Groundstate force less than 0.02 eV/A at step: 36, out of 45 steps for 0.01
eV/A
Groundstate force less than 0.02 eV/A at step: 38, out of 44 steps for 0.01
eV/A
Metastable force less than 0.02 eV/A at step: 28, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.02 eV/A at step: 31, out of 38 steps for 0.01
eV/A
Groundstate force less than 0.02 eV/A at step: 33, out of 42 steps for 0.01
eV/A
Metastable force less than 0.02 eV/A at step: 19, out of 25 steps for 0.01 e
V/A
Groundstate force less than 0.02 eV/A at step: 47, out of 51 steps for 0.01
eV/A
Metastable force less than 0.02 eV/A at step: 35, out of 44 steps for 0.01 e
V/A
Groundstate force less than 0.02 eV/A at step: 40, out of 56 steps for 0.01
eV/A
Groundstate force less than 0.02 eV/A at step: 31, out of 41 steps for 0.01
eV/A
Groundstate force less than 0.02 eV/A at step: 35, out of 47 steps for 0.01
eV/A
Metastable force less than 0.02 eV/A at step: 32, out of 35 steps for 0.01 e
V/A
Metastable force less than 0.02 eV/A at step: 28, out of 35 steps for 0.01 e
V/A
Metastable force less than 0.02 eV/A at step: 30, out of 39 steps for 0.01 e
V/A
Metastable force less than 0.02 eV/A at step: 30, out of 37 steps for 0.01 e
V/A
Metastable force less than 0.02 eV/A at step: 32, out of 37 steps for 0.01 e
V/A
Groundstate force less than 0.02 eV/A at step: 42, out of 44 steps for 0.01
eV/A
Metastable force less than 0.02 eV/A at step: 30, out of 38 steps for 0.01 e
V/A
Groundstate force less than 0.02 eV/A at step: 33, out of 40 steps for 0.01
eV/A
Groundstate force less than 0.02 eV/A at step: 31, out of 41 steps for 0.01
eV/A
```

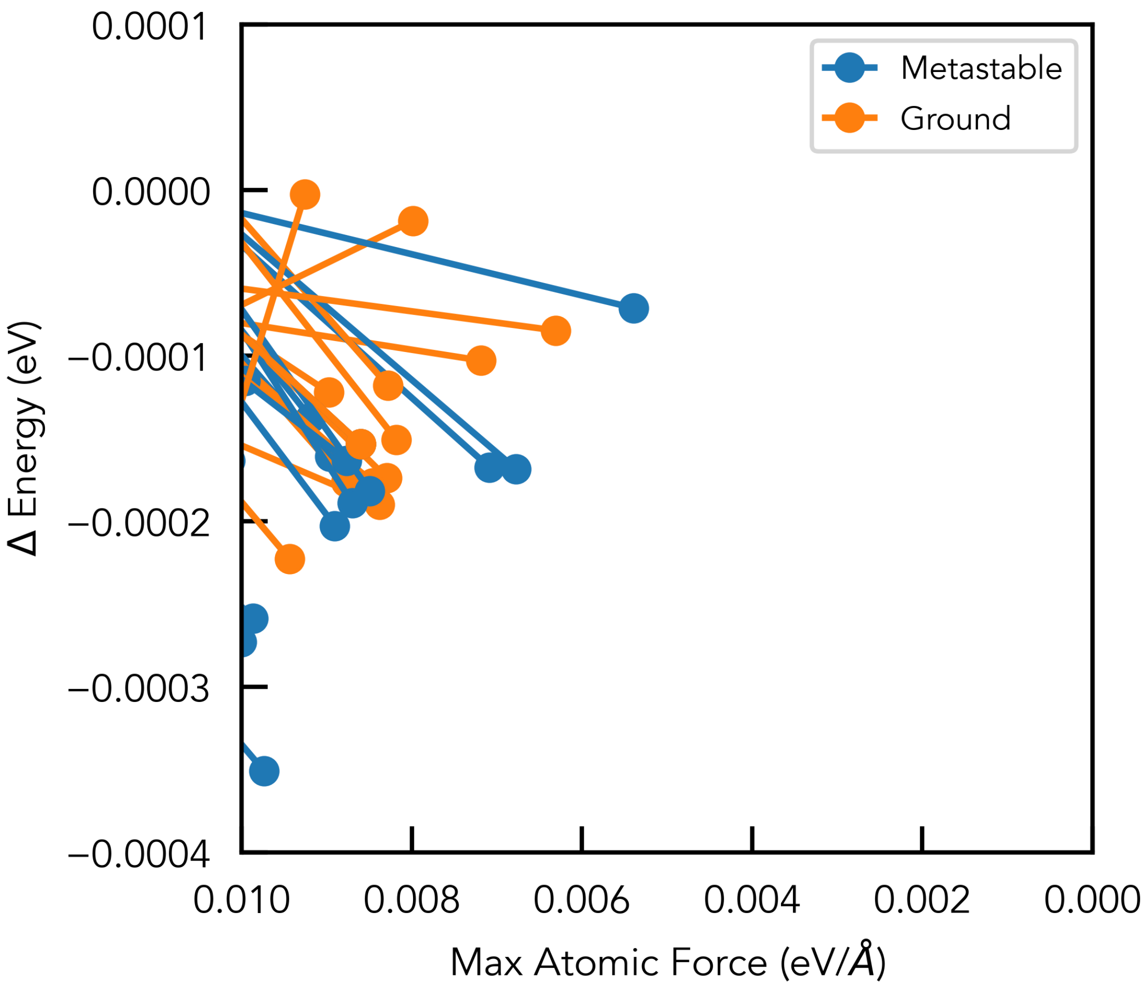
Just confirming that we only have one single step each after the max force < `EDIFFG` (i.e. when our stopping criterion is reached), as a sanity check for our force parsing:

In [52]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
```

```
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        df.loc[0, "Delta_E"] = df.loc[0, "Energy"]
        for i in range(1, len(df)):
            df.loc[i, "Delta_E"] = df.loc[i, "Energy"] - df.loc[i-1, "Energy
        ax.plot(df.Max_Norm_Force, df.Delta_E, color=c, marker="o")
ax.plot(df.Max_Norm_Force, df.Delta_E, label="Metastable", color="C0", marke
ax.plot(df.Max_Norm_Force, df.Delta_E, label="Ground", color="C1", marker="c
ax.plot(df.Max_Norm_Force, df.Delta_E, color=c, marker="o")
ax.invert_xaxis()
ax.set_xlim(0.01,0)
ax.set_ylim(-0.0004, 0.0001)
ax.legend()
ax.set_xlabel("Max Atomic Force (eV/$\AA$)")
ax.set_ylabel("$\Delta$ Energy (eV)")
```

Out[52]: Text(0, 0.5, '$\\Delta$ Energy (eV)')



Yep, only one datapoint each for 'Max Atomic Force' < 0.01

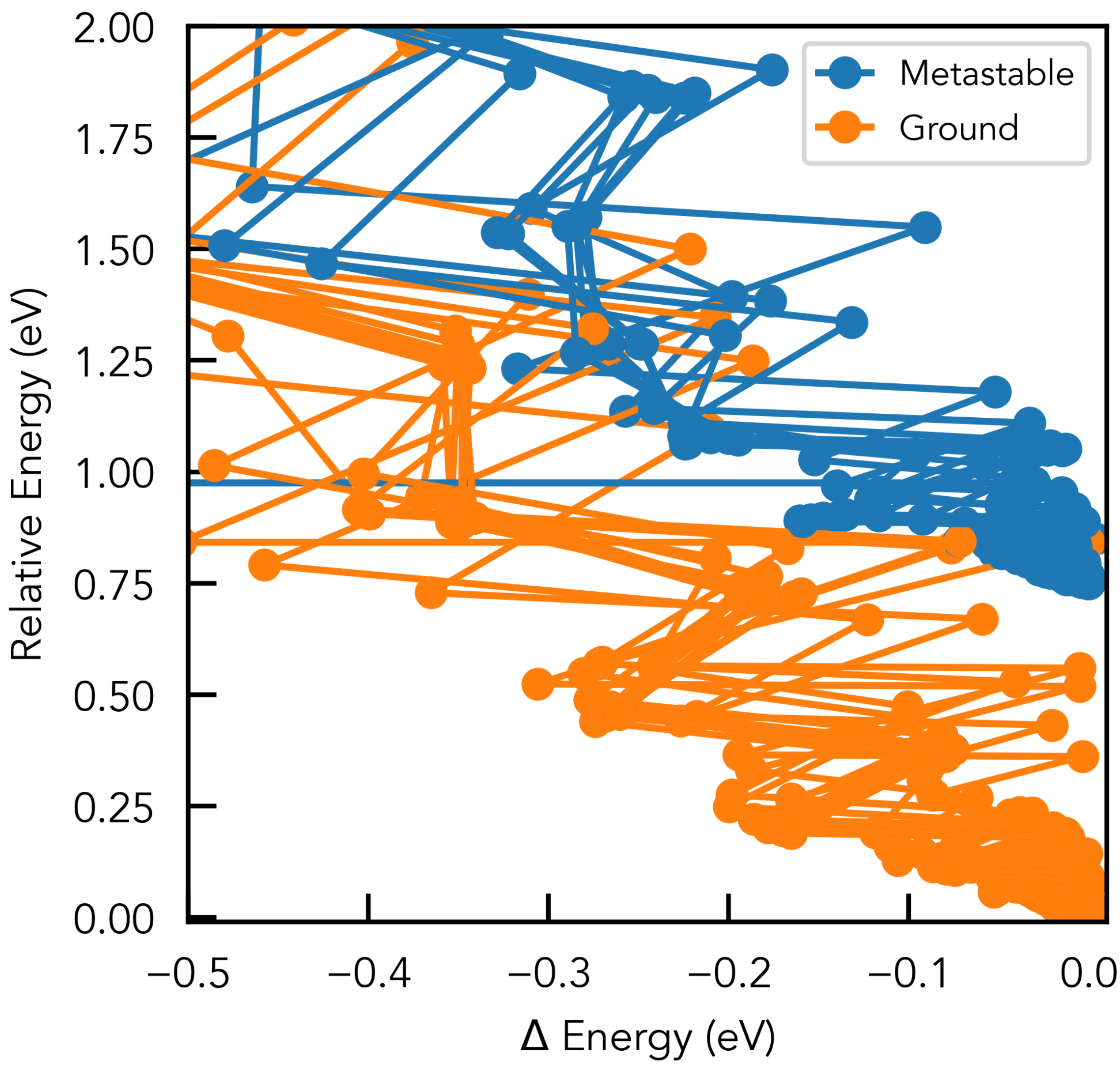
Could we possibly use energy differences for `EDIFFG` instead of forces?

In [81]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
```

```
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        df.loc[0, "Delta_E"] = df.loc[0, "Energy"]
        for i in range(1, len(df)):
            df.loc[i, "Delta_E"] = df.loc[i, "Energy"] - df.loc[i-1, "Energy
        ax.plot(df.Delta_E, df.Energy - e_ground, color=c, marker="o")
ax.plot(df.Delta_E, df.Energy - e_ground, label="Metastable", color="C0", ma
ax.plot(df.Delta_E, df.Energy - e_ground, label="Ground", color="C1", marker
ax.plot(df.Delta_E, df.Energy - e_ground, color=c, marker="o")
ax.set_xlim(-0.5, 0.01)
ax.set_ylim(-0.01, 2)
ax.legend()
ax.set_ylabel("Relative Energy (eV)")
ax.set_xlabel("$\Delta$ Energy (eV)")
```

Out[81]: `Text(0.5, 0, '$\\Delta$ Energy (eV)')`



No, we can see here some "Ground" datapoints have ΔE close to zero despite being far from fully relaxed. There are a few situations where this can happen I think, such as having significant forces and moving the atoms, but with this rearrangement actually giving a similar energy to the previous step despite not being converged (like sketch below). So no we should def stick with `EDIFFG` as a force convergence criterion

Let's test some other cases

V_{Na}^0 in $NaBiS_2$ ("The Disordered One")

Nice example of 'worst-case scenario', with a ground/metastable energy difference of only ~0.075 eV, and big disordered SQS supercell (-> soft-ish and complex PES, tricky convergence)

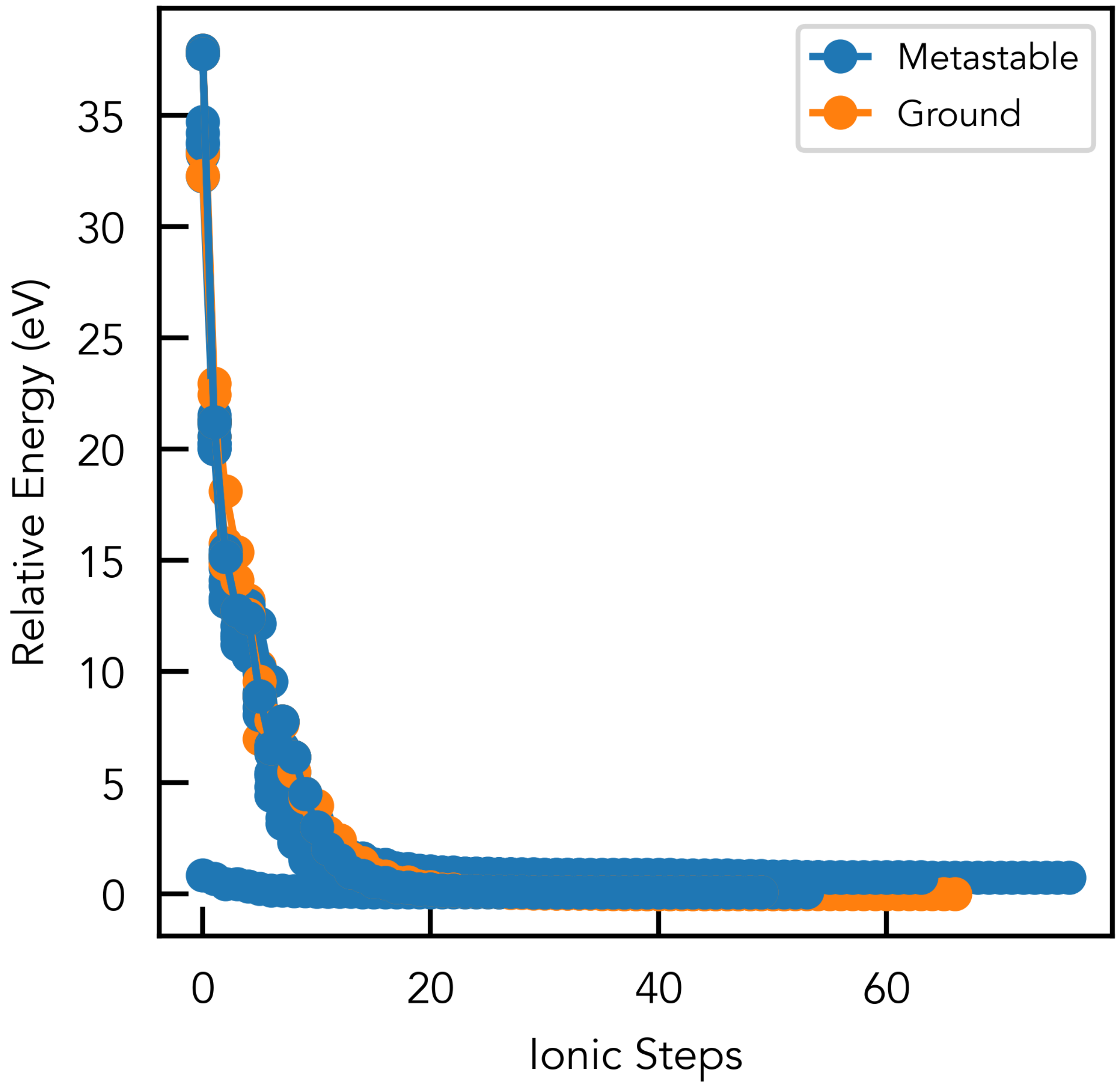
In [83]:
```
import os
import matplotlib.pyplot as plt
import pandas as pd
plt.style.use("../Packages/publication_style.mplstyle")

json_dir = "E_and_F_vs_Step_Tests/vac_10_Na_0/"
e_ground = -382.822
```

In [54]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Energy - e_ground, color=c, marker="o")  # groundstate en
ax.plot(df.Energy - e_ground, label="Metastable", color="C0", marker="o")
ax.plot(df.Energy - e_ground, label="Ground", color="C1", marker="o")
ax.plot(df.Energy - e_ground, color=c, marker="o")
#ax.set_xlim(0,30)
ax.legend()
ax.set_xlabel("Ionic Steps")
ax.set_ylabel("Relative Energy (eV)")
```

Out[54]:
```
Text(0, 0.5, 'Relative Energy (eV)')
```

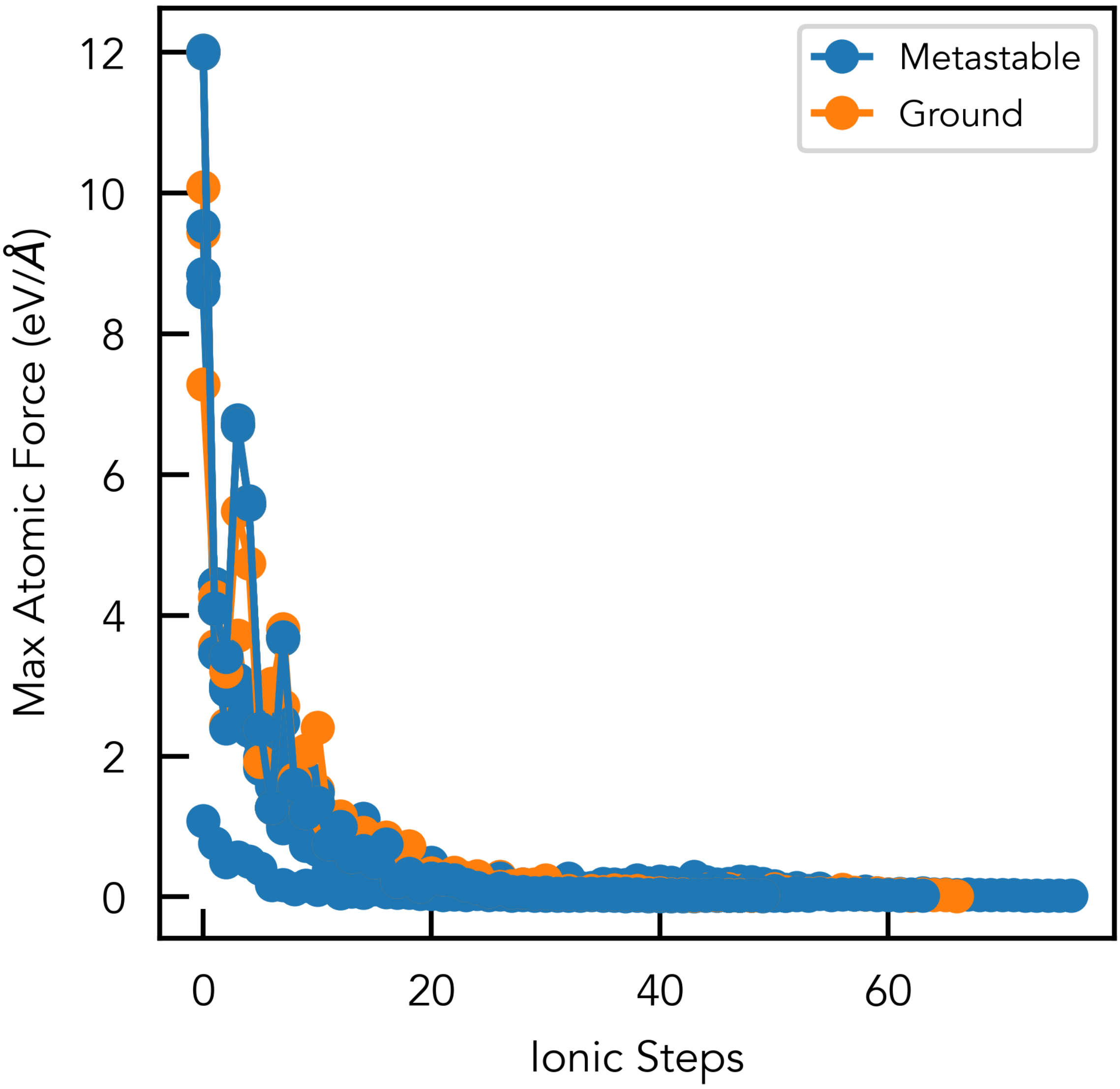


Yep again no clear correlation between ground/metastable and energy vs step. Quasi-flat line is unperturbed as you'd imagine

In [55]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.plot(df.Max_Norm_Force, label="Metastable", color="C0", marker="o")
ax.plot(df.Max_Norm_Force, label="Ground", color="C1", marker="o")
ax.plot(df.Max_Norm_Force, color=c, marker="o")
#ax.set_xlim(0,10)
ax.legend()
ax.set_xlabel("Ionic Steps")
ax.set_ylabel("Max Atomic Force (eV/$\AA$)")
```

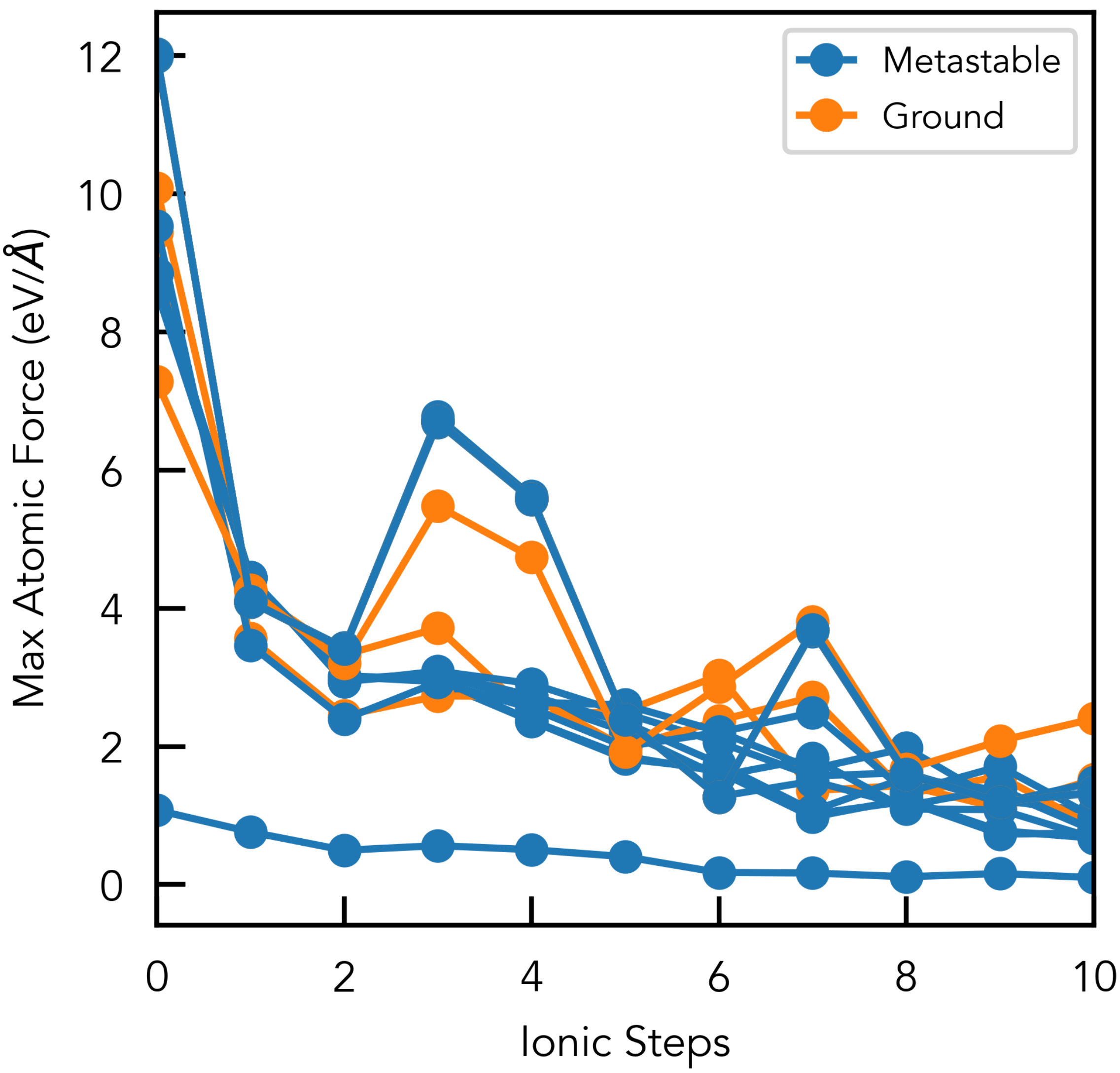
Out[55]:
```
Text(0, 0.5, 'Max Atomic Force (eV/$\\AA$)')
```



In [56]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.plot(df.Max_Norm_Force, label="Metastable", color="C0", marker="o")
ax.plot(df.Max_Norm_Force, label="Ground", color="C1", marker="o")
ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.set_xlim(0,10)
ax.legend()
ax.set_xlabel("Ionic Steps")
ax.set_ylabel("Max Atomic Force (eV/$\AA$)")
```

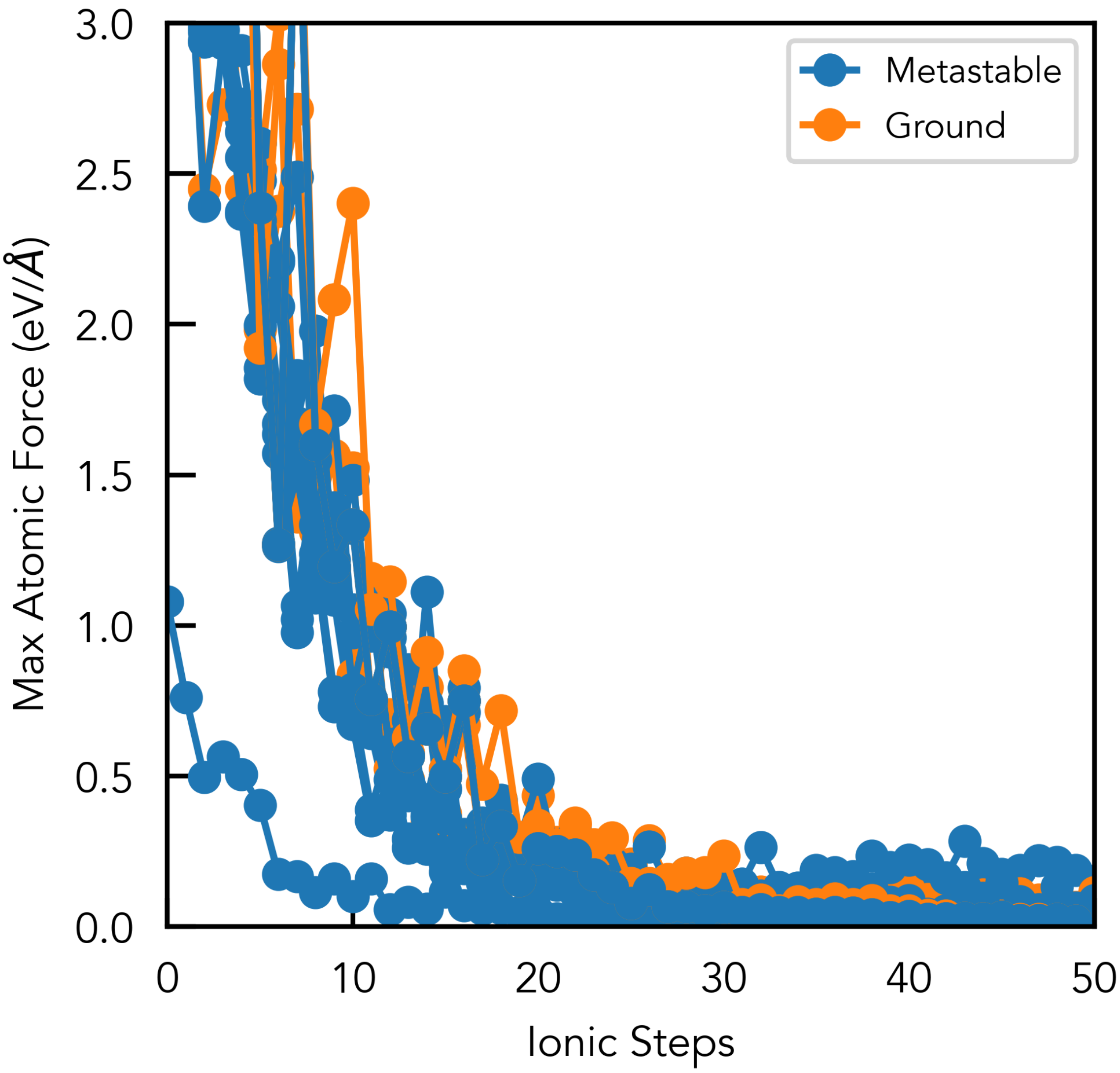
Out[56]:
```
Text(0, 0.5, 'Max Atomic Force (eV/$\\AA$)')
```



In [57]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.plot(df.Max_Norm_Force, label="Metastable", color="C0", marker="o")
ax.plot(df.Max_Norm_Force, label="Ground", color="C1", marker="o")
ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.set_xlim(0,50)
ax.set_ylim(0,3)
ax.legend()
ax.set_xlabel("Ionic Steps")
ax.set_ylabel("Max Atomic Force (eV/$\AA$)")
```

Out[57]:
```
Text(0, 0.5, 'Max Atomic Force (eV/$\\AA$)')
```

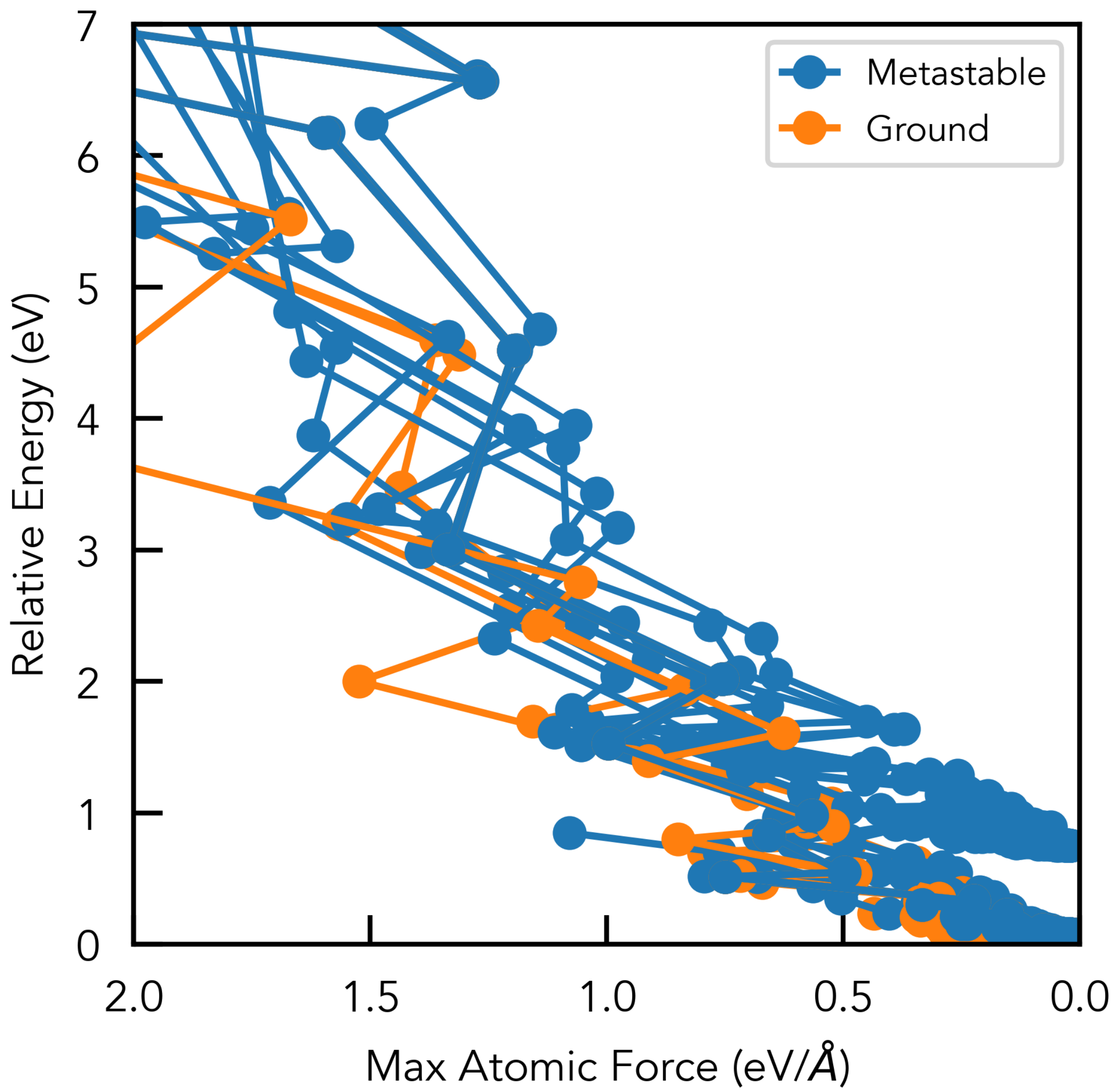


Again no real clear correlation

In [58]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o"
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Metastable", color="
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Ground", color="C1",
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o")
ax.invert_xaxis()
ax.set_xlim(2,0)
ax.set_ylim(0, 7)
ax.legend()
ax.set_xlabel("Max Atomic Force (eV/$\AA$)")
ax.set_ylabel("Relative Energy (eV)")
```

Out[58]: `Text(0, 0.5, 'Relative Energy (eV)')`

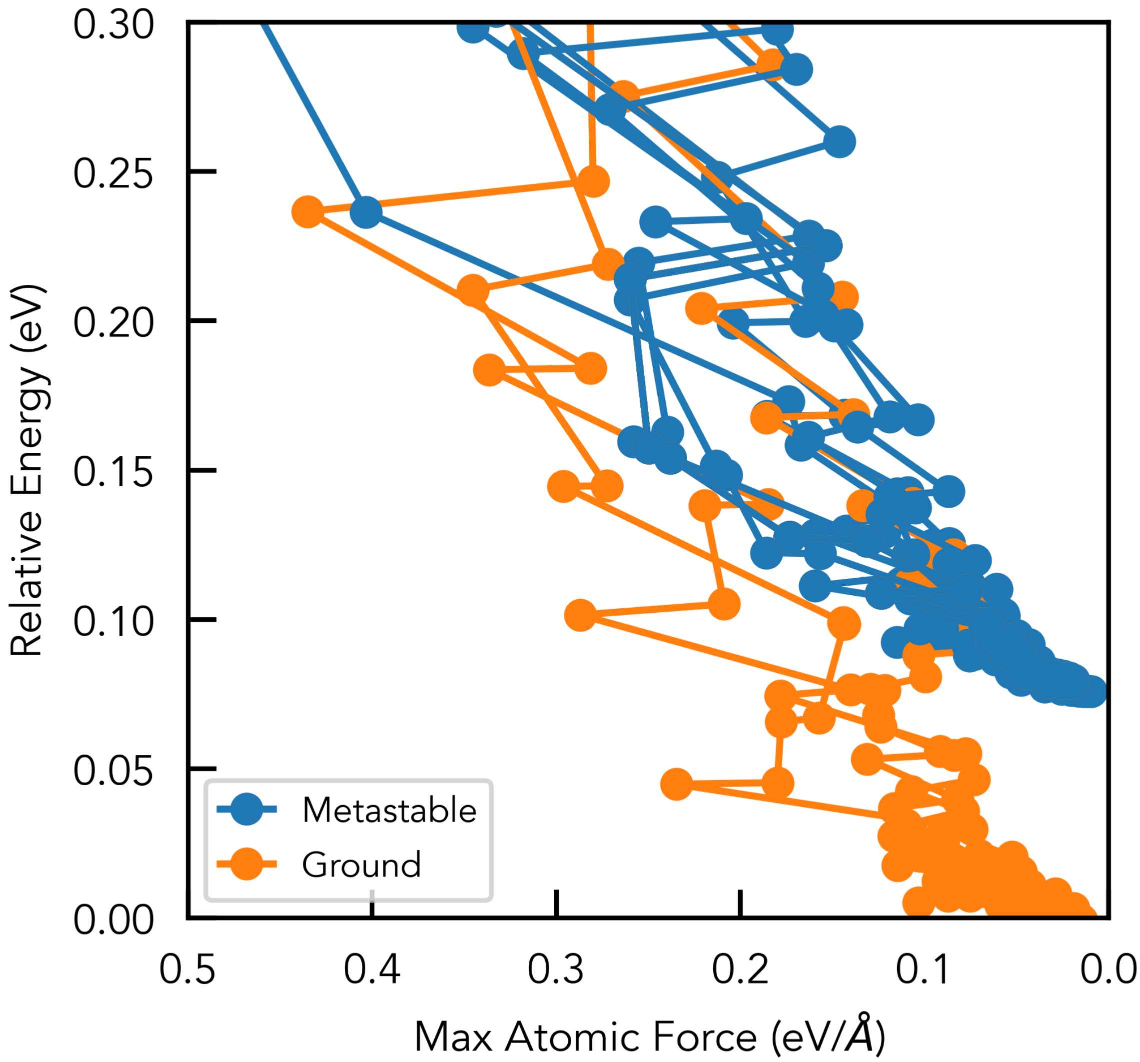


(We also had a higher energy metastable state ~0.8 eV higher than groundstate as seen in this plot above)

In [59]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o"
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Metastable", color="
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Ground", color="C1",
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o")
ax.invert_xaxis()
ax.set_xlim(0.5,0)
ax.set_ylim(0, 0.3)
ax.legend()
ax.set_xlabel("Max Atomic Force (eV/$\AA$)")
ax.set_ylabel("Relative Energy (eV)")
```

Out[59]:
```
Text(0, 0.5, 'Relative Energy (eV)')
```



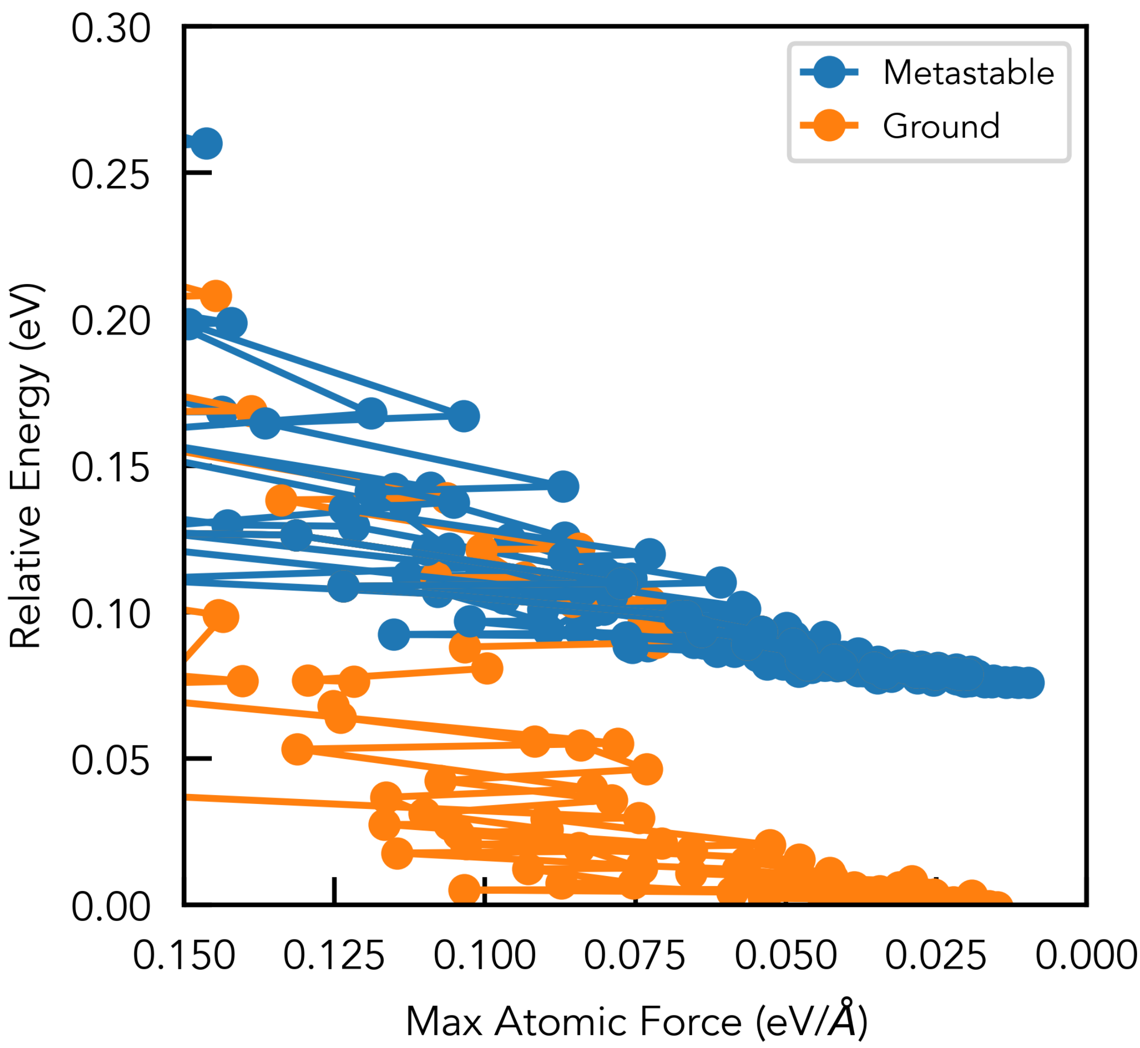
In [60]:

```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o"
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Metastable", color="
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Ground", color="C1",
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o")
ax.invert_xaxis()
ax.set_xlim(0.15,0)
ax.set_ylim(0, 0.3)
ax.legend()
ax.set_xlabel("Max Atomic Force (eV/$\AA$)")
ax.set_ylabel("Relative Energy (eV)")
```

Out[60]:

```
Text(0, 0.5, 'Relative Energy (eV)')
```



Ok so only really clear separation at `EDIFFG = -0.06` here (c.f. -0.4 for V_Cd above), exemplifying the system dependence of this. This does seem to be one of our 'worst-case scenarios' though, with a ground/metastable energy difference of only ~0.075 eV

In [61]:
```
for i in os.listdir("Rattling_Tests/vac_1_Cd_0/BDM/"):
    if "json" in i:
        df = pd.read_json(f"Rattling_Tests/vac_1_Cd_0/BDM/{i}")
        idx_0pt4_force = min(df.index[df.Max_Norm_Force < 0.06].tolist())
        if df.Energy.iloc[-1] < e_ground+0.05:
            print(f"Groundstate force less than 0.06 eV/A at step: {idx_0pt4
        else:
            print(f"Metastable force less than 0.06 eV/A at step: {idx_0pt4_
```

```
Metastable force less than 0.06 eV/A at step: 25, out of 38 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 39, out of 51 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 41 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 38 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 25, out of 41 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 27, out of 41 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 28, out of 45 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 32, out of 44 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 22, out of 38 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 27, out of 38 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 25, out of 42 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 14, out of 25 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 35, out of 51 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 25, out of 44 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 31, out of 56 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 27, out of 41 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 29, out of 47 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 22, out of 35 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 35 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 39 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 37 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 37 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 31, out of 44 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 38 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 28, out of 40 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 28, out of 41 steps for 0.02 e
V/A
```

Note here we actually used `EDIFFG = -0.02` in our calcs anyway

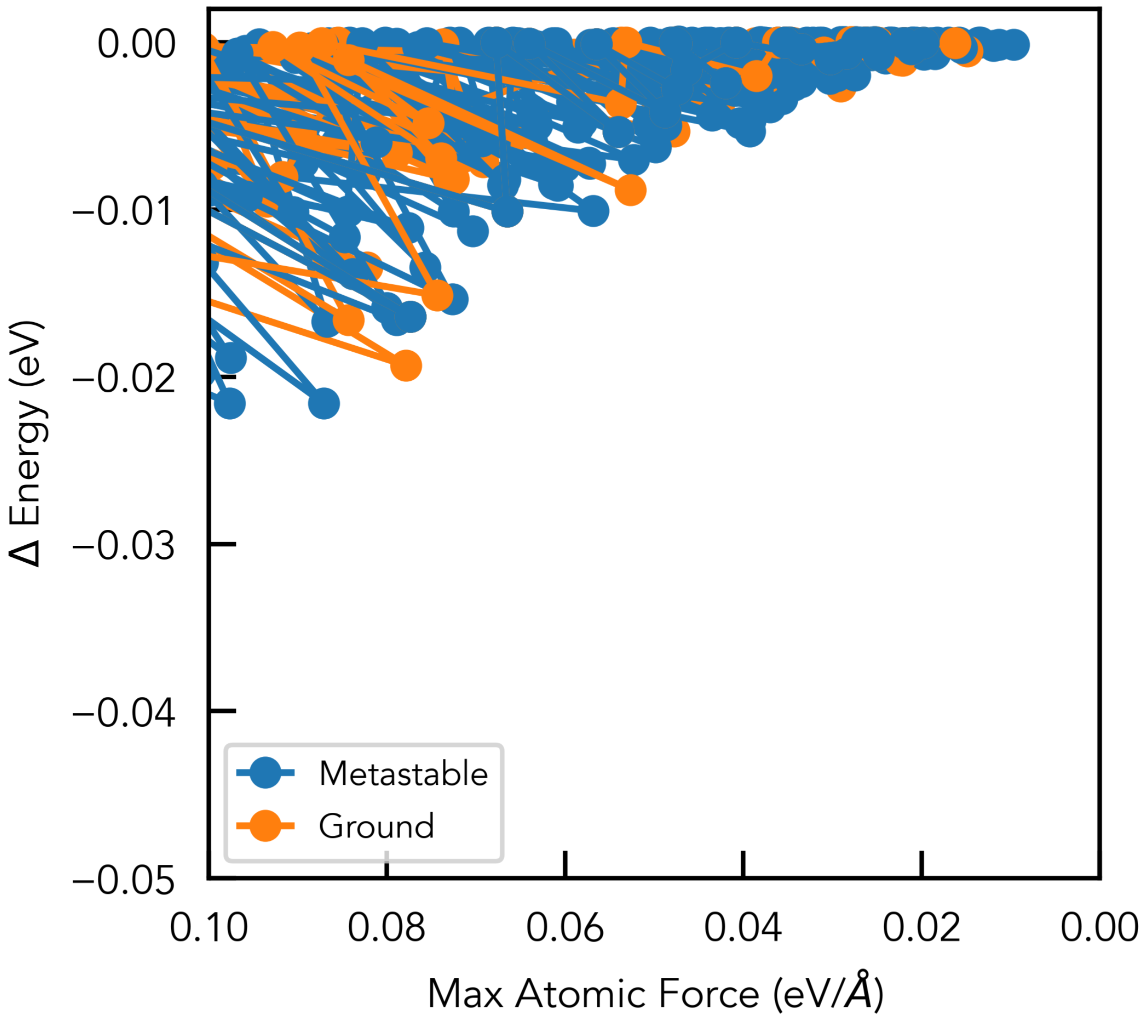
These results at least demonstrate that even a relatively small increase in `EDIFFG` does give a pretty significant reduction in required ionic steps (and thus calc cost)

In [84]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
```

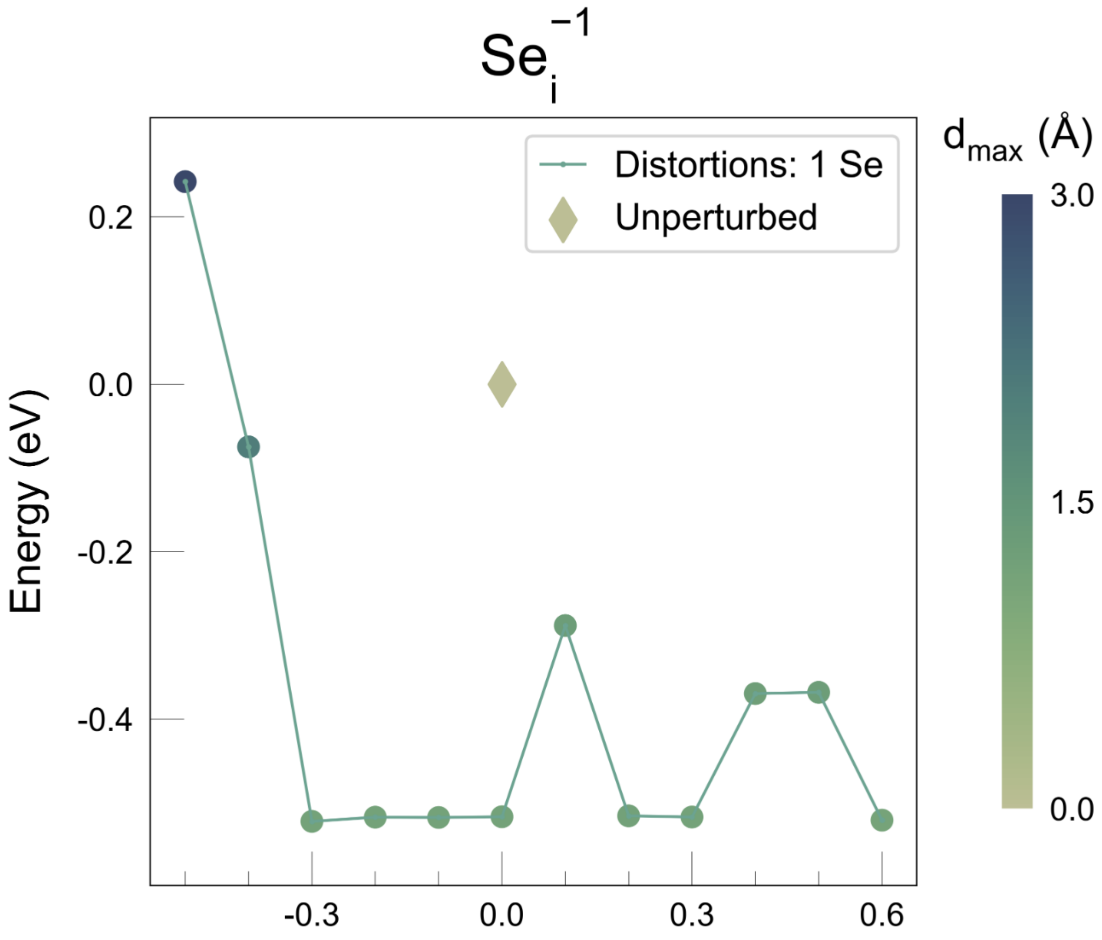
```
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        df.loc[0, "Delta_E"] = df.loc[0, "Energy"]
        for i in range(1, len(df)):
            df.loc[i, "Delta_E"] = df.loc[i, "Energy"] - df.loc[i-1, "Energy
        ax.plot(df.Max_Norm_Force, df.Delta_E, color=c, marker="o")
ax.plot(df.Max_Norm_Force, df.Delta_E, label="Metastable", color="C0", marke
ax.plot(df.Max_Norm_Force, df.Delta_E, label="Ground", color="C1", marker="c
ax.plot(df.Max_Norm_Force, df.Delta_E, color=c, marker="o")
ax.invert_xaxis()
ax.set_xlim(0.1,0)
ax.set_ylim(-0.05, 0.002)
ax.legend()
ax.set_xlabel("Max Atomic Force (eV/$\AA$)")
ax.set_ylabel("$\Delta$ Energy (eV)")
```

Out[84]: Text(0, 0.5, '$\\Delta$ Energy (eV)')



Ok so yeah as expected, once all energy differences below ~0.01 eV (which happens around EDIFFG = -0.05) we're sufficiently converged to distinguish our groundstate structures, in this worst-case scenario.

Se_{i}^{-1} in Se ("The 'Simple' One")



Nice example of typical scenario, with ground/metastable energy differences of only ~0.15 eV, and relatively soft material

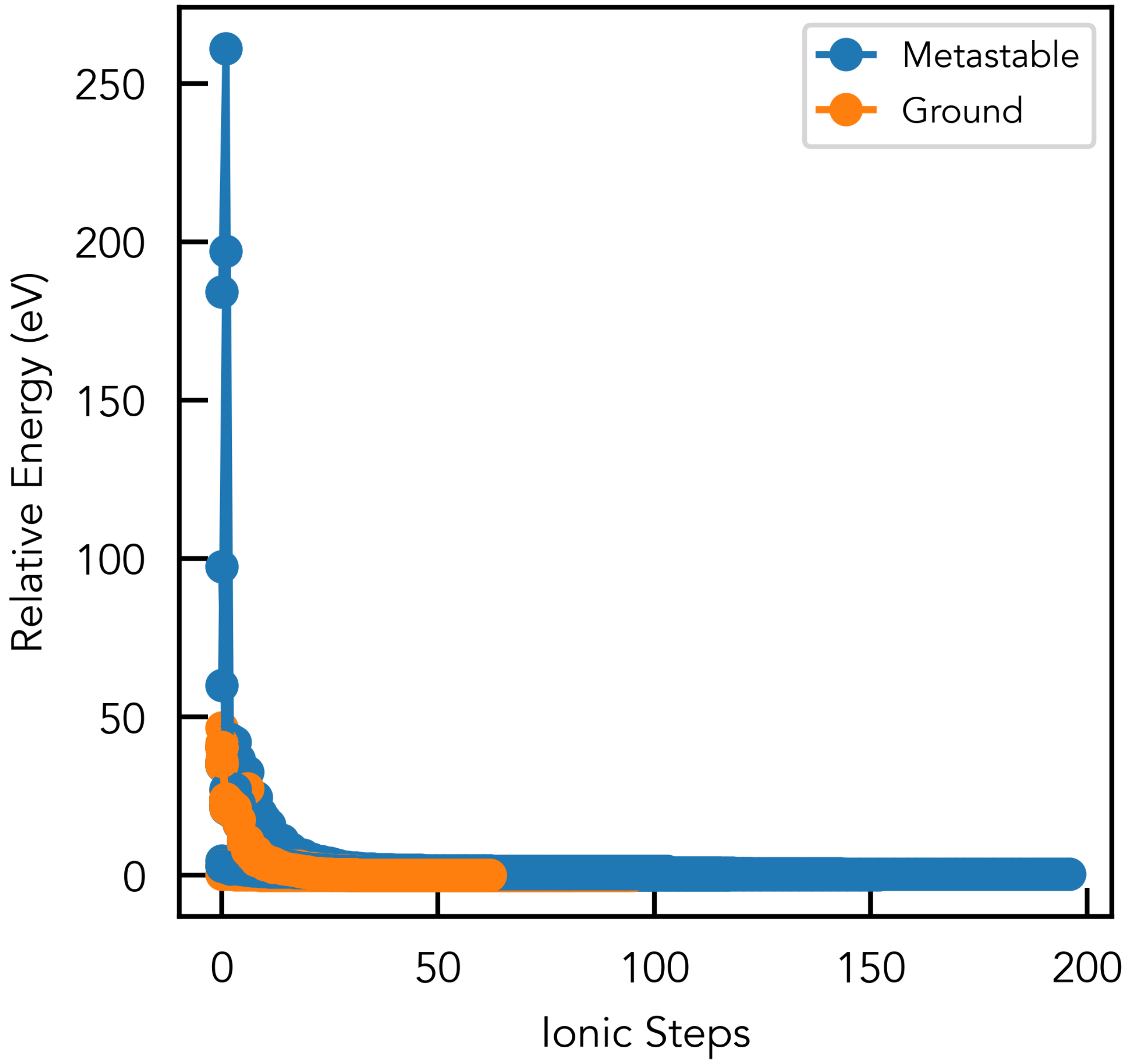
In [87]:
```
import os
import matplotlib.pyplot as plt
import pandas as pd
plt.style.use("../Packages/publication_style.mplstyle")

json_dir = "E_and_F_vs_Step_Tests/Int_Se_1_-1/"
e_ground = -376.83
```

In [88]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Energy - e_ground, color=c, marker="o")  # groundstate en
ax.plot(df.Energy - e_ground, label="Metastable", color="C0", marker="o")
ax.plot(df.Energy - e_ground, label="Ground", color="C1", marker="o")
ax.plot(df.Energy - e_ground, color=c, marker="o")
#ax.set_xlim(0,30)
ax.legend()
ax.set_xlabel("Ionic Steps")
ax.set_ylabel("Relative Energy (eV)")
```

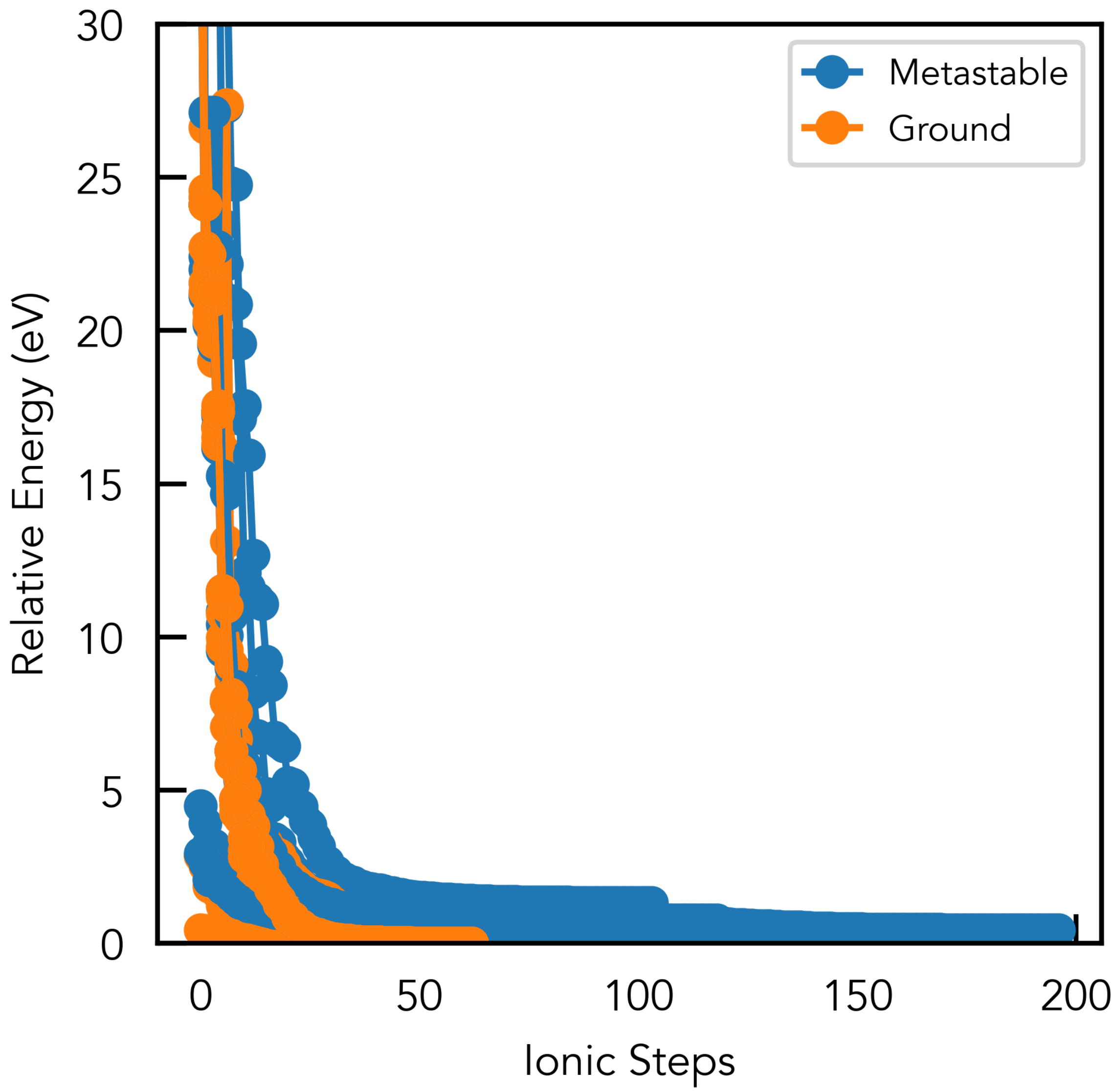
Out[88]:
```
Text(0, 0.5, 'Relative Energy (eV)')
```



In [89]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Energy - e_ground, color=c, marker="o")  # groundstate en
ax.plot(df.Energy - e_ground, label="Metastable", color="C0", marker="o")
ax.plot(df.Energy - e_ground, label="Ground", color="C1", marker="o")
ax.plot(df.Energy - e_ground, color=c, marker="o")
ax.set_ylim(0,30)
ax.legend()
ax.set_xlabel("Ionic Steps")
ax.set_ylabel("Relative Energy (eV)")
```

Out[89]:
```
Text(0, 0.5, 'Relative Energy (eV)')
```

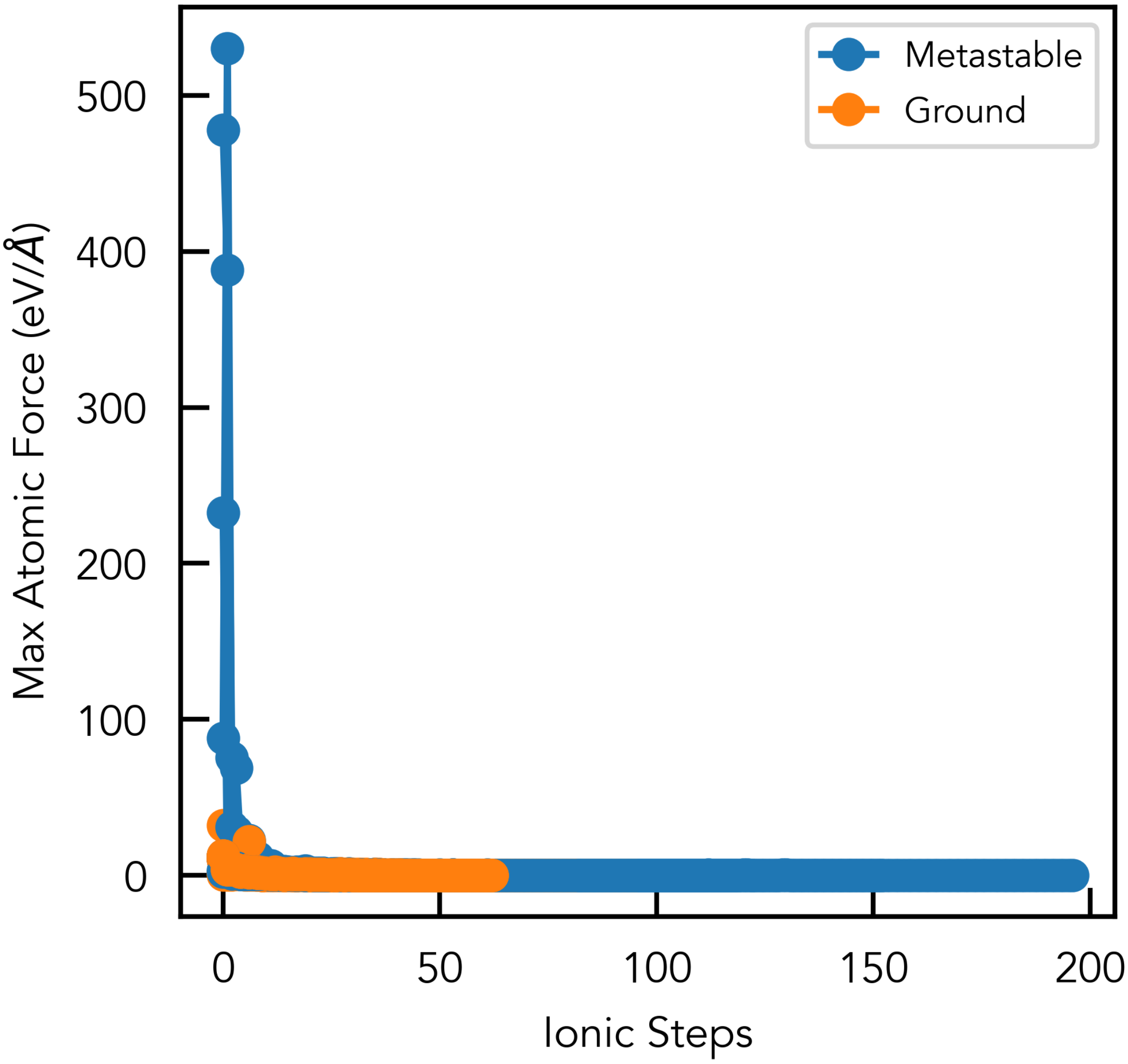


Yep again no clear correlation between ground/metastable and energy vs step

In [90]:
```python
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.plot(df.Max_Norm_Force, label="Metastable", color="C0", marker="o")
ax.plot(df.Max_Norm_Force, label="Ground", color="C1", marker="o")
ax.plot(df.Max_Norm_Force, color=c, marker="o")
#ax.set_xlim(0,10)
ax.legend()
ax.set_xlabel("Ionic Steps")
ax.set_ylabel("Max Atomic Force (eV/$\AA$)")
```

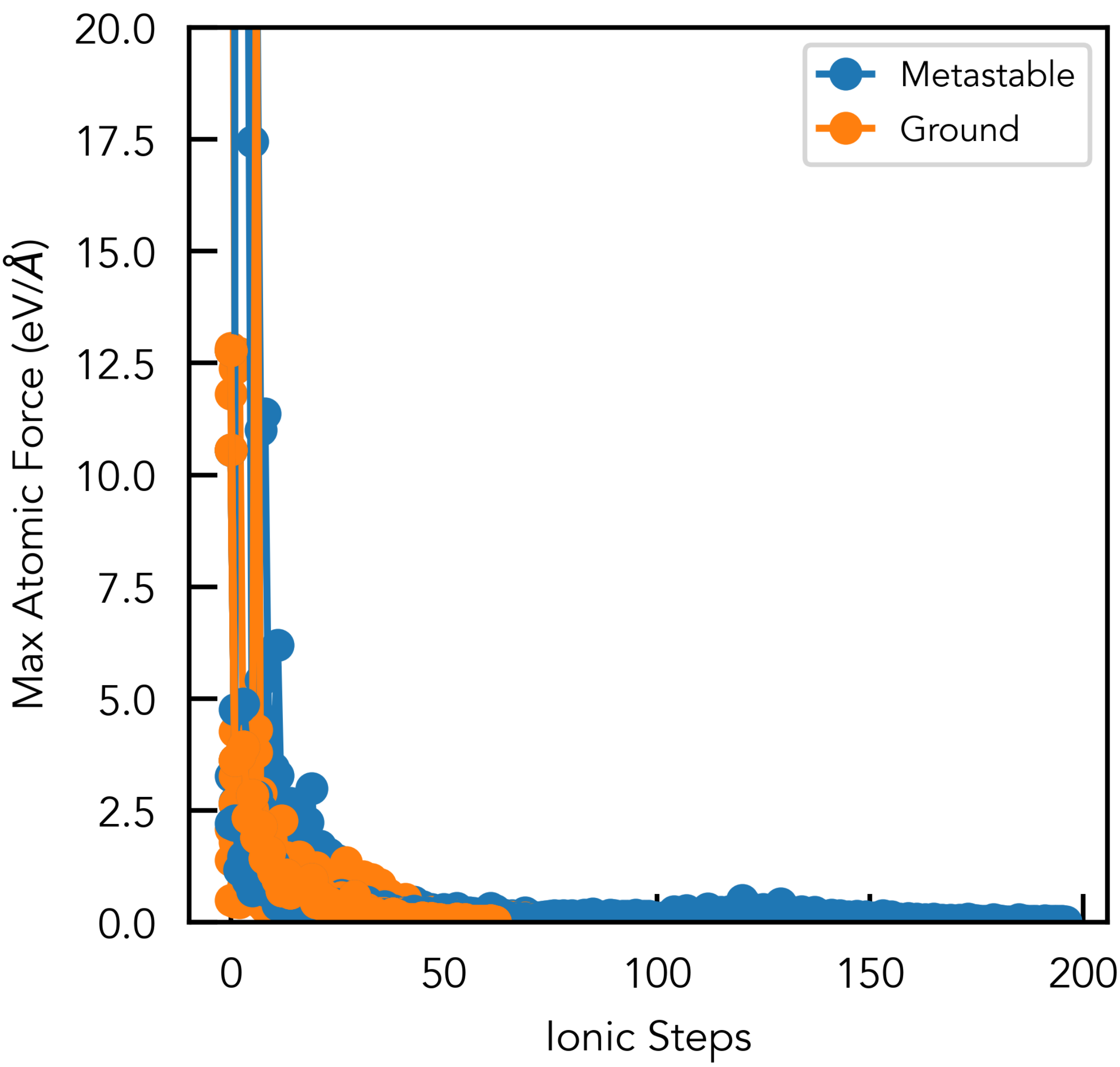
Out[90]:
```
Text(0, 0.5, 'Max Atomic Force (eV/$\\AA$)')
```



In [92]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.plot(df.Max_Norm_Force, label="Metastable", color="C0", marker="o")
ax.plot(df.Max_Norm_Force, label="Ground", color="C1", marker="o")
ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.set_ylim(0,20)
ax.legend()
ax.set_xlabel("Ionic Steps")
ax.set_ylabel("Max Atomic Force (eV/$\AA$)")
```

Out[92]:
```
Text(0, 0.5, 'Max Atomic Force (eV/$\\AA$)')
```



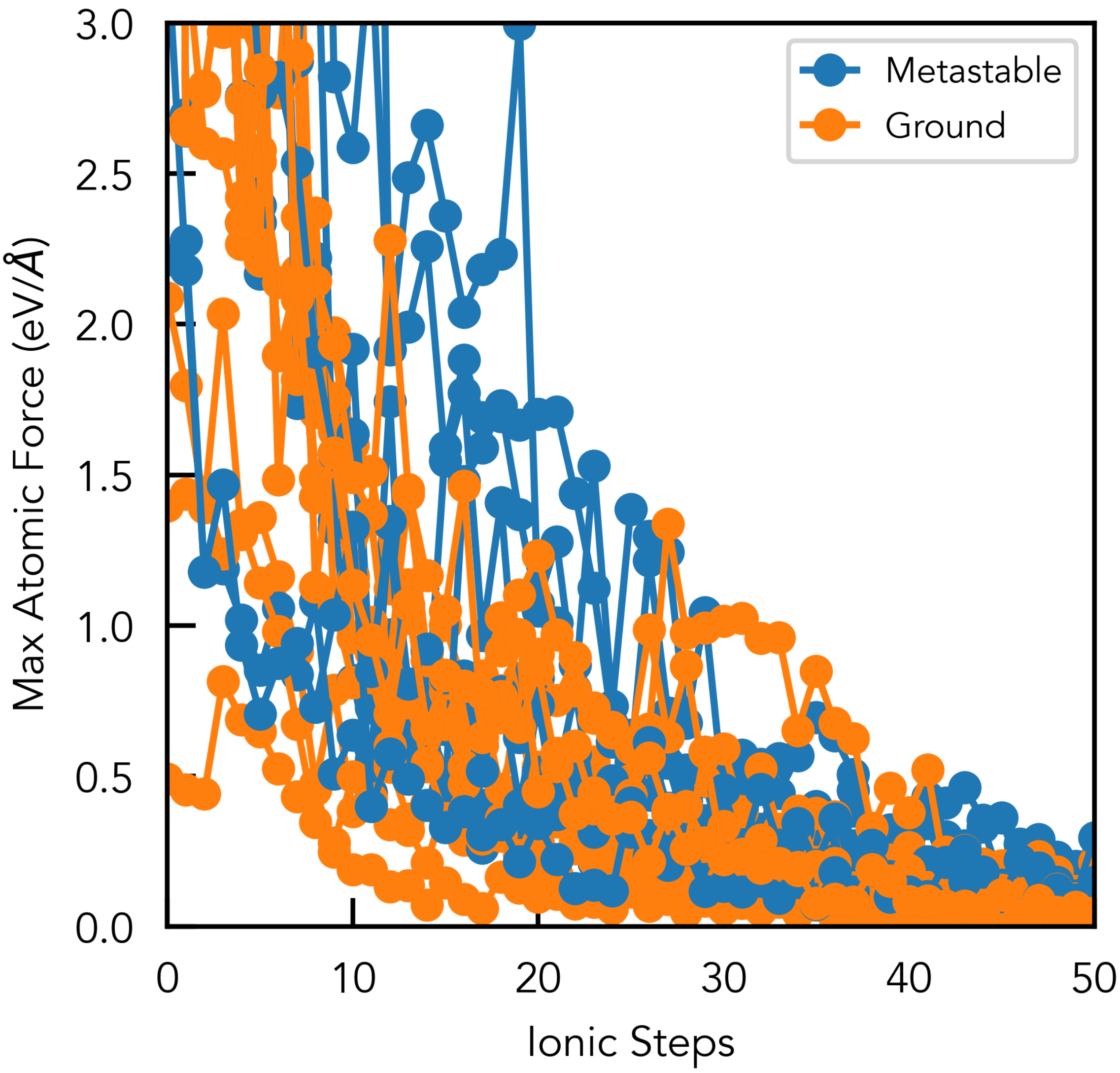
In [94]:

```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.plot(df.Max_Norm_Force, label="Metastable", color="C0", marker="o")
ax.plot(df.Max_Norm_Force, label="Ground", color="C1", marker="o")
ax.plot(df.Max_Norm_Force, color=c, marker="o")
ax.set_xlim(0,50)
ax.set_ylim(0,3)
ax.legend()
ax.set_xlabel("Ionic Steps")
ax.set_ylabel("Max Atomic Force (eV/$\AA$)")
```

Out[94]:

```
Text(0, 0.5, 'Max Atomic Force (eV/$\\AA$)')
```

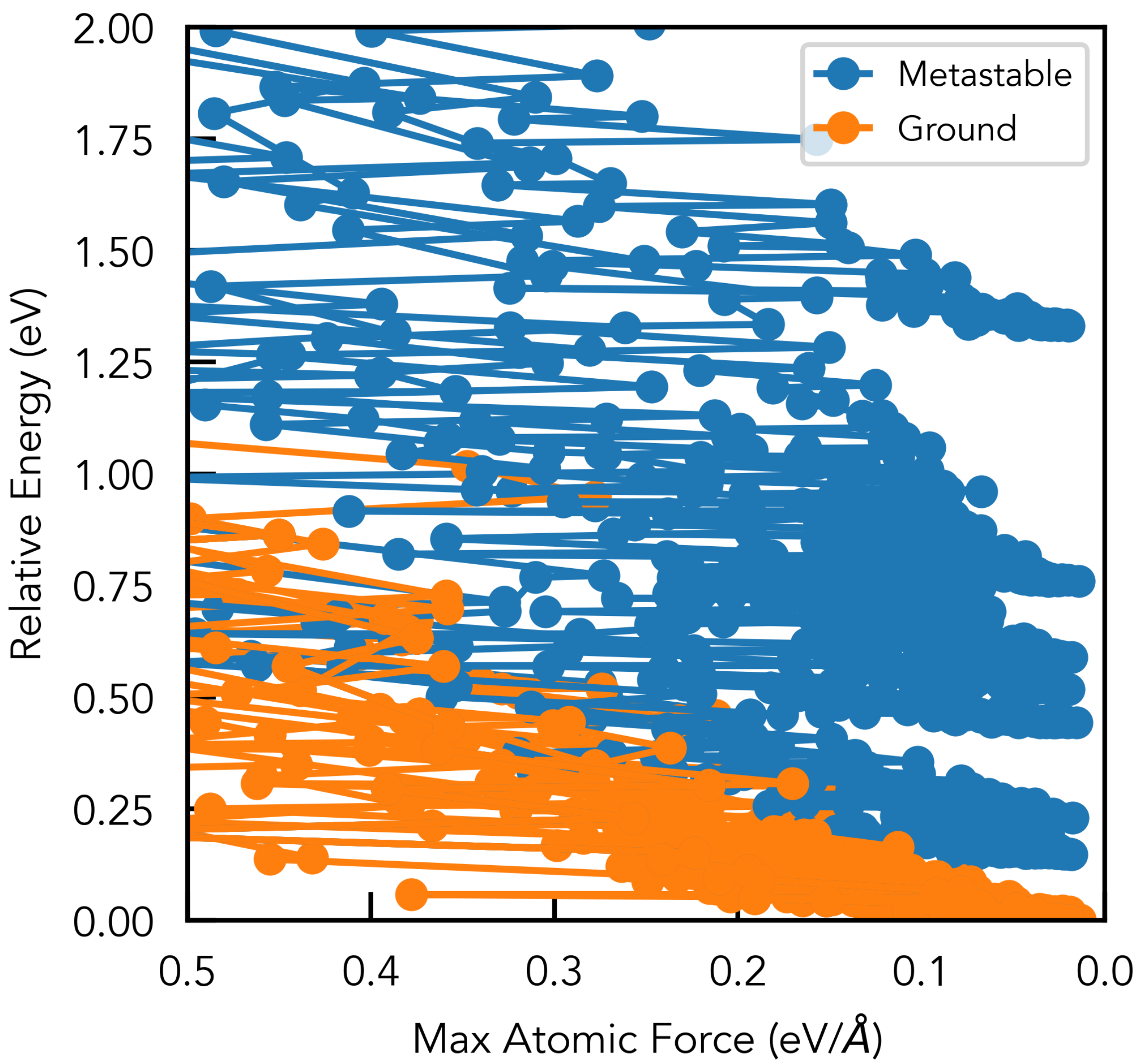


Again no real clear correlation

In [98]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o"
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Metastable", color="
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Ground", color="C1",
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o")
ax.invert_xaxis()
ax.set_xlim(0.5, 0)
ax.set_ylim(0, 2)
ax.legend()
ax.set_xlabel("Max Atomic Force (eV/$\AA$)")
ax.set_ylabel("Relative Energy (eV)")
```

Out[98]: Text(0, 0.5, 'Relative Energy (eV)')

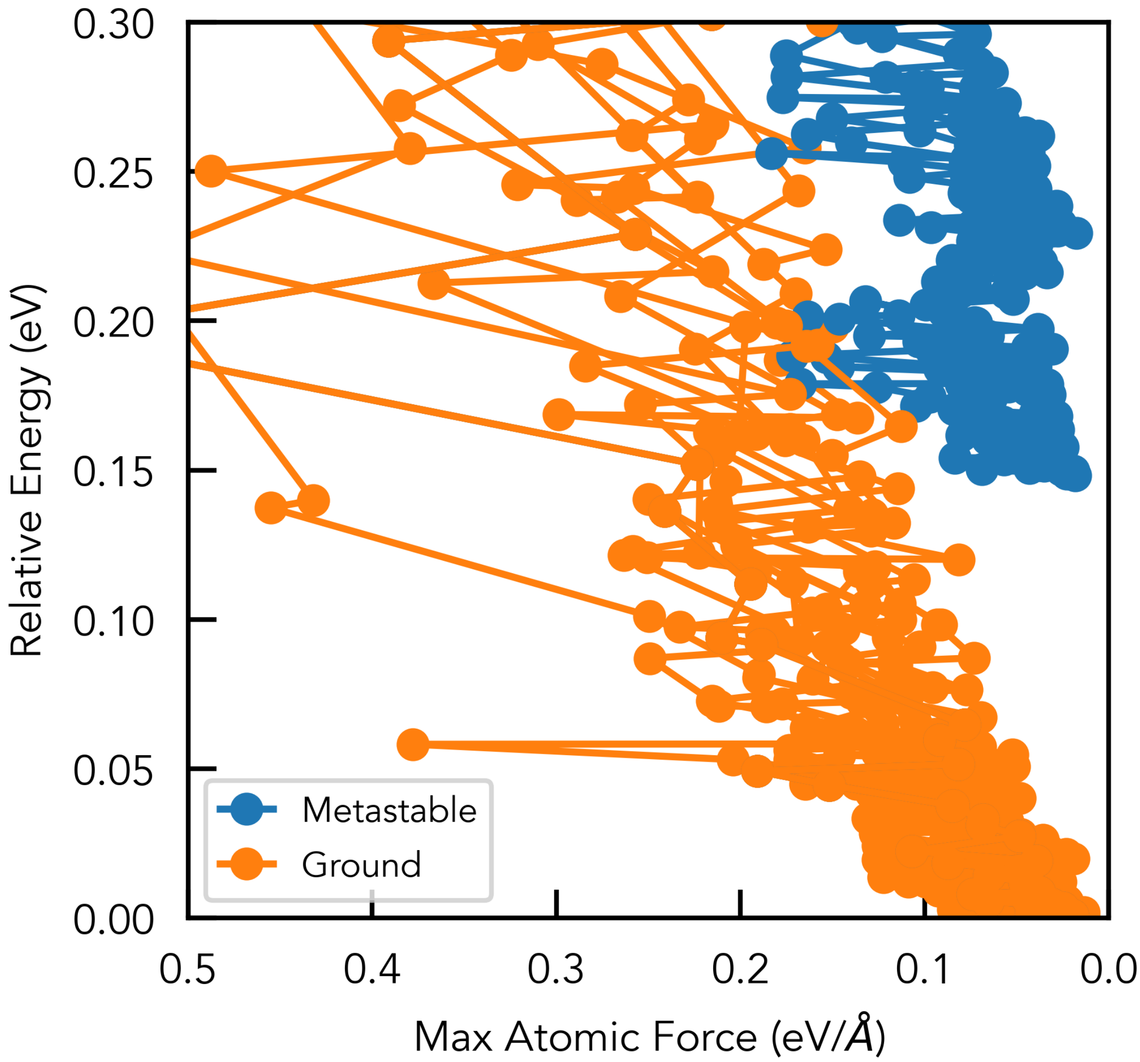


(Note several metastable states here)

In [99]:
```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o"
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Metastable", color="
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Ground", color="C1",
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o")
ax.invert_xaxis()
ax.set_xlim(0.5,0)
ax.set_ylim(0, 0.3)
ax.legend()
ax.set_xlabel("Max Atomic Force (eV/$\AA$)")
ax.set_ylabel("Relative Energy (eV)")
```

Out[99]:
```
Text(0, 0.5, 'Relative Energy (eV)')
```



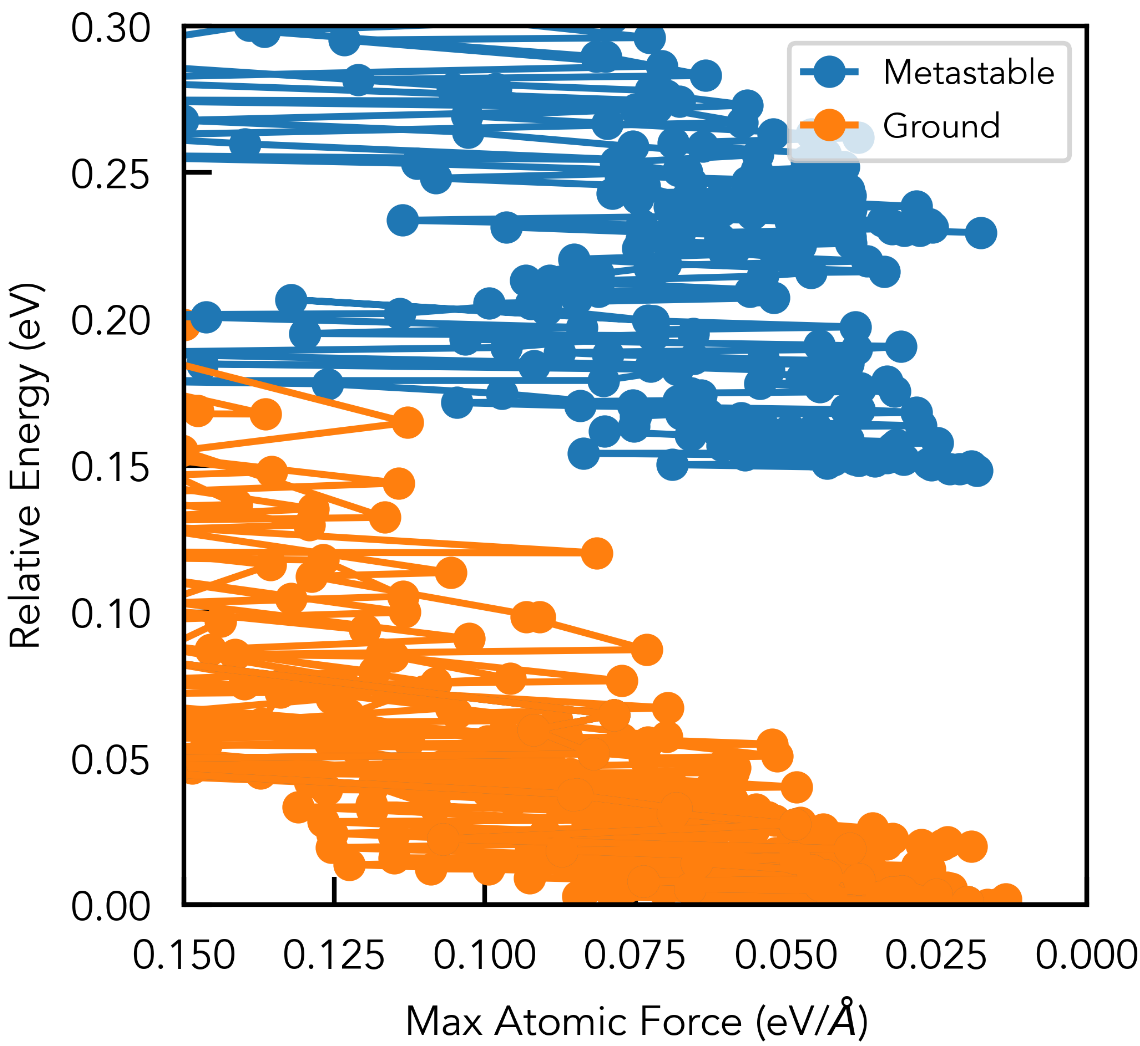
In [100…

```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o"
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Metastable", color="
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, label="Ground", color="C1",
ax.plot(df.Max_Norm_Force, df.Energy - e_ground, color=c, marker="o")
ax.invert_xaxis()
ax.set_xlim(0.15,0)
ax.set_ylim(0, 0.3)
ax.legend()
ax.set_xlabel("Max Atomic Force (eV/$\AA$)")
ax.set_ylabel("Relative Energy (eV)")
```

Out[100]:

```
Text(0, 0.5, 'Relative Energy (eV)')
```



Ok nice so again clear separation at `EDIFFG = -0.06` here, though actual energies not really converged till ~ `0.02`

In [102…

```
for i in os.listdir("Rattling_Tests/vac_1_Cd_0/BDM/"):
    if "json" in i:
        df = pd.read_json(f"Rattling_Tests/vac_1_Cd_0/BDM/{i}")
        idx_0pt4_force = min(df.index[df.Max_Norm_Force < 0.06].tolist())
        if df.Energy.iloc[-1] < e_ground+0.05:
            print(f"Groundstate force less than 0.06 eV/A at step: {idx_0pt4
        else:
            print(f"Metastable force less than 0.06 eV/A at step: {idx_0pt4_
```

```
Metastable force less than 0.06 eV/A at step: 25, out of 38 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 39, out of 51 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 41 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 38 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 25, out of 41 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 27, out of 41 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 28, out of 45 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 32, out of 44 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 22, out of 38 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 27, out of 38 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 25, out of 42 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 14, out of 25 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 35, out of 51 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 25, out of 44 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 31, out of 56 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 27, out of 41 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 29, out of 47 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 22, out of 35 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 35 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 39 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 37 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 37 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 31, out of 44 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 24, out of 38 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 28, out of 40 steps for 0.02 e
V/A
Metastable force less than 0.06 eV/A at step: 28, out of 41 steps for 0.02 e
V/A
```

Again we actually used `EDIFFG = -0.02` in our calcs anyway here

Again demonstrates that even a relatively small increase in `EDIFFG` does give a pretty significant reduction in required ionic steps (and thus calc cost)

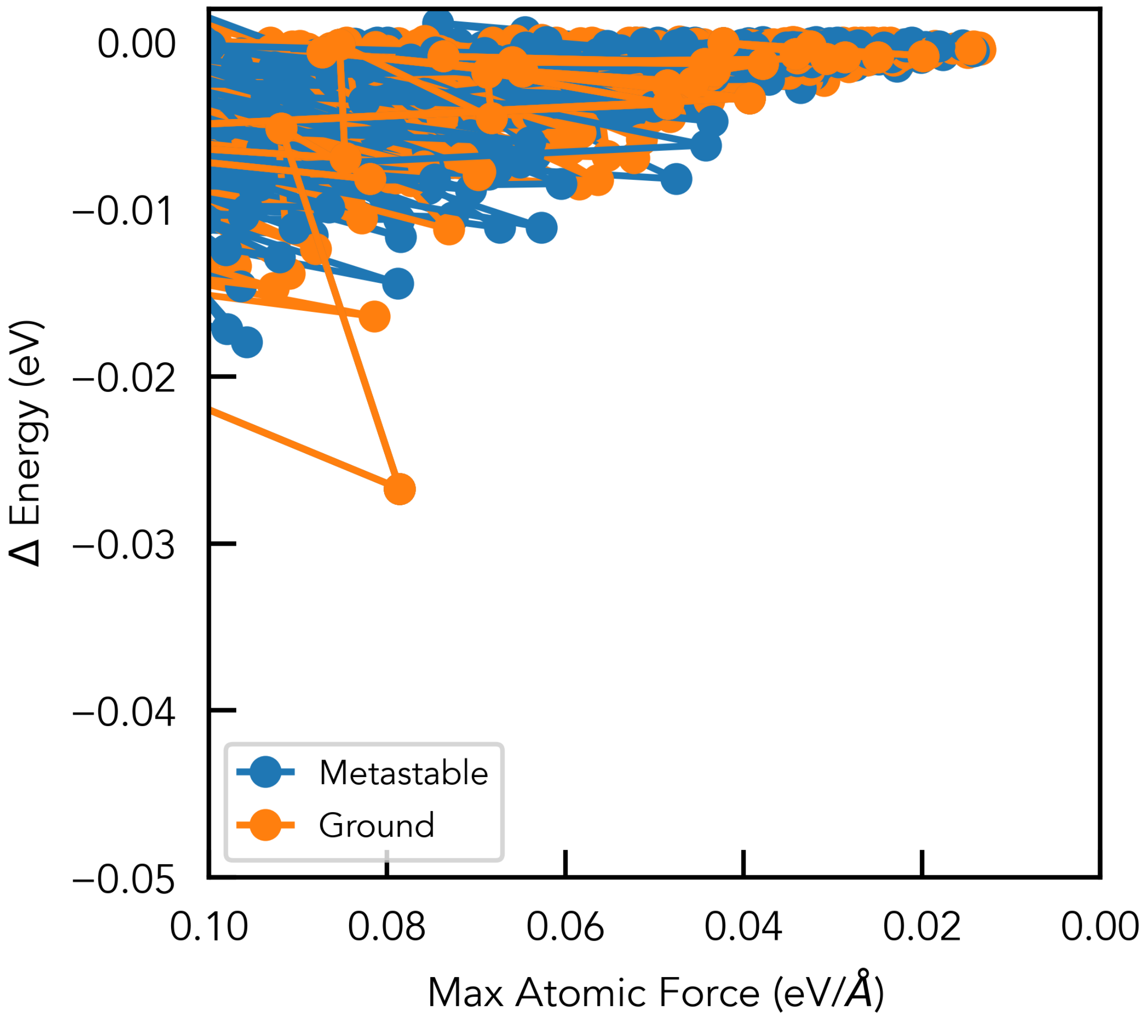
In [103…

```
f,ax = plt.subplots()

for i in os.listdir(json_dir):
    if "json" in i:
        df = pd.read_json(f"{json_dir}/{i}")
```

```
        if df.Energy.iloc[-1] < e_ground+0.05: # groundstate
            c = "C1"
        else:
            c = "C0"
        df.loc[0, "Delta_E"] = df.loc[0, "Energy"]
        for i in range(1, len(df)):
            df.loc[i, "Delta_E"] = df.loc[i, "Energy"] - df.loc[i-1, "Energy
        ax.plot(df.Max_Norm_Force, df.Delta_E, color=c, marker="o")
ax.plot(df.Max_Norm_Force, df.Delta_E, label="Metastable", color="C0", marke
ax.plot(df.Max_Norm_Force, df.Delta_E, label="Ground", color="C1", marker="c
ax.plot(df.Max_Norm_Force, df.Delta_E, color=c, marker="o")
ax.invert_xaxis()
ax.set_xlim(0.1,0)
ax.set_ylim(-0.05, 0.002)
ax.legend()
ax.set_xlabel("Max Atomic Force (eV/$\AA$)")
ax.set_ylabel("$\Delta$ Energy (eV)")
```

Out[103]:
```
Text(0, 0.5, '$\\Delta$ Energy (eV)')
```



Ok so yeah as expected, once all energy differences below ~0.01 eV (which happens around EDIFFG = -0.05) we're sufficiently converged to distinguish our groundstate structures.

In [ ]: