cov-matrix-processing

July 1, 2022

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.offsetbox import AnchoredText
%matplotlib inline
```

[2]:
```
n = 115
cov_nercome = np.loadtxt(f"../output/
 ↪Patchy_V6C_BOSS_DR12_NGC_z1_cov_nercome_{n}.matrix")
cov_sample = np.loadtxt(f"../output/Patchy_V6C_BOSS_DR12_NGC_z1_cov_sample_{n}.
 ↪matrix")

# 'Real' covariance matrix before processing.
# The term 'real' here indicates that this is the matrix that we use to compare
# our estimates to.
cov_real_preproc = np.loadtxt("../data/
 ↪C_2048_BOSS_DR12_NGC_z1_V6C_1_1_1_1_1_10_200_200_prerecon.matrix")
```

[3]:
```
print(cov_nercome)
print(cov_sample)
```

```
[[ 2.37349903e+08  1.38638149e+07  4.48436652e+05 …  9.97137860e+04
  -1.65641585e+05 -6.53831486e+04]
 [ 1.38638149e+07  5.84162471e+07  2.63751717e+06 …  3.17647865e+04
  -1.99745665e+04 -7.91663909e+04]
 [ 4.48436652e+05  2.63751717e+06  1.90414702e+07 …  2.37914001e+04
  -1.80217115e+04  1.72951449e+03]
 …
 [ 9.97137860e+04  3.17647865e+04  2.37914001e+04 …  1.95156160e+04
   3.76694250e+03  1.96116125e+03]
 [-1.65641585e+05 -1.99745665e+04 -1.80217115e+04 …  3.76694250e+03
   1.96850777e+04  3.85141521e+03]
 [-6.53831486e+04 -7.91663909e+04  1.72951449e+03 …  1.96116125e+03
   3.85141521e+03  1.84239586e+04]]
[[ 2.27110517e+08  1.24256921e+07  8.13417570e+05 …  1.01764188e+05
  -1.53971047e+05 -6.29215070e+04]
 [ 1.24256921e+07  4.90834998e+07  2.14455817e+06 …  2.52465463e+04
  -1.75551841e+04 -6.61119493e+04]
 [ 8.13417570e+05  2.14455817e+06  1.49413602e+07 …  1.89908195e+04
```

```
 -1.42352841e+04  1.27802892e+00]
…
[ 1.01764188e+05  2.52465463e+04  1.89908195e+04 …  4.68670653e+03
  1.59301659e+03  9.06353593e+02]
[-1.53971047e+05 -1.75551841e+04 -1.42352841e+04 …  1.59301659e+03
  4.44370058e+03  1.86719508e+03]
[-6.29215070e+04 -6.61119493e+04  1.27802892e+00 …  9.06353593e+02
  1.86719508e+03  4.19185250e+03]]
```

[4]:
```
indices = np.concatenate((np.arange(40), np.arange(40)+80, np.arange(40)+160))
cov_real = (cov_real_preproc[indices, :])[:, indices]
```

[5]:
```
print(cov_real)
```

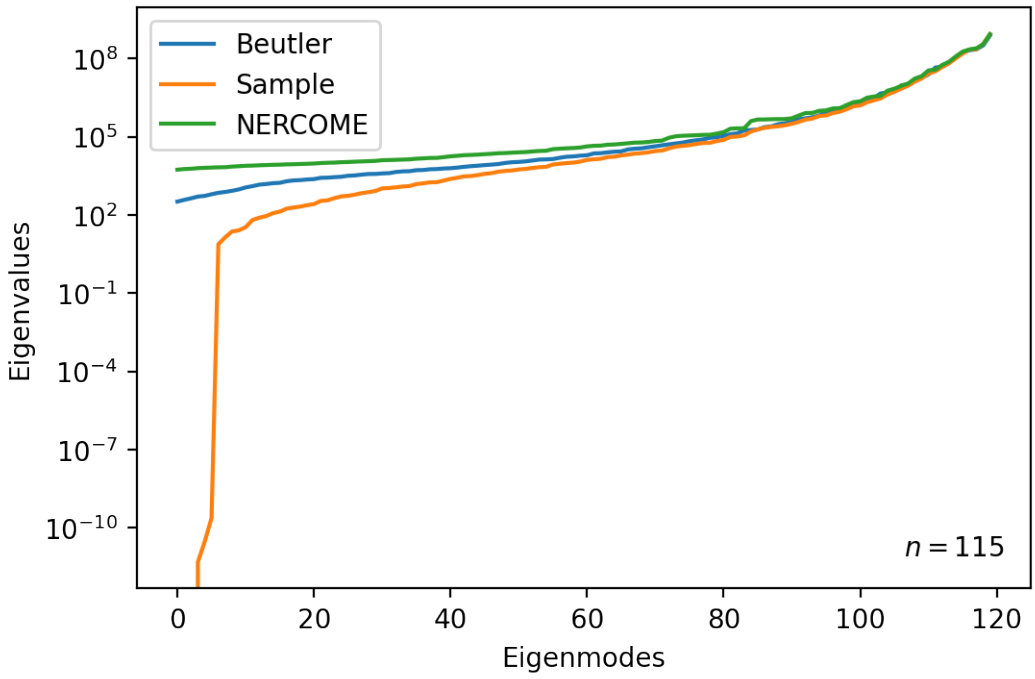
```
[[ 2.39531896e+08  1.91051563e+07  1.44038639e+05 …  9.68083967e+03
  -1.21684291e+04  2.21567909e+04]
 [ 1.91051563e+07  5.15256710e+07  4.78376836e+06 …  9.13608077e+03
   9.26311893e+03 -1.41639754e+04]
 [ 1.44038639e+05  4.78376836e+06  1.67772970e+07 … -5.49127765e+02
  -2.28200815e+03  6.76773412e+03]
…
 [ 9.68083967e+03  9.13608077e+03 -5.49127765e+02 …  4.22979902e+03
   1.28931823e+03  3.78440184e+02]
 [-1.21684291e+04  9.26311893e+03 -2.28200815e+03 …  1.28931823e+03
   3.88691287e+03  1.00320951e+03]
 [ 2.21567909e+04 -1.41639754e+04  6.76773412e+03 …  3.78440184e+02
   1.00320951e+03  3.62984716e+03]]
```

[6]:
```
evals_nercome = np.linalg.eigvalsh(cov_nercome)
evals_sample = np.linalg.eigvalsh(cov_sample)
evals_real = np.linalg.eigvalsh(cov_real)
```

[7]:
```
plt.figure(dpi=200)
plt.plot(evals_real, label="Beutler")
plt.plot(evals_sample, label="Sample")
plt.plot(evals_nercome, label="NERCOME")
plt.yscale("log")
plt.legend()
plt.xlabel("Eigenmodes")
plt.ylabel("Eigenvalues")

parameters = AnchoredText(fr"$n = {n}$", frameon=False, loc="lower right",␣
 ↪pad=0.5)
plt.setp(parameters.patch, facecolor='white', alpha=0.5)
plt.gca().add_artist(parameters)
```

[7]:
```
<matplotlib.offsetbox.AnchoredText at 0x7fac988c58b0>
```



[8]:
```
nercome_rel_errors = (cov_nercome-cov_real)/np.abs(cov_real)
print(nercome_rel_errors)
```

```
[[-9.10940713e-03 -2.74341715e-01  2.11330805e+00 …  9.30011749e+00
  -1.26124050e+01 -3.95093043e+00]
 [-2.74341715e-01  1.33730933e-01 -4.48652825e-01 …  2.47685044e+00
  -3.15635432e+00 -4.58927764e+00]
 [ 2.11330805e+00 -4.48652825e-01  1.34954589e-01 …  4.43258007e+01
  -6.89730374e+00 -7.44447040e-01]
 …
 [ 9.30011749e+00  2.47685044e+00  4.43258007e+01 …  3.61384001e+00
   1.92165457e+00  4.18222254e+00]
 [-1.26124050e+01 -3.15635432e+00 -6.89730374e+00 …  1.92165457e+00
   4.06445046e+00  2.83909358e+00]
 [-3.95093043e+00 -4.58927764e+00 -7.44447040e-01 …  4.18222254e+00
   2.83909358e+00  4.07568439e+00]]
```

[9]:
```
nercome_pos_count = 0
nercome_neg_count = 0
for row in nercome_rel_errors:
    for error in row:
        if error >= 0:
            nercome_pos_count += 1
        elif error < 0:
            nercome_neg_count += 1
```

```
print(f"NERCOME overestimated {nercome_pos_count} elements")
print(f"NERCOME underestimated {nercome_neg_count} elements")

index_max_nercome = np.unravel_index(np.abs(nercome_rel_errors).argmax(),␣
 ↪nercome_rel_errors.shape)

print(f"Maximum relative error is {nercome_rel_errors[index_max_nercome]}")
print(f"Maximum relative error index is {index_max_nercome}")
print(f"NERCOME: {cov_nercome[index_max_nercome]}, real:␣
 ↪{cov_real[index_max_nercome]}")
```

```
NERCOME overestimated 7659 elements
NERCOME underestimated 6741 elements
Maximum relative error is -11586.203487254506
Maximum relative error index is (1, 12)
NERCOME: -114552.34572310268, real: -9.88610805438137
```

[10]:
```
sample_rel_errors = (cov_sample-cov_real)/np.abs(cov_real)
print(sample_rel_errors)
```

```
[[ -0.05185689  -0.34961579   4.64721784 …   9.51191749 -11.65332163
   -3.83982943]
 [ -0.34961579  -0.04739718  -0.55170108 …   1.76338913  -2.8951699
   -3.66761255]
 [  4.64721784  -0.55170108  -0.10942983 …  35.58360818  -5.2380514
   -0.99981116]
 …
 [  9.51191749   1.76338913  35.58360818 …   0.10802109   0.23554958
    1.39497186]
 [-11.65332163  -2.8951699   -5.2380514  …   0.23554958   0.14324677
    0.86122146]
 [ -3.83982943  -3.66761255  -0.99981116 …   1.39497186   0.86122146
    0.15482893]]
```

[11]:
```
sample_pos_count = 0
sample_neg_count = 0
for row in sample_rel_errors:
    for error in row:
        if error >= 0:
            sample_pos_count += 1
        elif error < 0:
            sample_neg_count += 1

print(f"Sample overestimated {sample_pos_count} elements")
print(f"Sample underestimated {sample_neg_count} elements")
```

```
index_max_sample = np.unravel_index(np.abs(sample_rel_errors).argmax(),␣
 ↪sample_rel_errors.shape)

print(f"Maximum relative error is {sample_rel_errors[index_max_sample]}")
print(f"Maximum relative error index is {index_max_sample}")
print(f"Sample: {cov_sample[index_max_sample]}, real:␣
 ↪{cov_real[index_max_sample]}")
```

```
Sample overestimated 6843 elements
Sample underestimated 7557 elements
Maximum relative error is -9802.71847383865
Maximum relative error index is (1, 12)
Sample: -96920.62016710371, real: -9.88610805438137
```

[12]:
```
MSE_NERCOME = np.trace((cov_nercome-cov_real)@(cov_nercome-cov_real).T)
MSE_sample = np.trace((cov_sample-cov_real)@(cov_sample-cov_real).T)
print(f"MSE NERCOME: {MSE_NERCOME}")
print(f"MSE sample: {MSE_sample}")
```

```
MSE NERCOME: 6.310561645809585e+16
MSE sample: 6.35384882148548e+16
```

[ ]:
```

```