cov-matrix-processing

July 19, 2022

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

[2]:
```
n = 250
cov_nercome = np.loadtxt(f"../output/data/
 ↪Patchy_V6C_BOSS_DR12_NGC_z1_cov_nercome_{n}_avg10000.matrix")
cov_sample = np.loadtxt(f"../output/data/
 ↪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")
```

```
---------------------------------------------------------------------------
OSError                                   Traceback (most recent call last)
<ipython-input-2-37c9f68b0221> in <module>
      1 n = 250
----> 2 cov_nercome = np.loadtxt(f"../output/data/
 ↪Patchy_V6C_BOSS_DR12_NGC_z1_cov_nercome_{n}_avg10000.matrix")
      3 cov_sample = np.loadtxt(f"../output/data/
 ↪Patchy_V6C_BOSS_DR12_NGC_z1_cov_sample_{n}.matrix")
      4
      5 # 'Real' covariance matrix before processing.

~/opt/anaconda3/lib/python3.8/site-packages/numpy/lib/npyio.py in loadtxt(fname␣
 ↪dtype, comments, delimiter, converters, skiprows, usecols, unpack, ndmin,␣
 ↪encoding, max_rows)
    979             fname = os_fspath(fname)
    980         if _is_string_like(fname):
--> 981             fh = np.lib._datasource.open(fname, 'rt', encoding=encoding
    982             fencoding = getattr(fh, 'encoding', 'latin1')
    983             fh = iter(fh)
```

```
~/opt/anaconda3/lib/python3.8/site-packages/numpy/lib/_datasource.py in␣
 ↪open(path, mode, destpath, encoding, newline)
    267
    268     ds = DataSource(destpath)
--> 269     return ds.open(path, mode, encoding=encoding, newline=newline)
    270
    271

~/opt/anaconda3/lib/python3.8/site-packages/numpy/lib/_datasource.py in␣
 ↪open(self, path, mode, encoding, newline)
    621                                         encoding=encoding, newline=newlin )
    622         else:
--> 623             raise IOError("%s not found." % path)
    624
    625

OSError: ../output/data/Patchy_V6C_BOSS_DR12_NGC_z1_cov_nercome_250_avg10000.
 ↪matrix not found.
```

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

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

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

[ ]:
```
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}")
```

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

[ ]:
```
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"$n = {n}$" "\n"
```

```
    fr"MSE NERCOME $= {np.format_float_scientific(MSE_NERCOME, precision=3)}$"␣
 ↪"\n"
    fr"MSE Sample $={np.format_float_scientific(MSE_sample, precision=3)}$",
    frameon=False, loc="lower right", pad=0.5)
plt.setp(parameters.patch, facecolor='white', alpha=0.5)
plt.gca().add_artist(parameters)
```

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

[ ]:
```
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]}")
```

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

[ ]:
```
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]}")
```

[ ]: