```
reference 1.000e+00   nan            TeV nan nan   True
  lambda_ 2.384e+00   nan          TeV-1 nan nan  False
    alpha 1.000e+00   nan                nan nan   True
   name     value   error      unit      min max frozen
--------- --------- ----- -------------- --- --- ------
    index 2.258e+00   nan                nan nan  False
amplitude 1.123e-12   nan cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00   nan            TeV nan nan   True
  lambda_ 4.532e-01   nan          TeV-1 nan nan  False
    alpha 1.000e+00   nan                nan nan   True
   name     value   error      unit      min max frozen
--------- --------- ----- -------------- --- --- ------
    index 2.365e+00   nan                nan nan  False
amplitude 8.448e-13   nan cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00   nan            TeV nan nan   True
  lambda_ 1.827e-01   nan          TeV-1 nan nan  False
    alpha 1.000e+00   nan                nan nan   True
   name     value   error      unit      min max frozen
--------- --------- ----- -------------- --- --- ------
    index 2.293e+00   nan                nan nan  False
amplitude 1.105e-12   nan cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00   nan            TeV nan nan   True
  lambda_ 4.638e-01   nan          TeV-1 nan nan  False
    alpha 1.000e+00   nan                nan nan   True
   name     value   error      unit      min max frozen
--------- --------- ----- -------------- --- --- ------
    index 2.438e+00   nan                nan nan  False
amplitude 8.578e-13   nan cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00   nan            TeV nan nan   True
  lambda_ 2.039e-01   nan          TeV-1 nan nan  False
    alpha 1.000e+00   nan                nan nan   True
   name     value   error      unit      min max frozen
--------- --------- ----- -------------- --- --- ------
    index 2.380e+00   nan                nan nan  False
amplitude 8.799e-13   nan cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00   nan            TeV nan nan   True
  lambda_ 3.919e-01   nan          TeV-1 nan nan  False
    alpha 1.000e+00   nan                nan nan   True
```

[19]:

```
index = [[0 for i in range(cols)] for j in range(rows)]
amplitude = [[0 for i in range(cols)] for j in range(rows)]
reference = [[0 for i in range(cols)] for j in range(rows)]
lambda_ = [[0 for i in range(5)] for j in range(rows)]
alpha = [[0 for i in range(5)] for j in range(rows)]
covar = [[0 for i in range(cols)] for j in range(rows)]

for i in range(8):
```

```
    a = np.array([_["index"] for _ in res[i][0]])
    b = np.array([_["amplitude"] for _ in res[i][0]])
    c = np.array([_["reference"] for _ in res[i][0]])
    index[i][0]=a
    amplitude[i][0]=b
    reference[i][0]=c
for j in range(5):
    for i in range(8):
        a = np.array([_["index"] for _ in res[i][j+1]])
        b = np.array([_["amplitude"] for _ in res[i][j+1]])
        c = np.array([_["reference"] for _ in res[i][j+1]])
        d = np.array([_["lambda_"] for _ in res[i][j+1]])
        e = np.array([_["alpha"] for _ in res[i][j+1]])
        index[i][j+1]=a
        amplitude[i][j+1]=b
        reference[i][j+1]=c
        lambda_[i][j]=d
        alpha[i][j]=e

mu = [[0 for i in range(cols)] for j in range(rows)]
sigma = [[0 for i in range(cols)] for j in range(rows)]

for i in range(8):
    x = np.array([index[i][0], amplitude[i][0], reference[i][0]])
    covar[i][0]=np.cov(x)
for j in range(5):
    for i in range(8):
        x = np.array([index[i][j+1], amplitude[i][j+1], reference[i][j+1],␣
↪lambda_[i][j], alpha[i][j]])
        covar[i][j+1]=np.cov(x)
```

[20]:
```
def evaluate_err(self, covar, energy, epsilon=1e-4):


        p_cov = covar
        eps = np.sqrt(np.diag(covar)) * epsilon

        df_dp = self._evaluate_gradient(energy, eps)
        f_cov = df_dp.T @ p_cov @ df_dp
        f_err = np.sqrt(np.diagonal(f_cov))

        q = self(energy)
        return u.Quantity([q.value, f_err], unit=q.unit)
```

[21]:
```
def plot_error(
        self,
        covar,
        energy_range,
```

```
        ax=None,
        energy_unit="TeV",
        flux_unit="cm-2 s-1 TeV-1",
        energy_power=0,
        n_points=100,
        **kwargs,
    ):

        ax = plt.gca() if ax is None else ax

        kwargs.setdefault("facecolor", "black")
        kwargs.setdefault("alpha", 0.2)
        kwargs.setdefault("linewidth", 0)

        emin, emax = energy_range
        energy = MapAxis.from_energy_bounds(emin, emax, n_points, energy_unit).
↪edges

        flux, flux_err = evaluate_err(self,covar,energy).to(flux_unit)

        y_lo = self._plot_scale_flux(energy, flux - flux_err, energy_power)
        y_hi = self._plot_scale_flux(energy, flux + flux_err, energy_power)

        where = (energy >= energy_range[0]) & (energy <= energy_range[1])
        ax.fill_between(energy.value, y_lo.value, y_hi.value, where=where,␣
↪**kwargs)

        self._plot_format_ax(ax, energy, y_lo, energy_power)
        return ax
```

[22]:
```
for j in range(6):
    print(f"model: {j}")
    for i in range(8):
        mu[i][j]=index[i][j].mean()
        sigma[i][j]=index[i][j].std()
        print(f"index: {index[i][j].mean()} += {index[i][j].std()}")
```

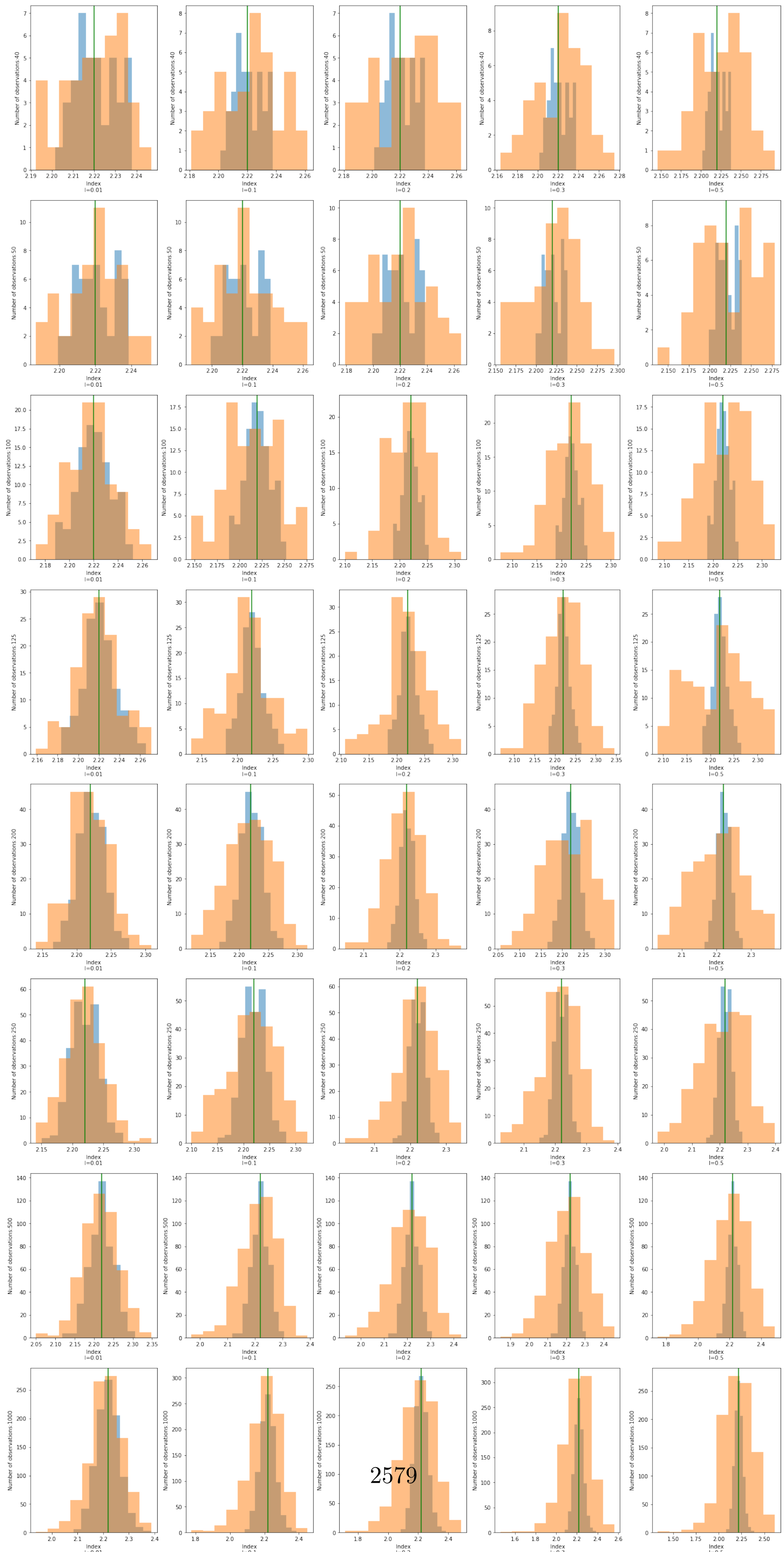
```
model: 0
index: 2.2208519415872168 += 0.009464742485055524
index: 2.2204071142702646 += 0.010643317914159624
index: 2.2203004860671913 += 0.014097739094078581
index: 2.220852816179499 += 0.016172109015838997
index: 2.2212194274890815 += 0.019215872375112058
index: 2.2199563069339674 += 0.022096616282109238
index: 2.2190755629793766 += 0.030988506389346652
index: 2.219433716740164 += 0.045081260977561495
model: 1
```

```
index: 2.220656261583631 += 0.013657305077428659
index: 2.219537607489587 += 0.015543153825233458
index: 2.2181176846218893 += 0.01893646593082573
index: 2.218481623164424 += 0.021270851790616947
index: 2.2185111525340933 += 0.02859104651839322
index: 2.218542656742381 += 0.03184761115120899
index: 2.216313736296541 += 0.04616541831378384
index: 2.20980342983713 += 0.0651520272855002
model: 2
index: 2.222668524556452 += 0.020071449729122202
index: 2.2225594753189397 += 0.01948890190403934
index: 2.2147408949865435 += 0.02941353971833464
index: 2.2158534947511623 += 0.03251658315774923
index: 2.216204303331999 += 0.038170143005702
index: 2.217125610531704 += 0.04520954508481233
index: 2.2125022241933583 += 0.0655338112407473
index: 2.200000534804982 += 0.09470243949549197
model: 3
index: 2.2236501416670293 += 0.022047323393647677
index: 2.219931292887791 += 0.021844555108037105
index: 2.2164016539031604 += 0.03509131909554301
index: 2.2179518489755656 += 0.038741370636685424
index: 2.2139994033069126 += 0.04955929204098608
index: 2.2166383570163797 += 0.05635295631011011
index: 2.210627501567632 += 0.08049310492882605
index: 2.1968224679666557 += 0.11652974737637158
model: 4
index: 2.222868905446897 += 0.024893832166725735
index: 2.2206572508669185 += 0.03163832107443028
index: 2.217058819379629 += 0.04196922244707493
index: 2.217765264430175 += 0.04689949278742754
index: 2.212486892142441 += 0.05482713790059148
index: 2.2138895180984646 += 0.062036070041959775
index: 2.2077430921248653 += 0.09777323893277609
index: 2.19755631012565 += 0.13442235594097948
model: 5
index: 2.224782058052607 += 0.02979815009186981
index: 2.223190858084319 += 0.032993897346219714
index: 2.2222979638115996 += 0.050745870934894716
index: 2.2157646774812667 += 0.06200991688952796
index: 2.2068252015013847 += 0.07342614501247902
index: 2.209039743769255 3 += 0.0833158708555063
index: 2.2001547172563996 += 0.1169845131986335
index: 2.1982709354485923 += 0.16994398733576785
```

[23]:
```
fig = plt.figure(figsize=[20,40],constrained_layout=True)
```

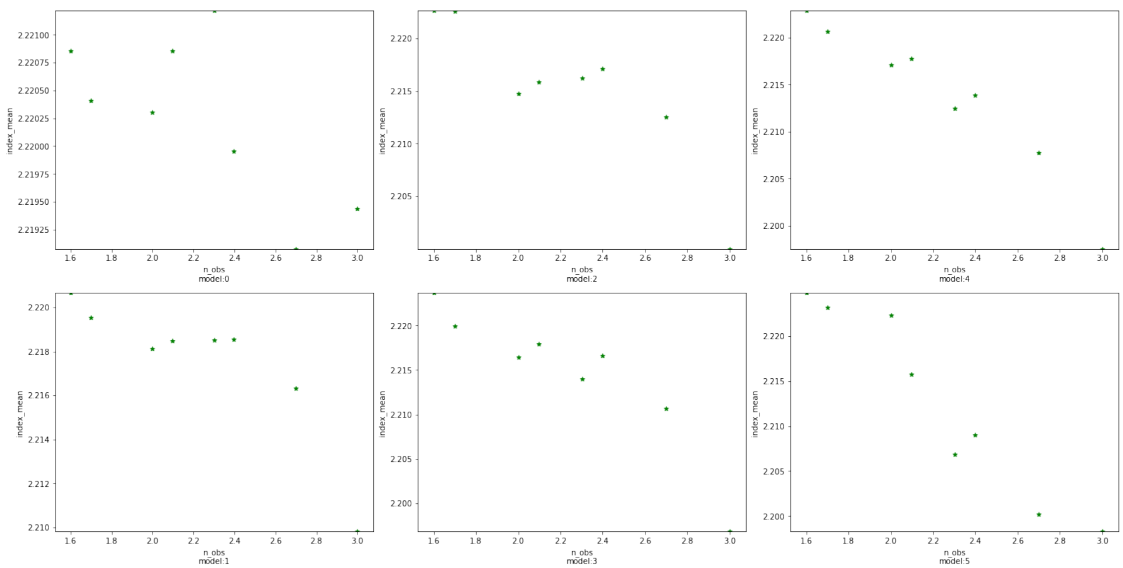
```python
import matplotlib.gridspec as gridspec

gs0 = gridspec.GridSpec(1, 5, figure=fig)

gs1 = gridspec.GridSpecFromSubplotSpec(8, 1, subplot_spec=gs0[0])
for n in range(8):
    ax = fig.add_subplot(gs1[n])
    plt.hist(index[n][0], bins=10, alpha=0.5)
    plt.hist(index[n][1], bins=10, alpha=0.5)
    plt.axvline(x=model_simu.parameters["index"].value, color="green")
    plt.xlabel('Index\nl=0.01')
    plt.ylabel(f'Number of observations:{n_obs[n]}')

gs2 = gridspec.GridSpecFromSubplotSpec(8, 1, subplot_spec=gs0[1])
for n in range(8):
    ax = fig.add_subplot(gs2[n])
    plt.hist(index[n][0], bins=10, alpha=0.5)
    plt.hist(index[n][2], bins=10, alpha=0.5)
    plt.axvline(x=model_simu.parameters["index"].value, color="green")
    plt.xlabel('Index\nl=0.1')
    plt.ylabel(f'Number of observations:{n_obs[n]}')

gs3 = gridspec.GridSpecFromSubplotSpec(8, 1, subplot_spec=gs0[2])
for n in range(8):
    ax = fig.add_subplot(gs3[n])
    plt.hist(index[n][0], bins=10, alpha=0.5)
    plt.hist(index[n][3], bins=10, alpha=0.5)
    plt.axvline(x=model_simu.parameters["index"].value, color="green")
    plt.xlabel('Index\nl=0.2')
    plt.ylabel(f'Number of observations:{n_obs[n]}')

gs4 = gridspec.GridSpecFromSubplotSpec(8, 1, subplot_spec=gs0[3])
for n in range(8):
    ax = fig.add_subplot(gs4[n])
    plt.hist(index[n][0], bins=10, alpha=0.5)
    plt.hist(index[n][4], bins=10, alpha=0.5)
    plt.axvline(x=model_simu.parameters["index"].value, color="green")
    plt.xlabel('Index\nl=0.3')
    plt.ylabel(f'Number of observations:{n_obs[n]}')

gs5 = gridspec.GridSpecFromSubplotSpec(8, 1, subplot_spec=gs0[4])
for n in range(8):
    ax = fig.add_subplot(gs5[n])
    plt.hist(index[n][0], bins=10, alpha=0.5)
    plt.hist(index[n][5], bins=10, alpha=0.5)
    plt.axvline(x=model_simu.parameters["index"].value, color="green")
    plt.xlabel('Index\nl=0.5')
```

```
    plt.ylabel(f'Number of observations:{n_obs[n]}')

plt.show()
```


Number of observations 40
Number of observations 50
Number of observations 100
Number of observations 125
Number of observations 200
Number of observations 250
Number of observations 500
Number of observations 1000
Index
l=0.01
l=0.1
l=0.2
l=0.3
l=0.5

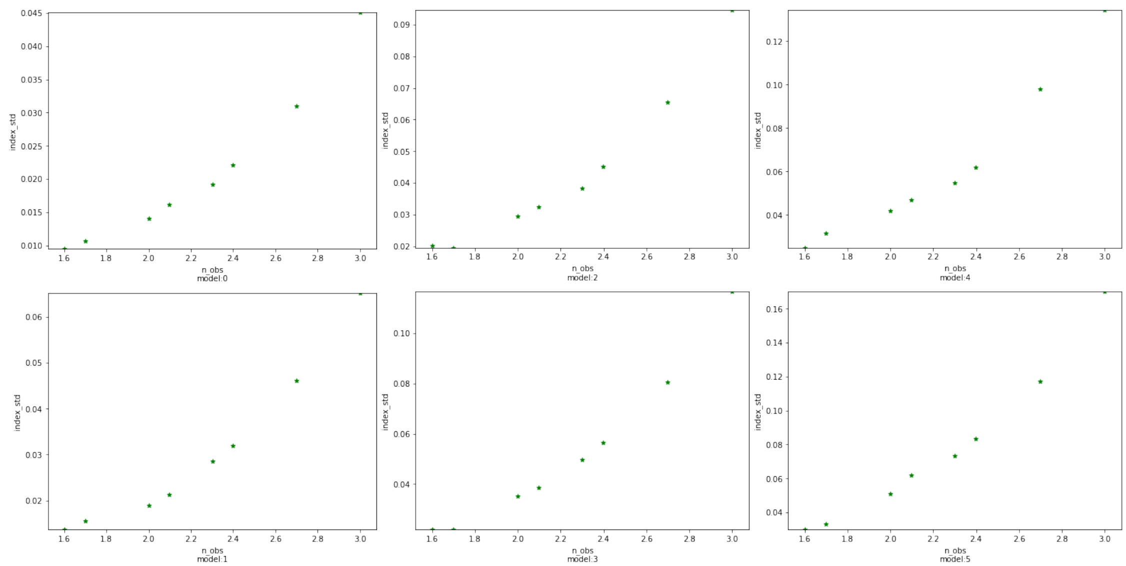
[24]:

```python
fig = plt.figure(figsize=[20,10],constrained_layout=True)

import matplotlib.gridspec as gridspec

gs0 = gridspec.GridSpec(1, 3, figure=fig)

gs1 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[0])
for n in range(2):
    ax = fig.add_subplot(gs1[n])
    plt.scatter(np.log10(n_obs), [row[n] for row in mu], label= "stars", color=␣
↪"green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n}')
    plt.ylabel('index_mean')
    plt.ylim(min([row[n] for row in mu]),max([row[n] for row in mu]))

gs2 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[1])
for n in range(2):
    ax = fig.add_subplot(gs2[n])
    plt.scatter(np.log10(n_obs), [row[n+2] for row in mu], label= "stars",␣
↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+2}')
    plt.ylabel('index_mean')
    plt.ylim(min([row[n+2] for row in mu]),max([row[n+2] for row in mu]))

gs3 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[2])
for n in range(2):
    ax = fig.add_subplot(gs3[n])
    plt.scatter(np.log10(n_obs), [row[n+4] for row in mu], label= "stars",␣
↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+4}')
    plt.ylabel('index_mean')
    plt.ylim(min([row[n+4] for row in mu]),max([row[n+4] for row in mu]))

plt.show()
```




```
[25]: fig = plt.figure(figsize=[20,10],constrained_layout=True)

import matplotlib.gridspec as gridspec

gs0 = gridspec.GridSpec(1, 3, figure=fig)

gs1 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[0])
for n in range(2):
    ax = fig.add_subplot(gs1[n])
    plt.scatter(np.log10(n_obs), [row[n] for row in sigma], label= "stars",␣
 ↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n}')
    plt.ylabel('index_std')
    plt.ylim(min([row[n] for row in sigma]),max([row[n] for row in sigma]))

gs2 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[1])
for n in range(2):
    ax = fig.add_subplot(gs2[n])
    plt.scatter(np.log10(n_obs), [row[n+2] for row in sigma], label= "stars",␣
 ↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+2}')
    plt.ylabel('index_std')
    plt.ylim(min([row[n+2] for row in sigma]),max([row[n+2] for row in sigma]))

gs3 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[2])
for n in range(2):
```

```
    ax = fig.add_subplot(gs3[n])
    plt.scatter(np.log10(n_obs), [row[n+4] for row in sigma], label= "stars",␣
 ↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+4}')
    plt.ylabel('index_std')
    plt.ylim(min([row[n+4] for row in sigma]),max([row[n+4] for row in sigma]))

plt.show()
```



[26]:

```
for j in range(6):
    print(f"model: {j}")
    for i in range(8):
        mu[i][j]=amplitude[i][j].mean()
        sigma[i][j]=amplitude[i][j].std()
        print(f"amplitude: {amplitude[i][j].mean()} += {amplitude[i][j].std()}")
```

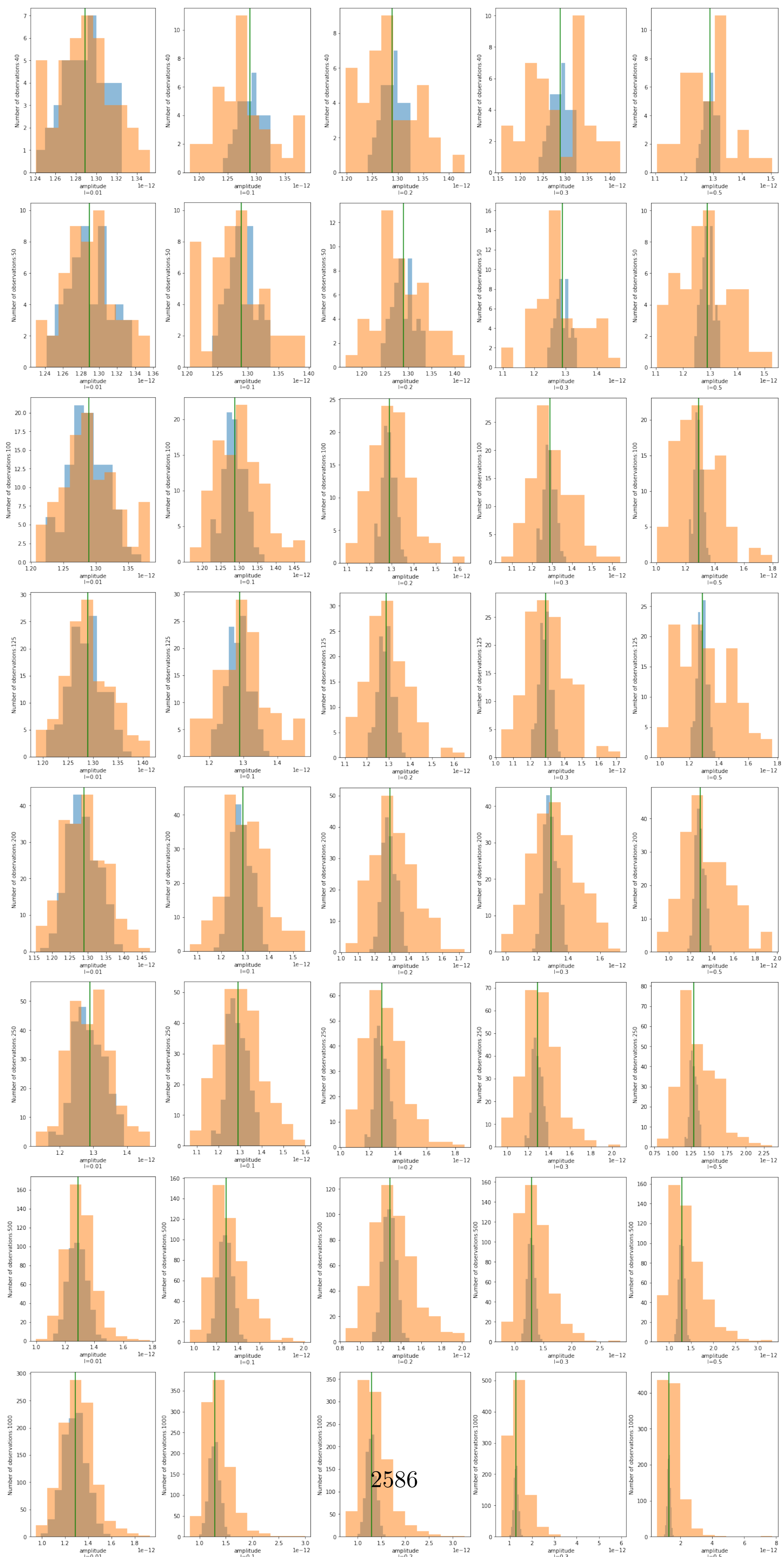
```
model: 0
amplitude: 1.2882109734645357e-12 += 2.020774416400257e-14
amplitude: 1.2887930617364896e-12 += 2.321243734503526e-14
amplitude: 1.2875057637201671e-12 += 2.943738660653206e-14
amplitude: 1.2855551328222816e-12 += 3.4620322478409695e-14
amplitude: 1.2852616877373924e-12 += 4.1590131875188173e-14
amplitude: 1.2870704254498414e-12 += 4.604036230233484e-14
amplitude: 1.2848003538014906e-12 += 6.382360983727948e-14
amplitude: 1.2834022687629476e-12 += 9.812239252655629e-14
model: 1
amplitude: 1.2889578883338907e-12 += 2.7427021702990073e-14
amplitude: 1.2894684378817922e-12 += 2.813899369747791e-14
```

```
amplitude: 1.2903669824825752e-12 += 4.2053504202932235e-14
amplitude: 1.2893540181865374e-12 += 4.768668915877586e-14
amplitude: 1.2898703913381476e-12 += 6.143963128087334e-14
amplitude: 1.2890129642059102e-12 += 6.224726161695642e-14
amplitude: 1.2967926229286649e-12 += 9.656037935907711e-14
amplitude: 1.3095492576963572e-12 += 1.3837863098665463e-13
model: 2
amplitude: 1.280073301077982e-12 += 4.911939253338424e-14
amplitude: 1.2818934446481758e-12 += 4.582631523140429e-14
amplitude: 1.3014742975846328e-12 += 6.449683264815556e-14
amplitude: 1.2993266293032354e-12 += 7.372640662452622e-14
amplitude: 1.3003213530204016e-12 += 8.916511962822823e-14
amplitude: 1.2985776217073962e-12 += 9.838689427012748e-14
amplitude: 1.3147006361628989e-12 += 1.5577758987136388e-13
amplitude: 1.3501367000923008e-12 += 2.4075702300659865e-13
model: 3
amplitude: 1.2828685431425812e-12 += 5.4293606809276975e-14
amplitude: 1.2871597301817917e-12 += 5.358895374105969e-14
amplitude: 1.2991825028209179e-12 += 8.860020993480923e-14
amplitude: 1.2951170945882896e-12 += 9.64975123401984e-14
amplitude: 1.3101894810993303e-12 += 1.238184067952117e-13
amplitude: 1.3068629610758982e-12 += 1.3662832369719393e-13
amplitude: 1.3290301800221315e-12 += 2.0686240490168188e-13
amplitude: 1.3738417902790715e-12 += 3.2228253156255256e-13
model: 4
amplitude: 1.2829487010062054e-12 += 6.819941025025066e-14
amplitude: 1.2842954246614235e-12 += 7.981240973194117e-14
amplitude: 1.2966408475096802e-12 += 1.0137970787848202e-13
amplitude: 1.3005923194395004e-12 += 1.2235363701347066e-13
amplitude: 1.3151374684412564e-12 += 1.44893999138725e-13
amplitude: 1.3129271261631717e-12 += 1.6952633299939272e-13
amplitude: 1.3471001029607577e-12 += 2.7036482562194493e-13
amplitude: 1.3927390374583621e-12 += 4.2599539418517427e-13
model: 5
amplitude: 1.284915019216048e-12 += 8.399892508051611e-14
amplitude: 1.2812145500869038e-12 += 9.472282099032611e-14
amplitude: 1.2853031988054163e-12 += 1.4784152019673303e-13
amplitude: 1.3102229254579215e-12 += 1.7825011526992521e-13
amplitude: 1.3425964936738894e-12 += 2.166506133662692e-13
amplitude: 1.3415548283233114e-12 += 2.50102049593457e-13
amplitude: 1.3951755496509142e-12 += 3.819799339836384e-13
amplitude: 1.4478668077370253e-12 += 6.266657013555647e-13
```

[27]:
```python
fig = plt.figure(figsize=[20,40],constrained_layout=True)

import matplotlib.gridspec as gridspec
```

```python
gs0 = gridspec.GridSpec(1, 5, figure=fig)

gs1 = gridspec.GridSpecFromSubplotSpec(8, 1, subplot_spec=gs0[0])
for n in range(8):
    ax = fig.add_subplot(gs1[n])
    plt.hist(amplitude[n][0], bins=10, alpha=0.5)
    plt.hist(amplitude[n][1], bins=10, alpha=0.5)
    plt.axvline(x=model_simu.parameters["amplitude"].value, color="green")
    plt.xlabel('amplitude\nl=0.01')
    plt.ylabel(f'Number of observations:{n_obs[n]}')

gs2 = gridspec.GridSpecFromSubplotSpec(8, 1, subplot_spec=gs0[1])
for n in range(8):
    ax = fig.add_subplot(gs2[n])
    plt.hist(amplitude[n][0], bins=10, alpha=0.5)
    plt.hist(amplitude[n][2], bins=10, alpha=0.5)
    plt.axvline(x=model_simu.parameters["amplitude"].value, color="green")
    plt.xlabel('amplitude\nl=0.1')
    plt.ylabel(f'Number of observations:{n_obs[n]}')

gs3 = gridspec.GridSpecFromSubplotSpec(8, 1, subplot_spec=gs0[2])
for n in range(8):
    ax = fig.add_subplot(gs3[n])
    plt.hist(amplitude[n][0], bins=10, alpha=0.5)
    plt.hist(amplitude[n][3], bins=10, alpha=0.5)
    plt.axvline(x=model_simu.parameters["amplitude"].value, color="green")
    plt.xlabel('amplitude\nl=0.2')
    plt.ylabel(f'Number of observations:{n_obs[n]}')

gs4 = gridspec.GridSpecFromSubplotSpec(8, 1, subplot_spec=gs0[3])
for n in range(8):
    ax = fig.add_subplot(gs4[n])
    plt.hist(amplitude[n][0], bins=10, alpha=0.5)
    plt.hist(amplitude[n][4], bins=10, alpha=0.5)
    plt.axvline(x=model_simu.parameters["amplitude"].value, color="green")
    plt.xlabel('amplitude\nl=0.3')
    plt.ylabel(f'Number of observations:{n_obs[n]}')

gs5 = gridspec.GridSpecFromSubplotSpec(8, 1, subplot_spec=gs0[4])
for n in range(8):
    ax = fig.add_subplot(gs5[n])
    plt.hist(amplitude[n][0], bins=10, alpha=0.5)
    plt.hist(amplitude[n][5], bins=10, alpha=0.5)
    plt.axvline(x=model_simu.parameters["amplitude"].value, color="green")
    plt.xlabel('amplitude\nl=0.5')
    plt.ylabel(f'Number of observations:{n_obs[n]}')
```

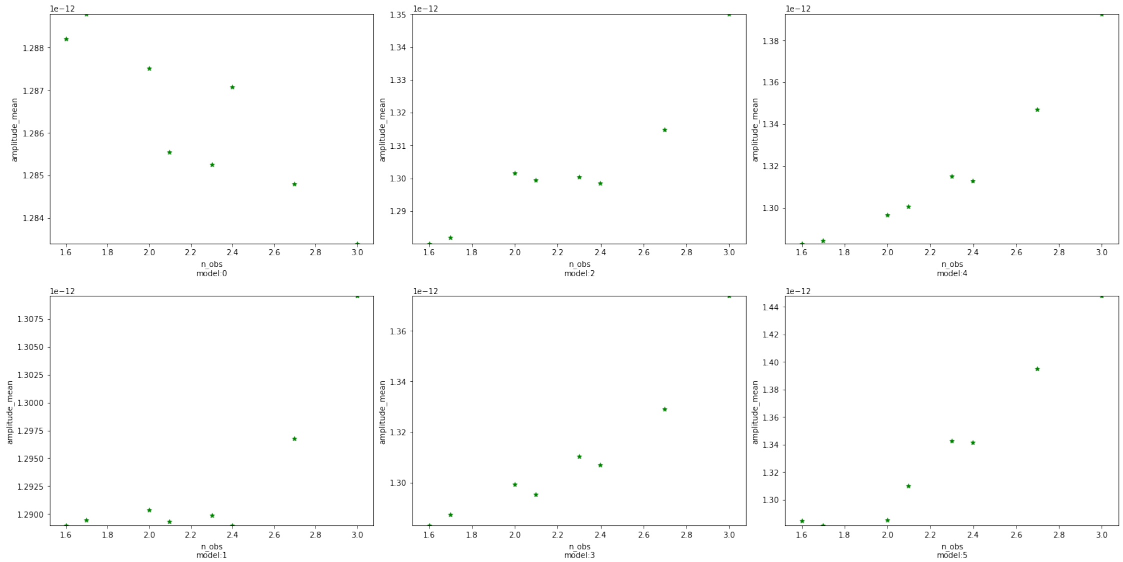
```
plt.show()
```



Number of observations 40
Number of observations 50
Number of observations 100
Number of observations 125
Number of observations 200
Number of observations 250
Number of observations 500
Number of observations 1000
amplitude
l=0.01
l=0.1
l=0.2
l=0.3
l=0.5
1e−12

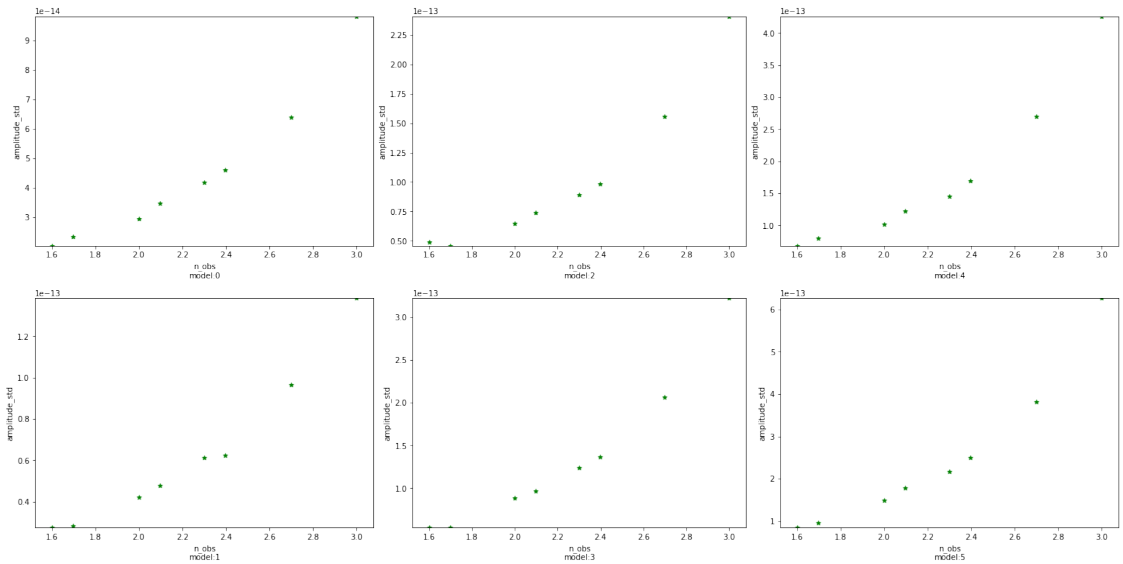
[28]:

```python
fig = plt.figure(figsize=[20,10],constrained_layout=True)

import matplotlib.gridspec as gridspec

gs0 = gridspec.GridSpec(1, 3, figure=fig)

gs1 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[0])
for n in range(2):
    ax = fig.add_subplot(gs1[n])
    plt.scatter(np.log10(n_obs), [row[n] for row in mu], label= "stars", color=␣
 ↪"green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n}')
    plt.ylabel('amplitude_mean')
    plt.ylim(min([row[n] for row in mu]),max([row[n] for row in mu]))

gs2 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[1])
for n in range(2):
    ax = fig.add_subplot(gs2[n])
    plt.scatter(np.log10(n_obs), [row[n+2] for row in mu], label= "stars",␣
 ↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+2}')
    plt.ylabel('amplitude_mean')
    plt.ylim(min([row[n+2] for row in mu]),max([row[n+2] for row in mu]))

gs3 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[2])
for n in range(2):
    ax = fig.add_subplot(gs3[n])
    plt.scatter(np.log10(n_obs), [row[n+4] for row in mu], label= "stars",␣
 ↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+4}')
    plt.ylabel('amplitude_mean')
    plt.ylim(min([row[n+4] for row in mu]),max([row[n+4] for row in mu]))

plt.show()
```



[29]:
```
fig = plt.figure(figsize=[20,10],constrained_layout=True)

import matplotlib.gridspec as gridspec

gs0 = gridspec.GridSpec(1, 3, figure=fig)

gs1 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[0])
for n in range(2):
    ax = fig.add_subplot(gs1[n])
    plt.scatter(np.log10(n_obs), [row[n] for row in sigma], label= "stars",␣
 ↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n}')
    plt.ylabel('amplitude_std')
    plt.ylim(min([row[n] for row in sigma]),max([row[n] for row in sigma]))

gs2 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[1])
for n in range(2):
    ax = fig.add_subplot(gs2[n])
    plt.scatter(np.log10(n_obs), [row[n+2] for row in sigma], label= "stars",␣
 ↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+2}')
    plt.ylabel('amplitude_std')
    plt.ylim(min([row[n+2] for row in sigma]),max([row[n+2] for row in sigma]))

gs3 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[2])
for n in range(2):
```

```
    ax = fig.add_subplot(gs3[n])
    plt.scatter(np.log10(n_obs), [row[n+4] for row in sigma], label= "stars",␣
↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+4}')
    plt.ylabel('amplitude_std')
    plt.ylim(min([row[n+4] for row in sigma]),max([row[n+4] for row in sigma]))

plt.show()
```



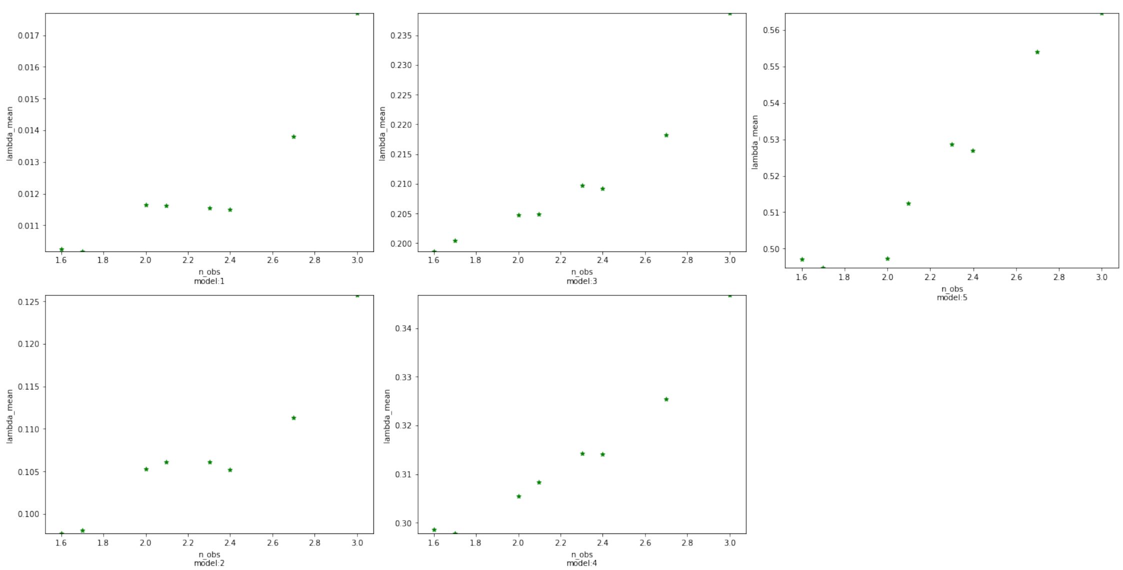
[30]:
```
for j in range(5):
    print(f"model: {j}")
    for i in range(8):
        mu[i][j]=lambda_[i][j].mean()
        sigma[i][j]=lambda_[i][j].std()
        print(f"lambda_: {lambda_[i][j].mean()} += {lambda_[i][j].std()}")
```

```
model: 0
lambda_: 0.010258673762881799 += 0.003816841078315301
lambda_: 0.010184648621490816 += 0.003938467864632929
lambda_: 0.011639173153405624 += 0.006698493085092638
lambda_: 0.011613996276705387 += 0.00709438332512854
lambda_: 0.011542533059089654 += 0.009815750440930034
lambda_: 0.011494085807908082 += 0.010726172710231565
lambda_: 0.013796467677061698 += 0.017979523189227093
lambda_: 0.017706978731673047 += 0.026640397643808697
model: 1
lambda_: 0.09767607967793786 += 0.01422881257575096
lambda_: 0.09810229530400459 += 0.013334035654514477
```

```
lambda_: 0.10530386799605303 += 0.0200096434828582
lambda_: 0.10614898913685648 += 0.023933350206845404
lambda_: 0.10612298373787767 += 0.027953387701715772
lambda_: 0.10523471201628334 += 0.031548960858549996
lambda_: 0.11133697949662273 += 0.05118767665273025
lambda_: 0.12571580037646582 += 0.07684586566480776
model: 2
lambda_: 0.198664559612202 += 0.022102767761110198
lambda_: 0.20048388924667912 += 0.023235245519206204
lambda_: 0.20480604169635522 += 0.03665799752360788
lambda_: 0.2049514489282098 += 0.039688459940184176
lambda_: 0.2097152293468037 += 0.04973796038628897
lambda_: 0.20923861643387698 += 0.054738991848648826
lambda_: 0.21816951942121346 += 0.08269890720724825
lambda_: 0.23877889764436394 += 0.12530023185713818
model: 3
lambda_: 0.2986029959571933 += 0.03362319511056934
lambda_: 0.29776875888745835 += 0.03657491320811148
lambda_: 0.30550168844294684 += 0.04859988845630001
lambda_: 0.30832070918112237 += 0.05972855771662421
lambda_: 0.3143134408824053 += 0.06668801664102199
lambda_: 0.31410960776976843 += 0.07923147595501738
lambda_: 0.32538261443866806 += 0.12291353433118686
lambda_: 0.3467775680078413 += 0.17821980257402809
model: 4
lambda_: 0.4970171794676251 += 0.049303822838598924
lambda_: 0.4946810923652535 += 0.05393564459873155
lambda_: 0.49728011612120143 += 0.08545717570847972
lambda_: 0.5124402520665081 += 0.10650266095882245
lambda_: 0.5285900023107939 += 0.12512264393374997
lambda_: 0.5268403993389769 += 0.14759931081631383
lambda_: 0.5539721622880759 += 0.20349039915270206
lambda_: 0.5646443353604355 += 0.29789855038523
```

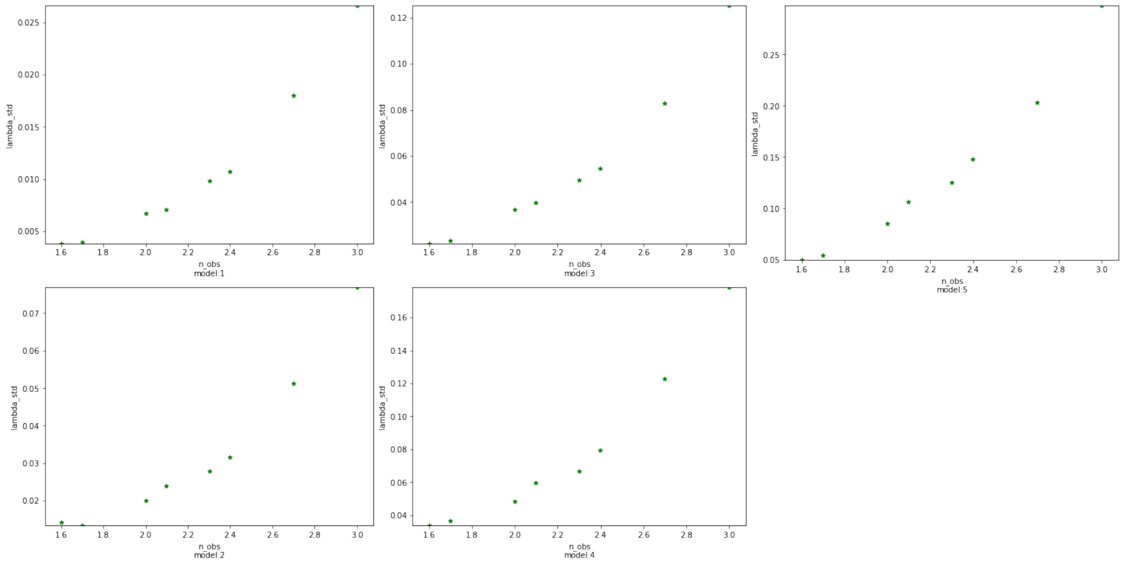
[31]:
```
fig = plt.figure(figsize=[20,10],constrained_layout=True)

import matplotlib.gridspec as gridspec

gs0 = gridspec.GridSpec(1, 3, figure=fig)

gs1 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[0])
for n in range(2):
    ax = fig.add_subplot(gs1[n])
    plt.scatter(np.log10(n_obs), [row[n] for row in mu], label= "stars", color=␣
 ↪"green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+1}')
```

```
    plt.ylabel('lambda_mean')
    plt.ylim(min([row[n] for row in mu]),max([row[n] for row in mu]))

gs2 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[1])
for n in range(2):
    ax = fig.add_subplot(gs2[n])
    plt.scatter(np.log10(n_obs), [row[n+2] for row in mu], label= "stars",␣
 ↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+3}')
    plt.ylabel('lambda_mean')
    plt.ylim(min([row[n+2] for row in mu]),max([row[n+2] for row in mu]))

gs3 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[2])
for n in range(1):
    ax = fig.add_subplot(gs3[n])
    plt.scatter(np.log10(n_obs), [row[n+4] for row in mu], label= "stars",␣
 ↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+5}')
    plt.ylabel('lambda_mean')
    plt.ylim(min([row[n+4] for row in mu]),max([row[n+4] for row in mu]))

plt.show()
```



[32]:
```
fig = plt.figure(figsize=[20,10],constrained_layout=True)

import matplotlib.gridspec as gridspec
```

```
gs0 = gridspec.GridSpec(1, 3, figure=fig)

gs1 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[0])
for n in range(2):
    ax = fig.add_subplot(gs1[n])
    plt.scatter(np.log10(n_obs), [row[n] for row in sigma], label= "stars",␣
 ↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+1}')
    plt.ylabel('lambda_std')
    plt.ylim(min([row[n] for row in sigma]),max([row[n] for row in sigma]))

gs2 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[1])
for n in range(2):
    ax = fig.add_subplot(gs2[n])
    plt.scatter(np.log10(n_obs), [row[n+2] for row in sigma], label= "stars",␣
 ↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+3}')
    plt.ylabel('lambda_std')
    plt.ylim(min([row[n+2] for row in sigma]),max([row[n+2] for row in sigma]))

gs3 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[2])
for n in range(1):
    ax = fig.add_subplot(gs3[n])
    plt.scatter(np.log10(n_obs), [row[n+4] for row in sigma], label= "stars",␣
 ↪color= "green",
                marker= "*", s=30)
    plt.xlabel(f'n_obs\nmodel:{n+5}')
    plt.ylabel('lambda_std')
    plt.ylim(min([row[n+4] for row in sigma]),max([row[n+4] for row in sigma]))

plt.show()
```



[33]:

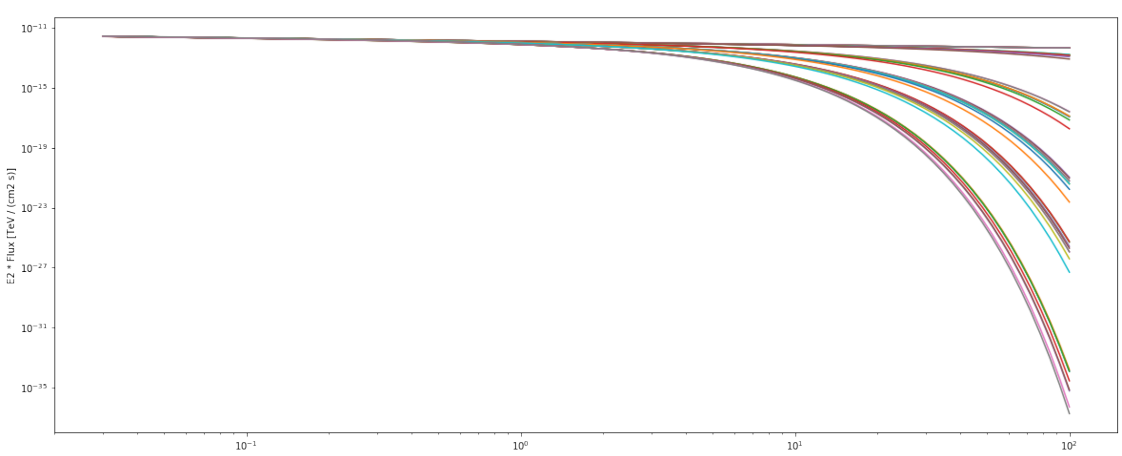
```
sim = [[0 for i in range(cols)] for j in range(rows)]
for i in range(8):
    s = PowerLawSpectralModel(
        index=index[i][0].mean(),
        amplitude=amplitude[i][0].mean() * u.Unit("cm-2 s-1 TeV-1"),
        reference=1 * u.TeV,
    )
    print(s)
    sim[i][0]=s

for j in range(5):
    for i in range(8):
        s = ExpCutoffPowerLawSpectralModel(
            index=index[i][j+1].mean(),
            amplitude=amplitude[i][j+1].mean() * u.Unit("cm-2 s-1 TeV-1"),
            reference=1 * u.TeV,
            lambda_=lambda_[i][j].mean() * u.Unit("TeV-1"),
            alpha = 1,
        )
        print(s)
        sim[i][j+1]=s
```

```
PowerLawSpectralModel

   name     value    error      unit      min max frozen
--------- --------- ----- -------------- --- --- ------
    index 2.221e+00   nan                nan nan  False
amplitude 1.288e-12   nan cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00   nan            TeV nan nan   True
```

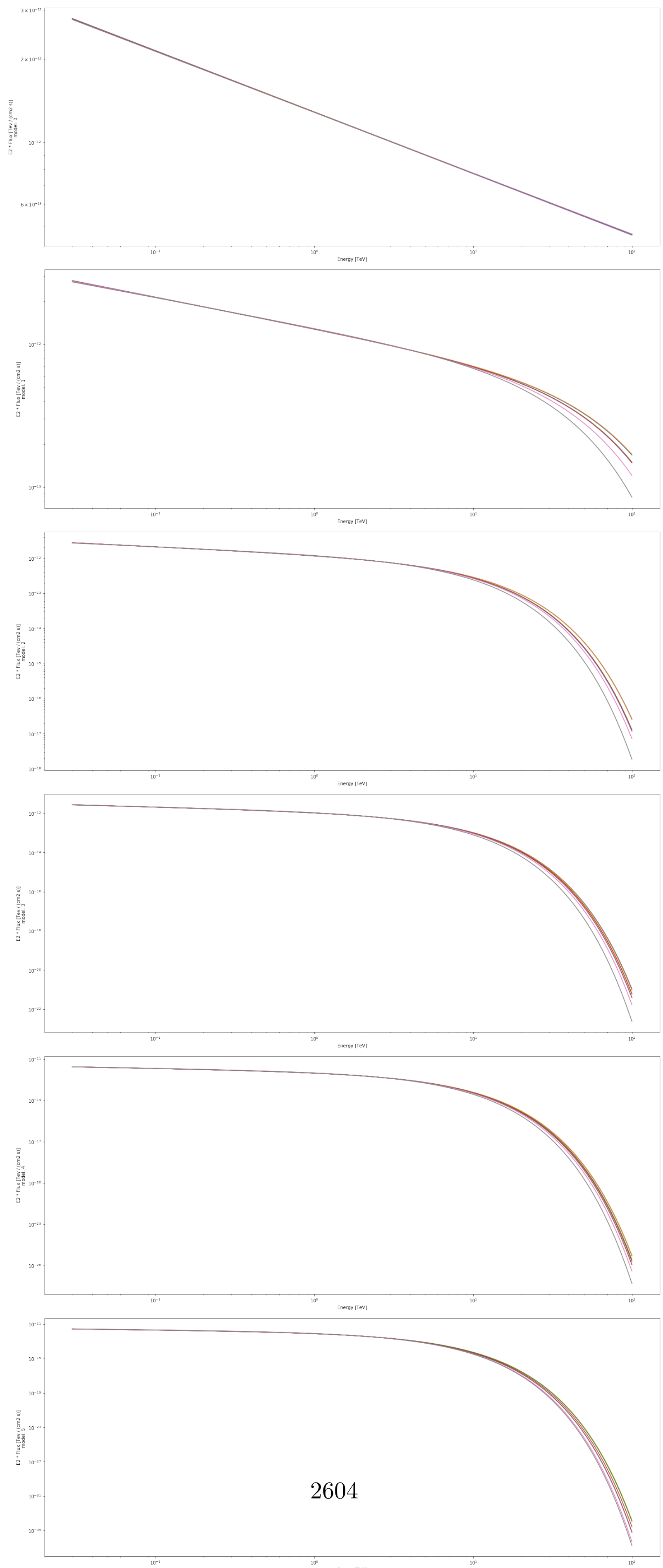
```
   name     value   error      unit      min max frozen
--------- --------- ----- -------------- --- --- ------
    index 2.209e+00   nan                nan nan  False
amplitude 1.342e-12   nan cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00   nan            TeV nan nan   True
  lambda_ 5.268e-01   nan          TeV-1 nan nan  False
    alpha 1.000e+00   nan                nan nan   True
ExpCutoffPowerLawSpectralModel

   name     value   error      unit      min max frozen
--------- --------- ----- -------------- --- --- ------
    index 2.200e+00   nan                nan nan  False
amplitude 1.395e-12   nan cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00   nan            TeV nan nan   True
  lambda_ 5.540e-01   nan          TeV-1 nan nan  False
    alpha 1.000e+00   nan                nan nan   True
ExpCutoffPowerLawSpectralModel

   name     value   error      unit      min max frozen
--------- --------- ----- -------------- --- --- ------
    index 2.198e+00   nan                nan nan  False
amplitude 1.448e-12   nan cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00   nan            TeV nan nan   True
  lambda_ 5.646e-01   nan          TeV-1 nan nan  False
    alpha 1.000e+00   nan                nan nan   True
```

[34]:
```
plt.figure(figsize=[20,8])
energy_range = [0.03, 100] * u.TeV
for j in range(6):
    for i in range(8):
        sim[i][j].plot(energy_range=energy_range, energy_power=2)
plt.show
```

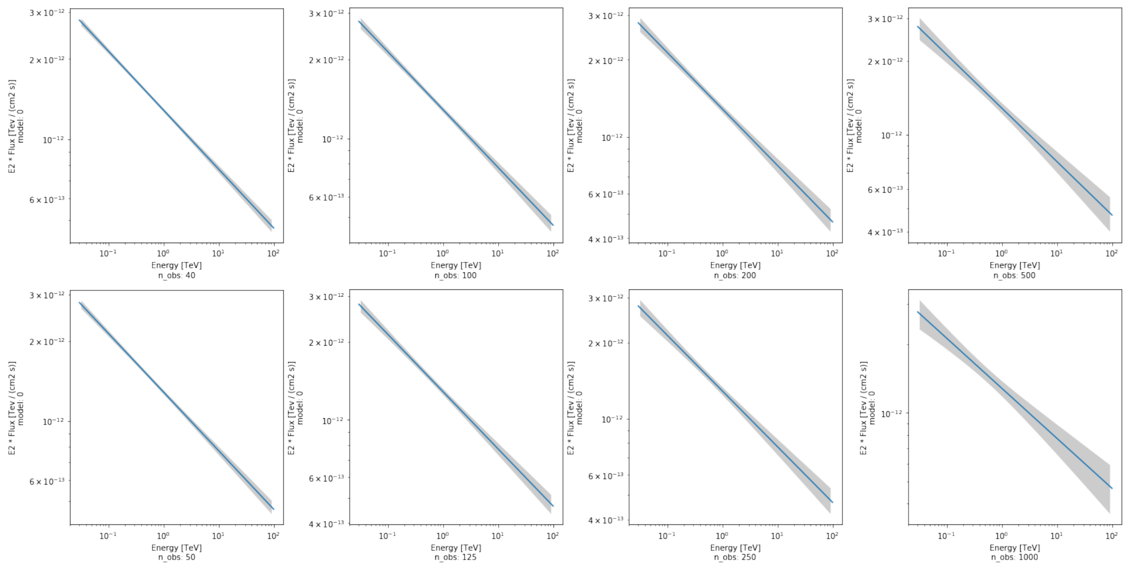
[34]:
```
<function matplotlib.pyplot.show(*args, **kw)>
```



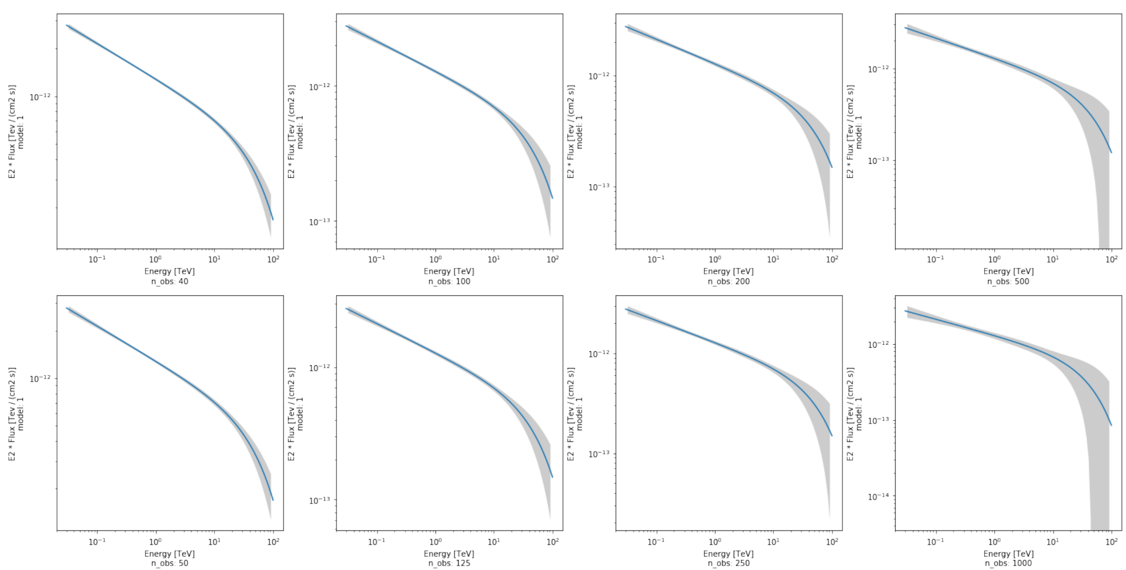
[35]:
```
fig = plt.figure(figsize=[20,48],constrained_layout=True)

import matplotlib.gridspec as gridspec

gs0 = gridspec.GridSpec(1, 1, figure=fig)

gs1 = gridspec.GridSpecFromSubplotSpec(6, 1, subplot_spec=gs0[0])
for n in range(6):
    ax = fig.add_subplot(gs1[n])
    for i in range(8):
        sim[i][n].plot(energy_range=energy_range, energy_power=2)
    plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {n}')

plt.show()
```



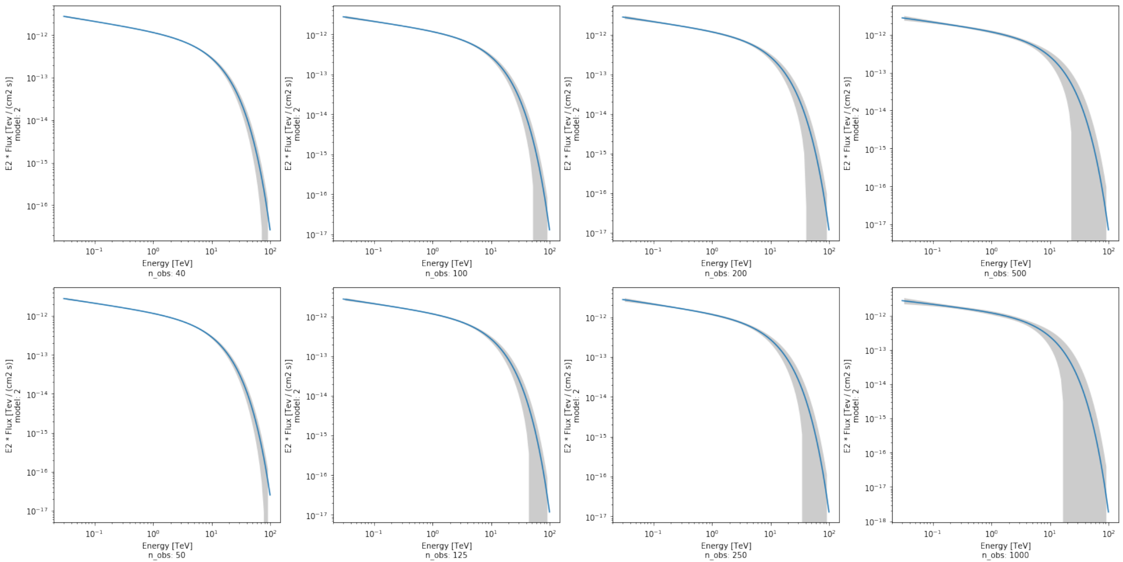
[36]:
```
for i in range(6):
    fig = plt.figure(figsize=[20,10],constrained_layout=True)

    import matplotlib.gridspec as gridspec

    gs0 = gridspec.GridSpec(1, 4, figure=fig)

    gs1 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[0])
    for n in range(2):
        ax = fig.add_subplot(gs1[n])
        sim[n][i].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[n][i], covar=covar[n][i],energy_range=energy_range,␣
↪energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[n]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {i}')

    gs2 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[1])
    for n in range(2):
        ax = fig.add_subplot(gs2[n])
        sim[n+2][i].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[n+2][i],␣
↪covar=covar[n+2][i],energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[n+2]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {i}')

    gs3 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[2])
    for n in range(2):
        ax = fig.add_subplot(gs3[n])
        sim[n+4][i].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[n+4][i],␣
↪covar=covar[n+4][i],energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[n+4]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {i}')

    gs4 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[3])
    for n in range(2):
        ax = fig.add_subplot(gs4[n])
        sim[n+6][i].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[n+6][i],␣
↪covar=covar[n+6][i],energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[n+6]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {i}')

    plt.show()
```

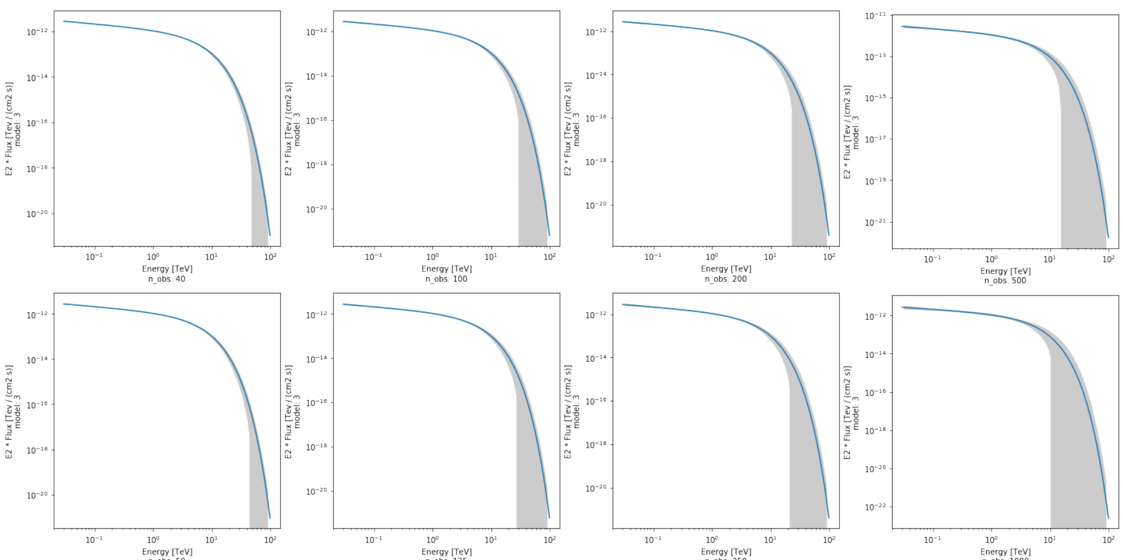


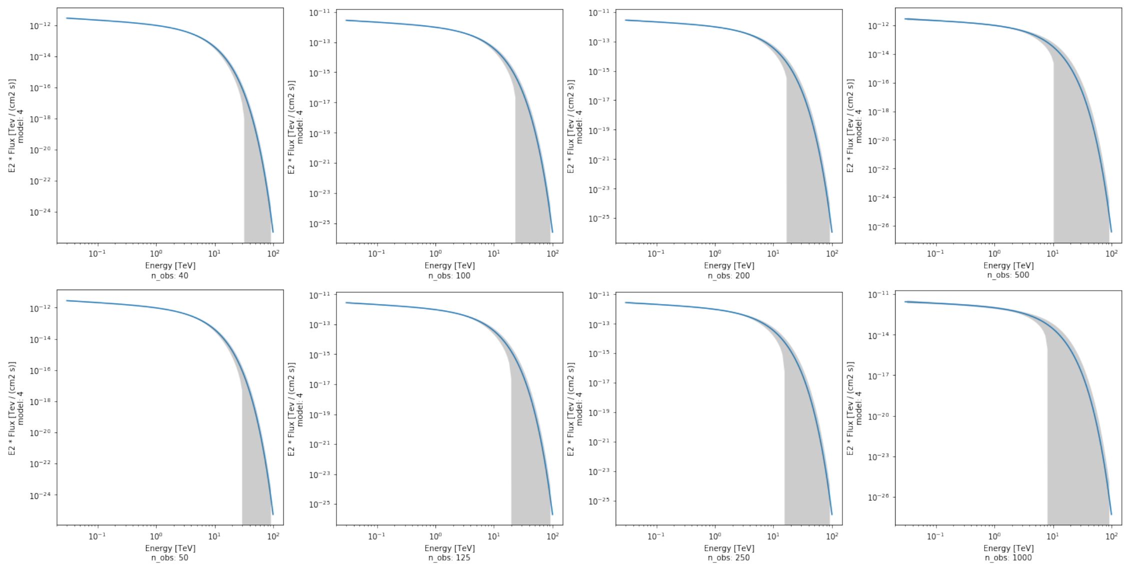
E2 * Flux [Tev / (cm2 s)]
model: 0
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000

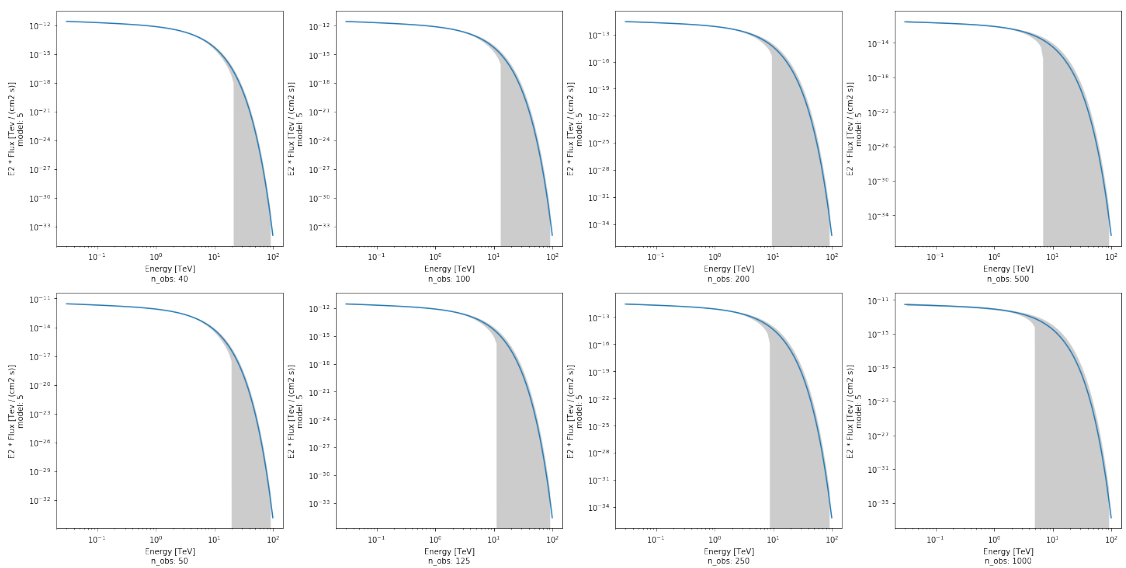


E2 * Flux [Tev / (cm2 s)]
model: 1
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000


E2 * Flux [Tev / (cm2 s)]
model: 2
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000

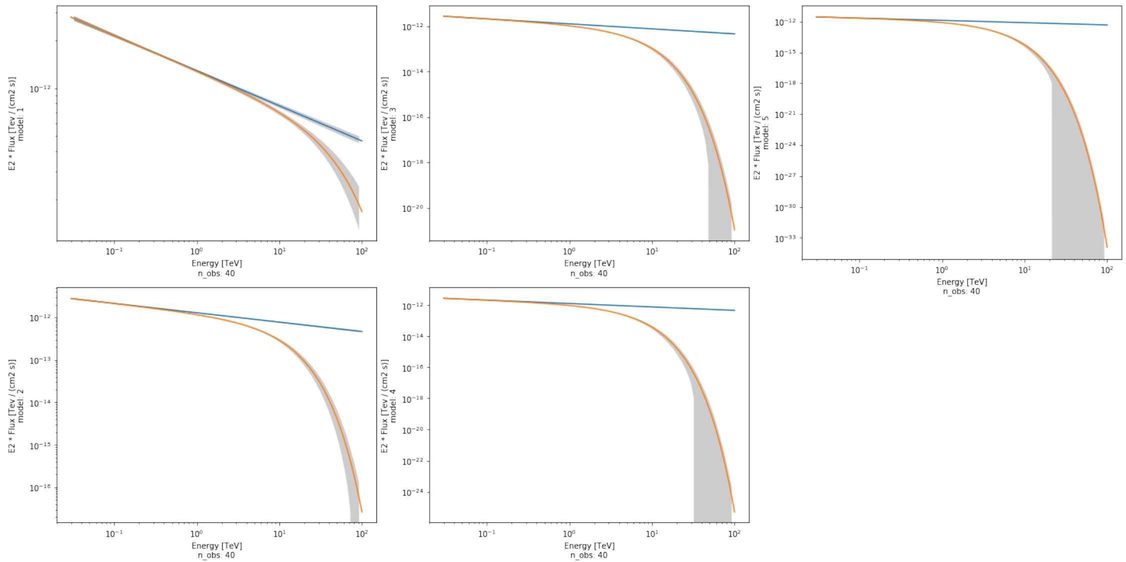

E2 * Flux [Tev / (cm2 s)]
model: 3
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000



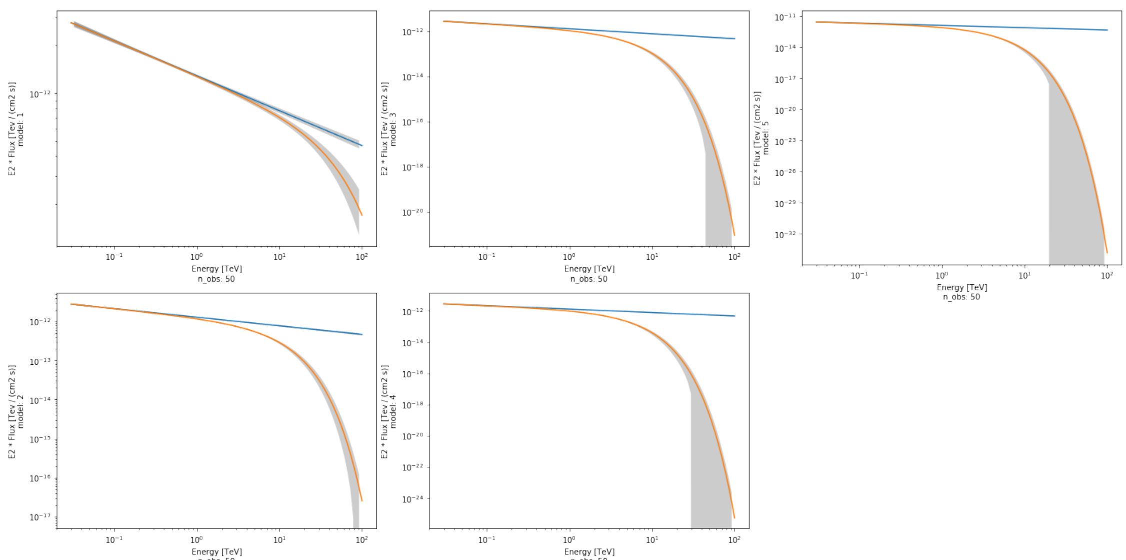


[37]:
```
for i in range(8):
    fig = plt.figure(figsize=[20,10],constrained_layout=True)

    import matplotlib.gridspec as gridspec

    gs0 = gridspec.GridSpec(1, 3, figure=fig)

    gs1 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[0])
    for n in range(2):
        ax = fig.add_subplot(gs1[n])
```

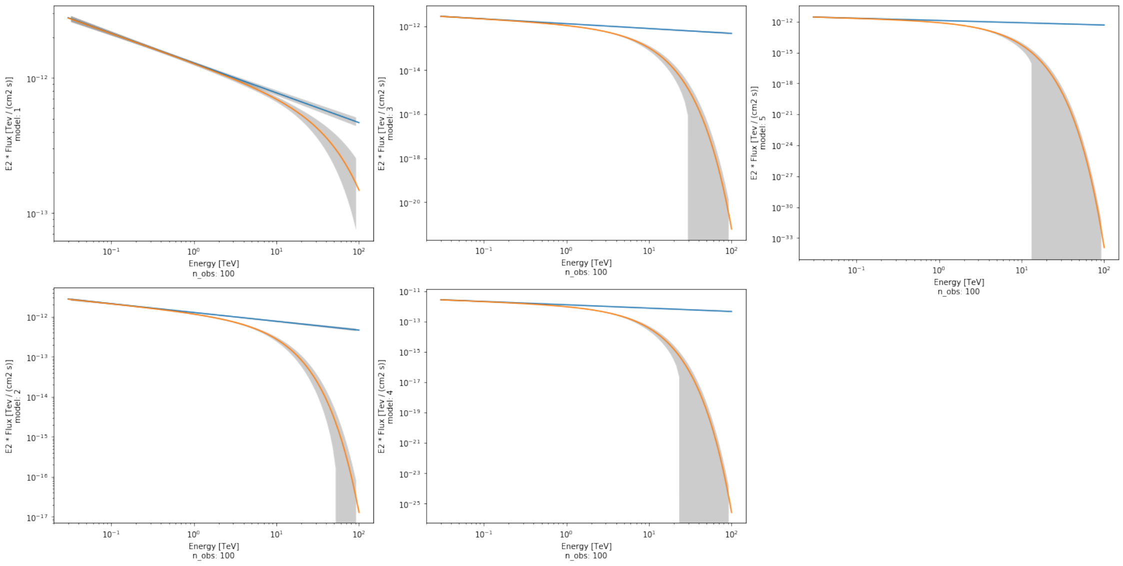
```
        sim[i][0].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[i][0], covar=covar[i][0],energy_range=energy_range,␣
↪energy_power=2)
        sim[i][n+1].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[i][n+1],␣
↪covar=covar[i][n+1],energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[i]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {n+1}')

    gs2 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[1])
    for n in range(2):
        ax = fig.add_subplot(gs2[n])
        sim[i][0].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[i][0], covar=covar[i][0],energy_range=energy_range,␣
↪energy_power=2)
        sim[i][n+3].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[i][n+3],␣
↪covar=covar[i][n+3],energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[i]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {n+3}')

    gs3 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[2])
    for n in range(1):
        ax = fig.add_subplot(gs3[n])
        sim[i][0].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[i][0], covar=covar[i][0],energy_range=energy_range,␣
↪energy_power=2)
        sim[i][n+5].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[i][n+5],␣
↪covar=covar[i][n+5],energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[i]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {n+5}')

    plt.show()
```

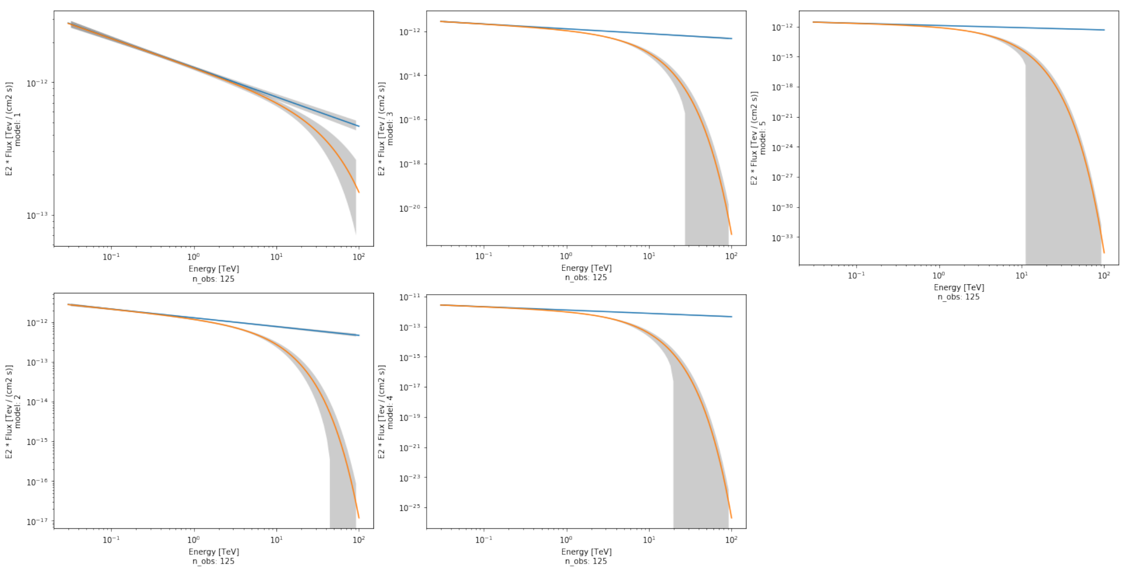


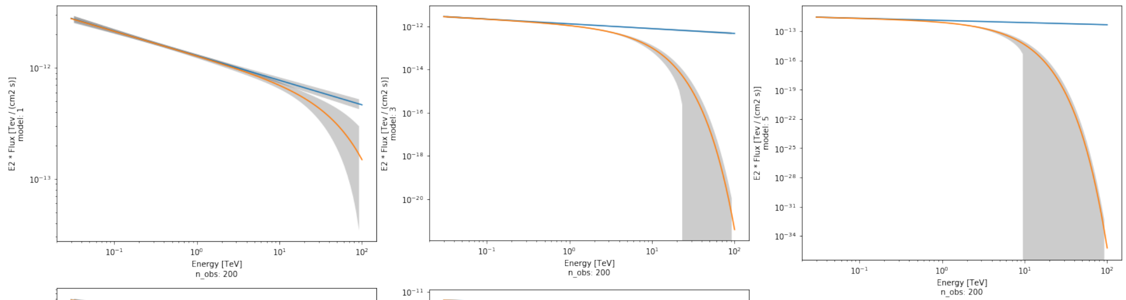
E2 * Flux [Tev / (cm2 s)]
model: 1
model: 2
model: 3
model: 4
model: 5
Energy [TeV]
n_obs: 40

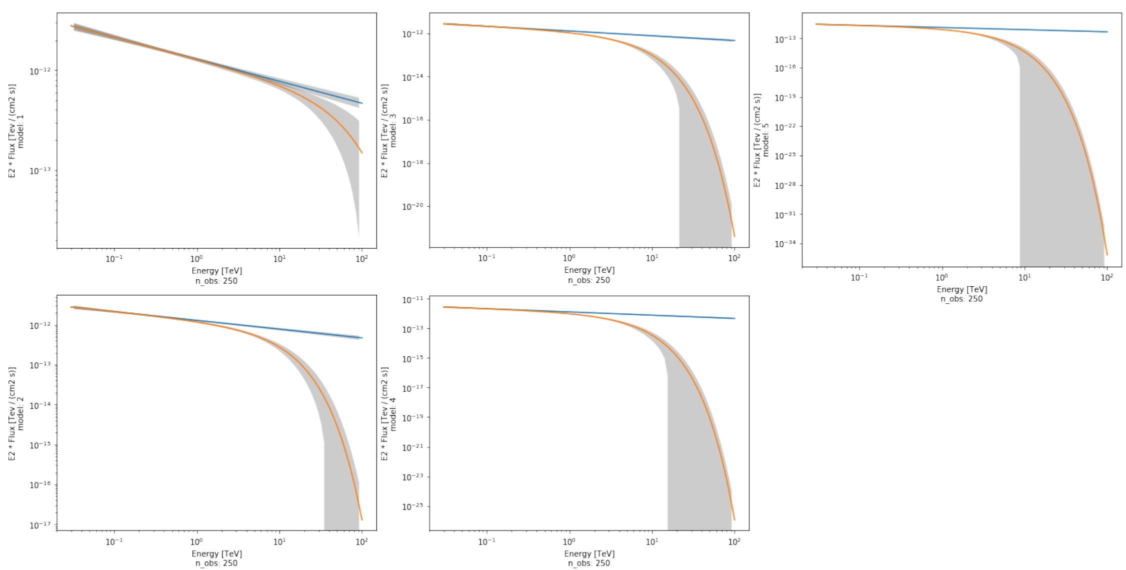


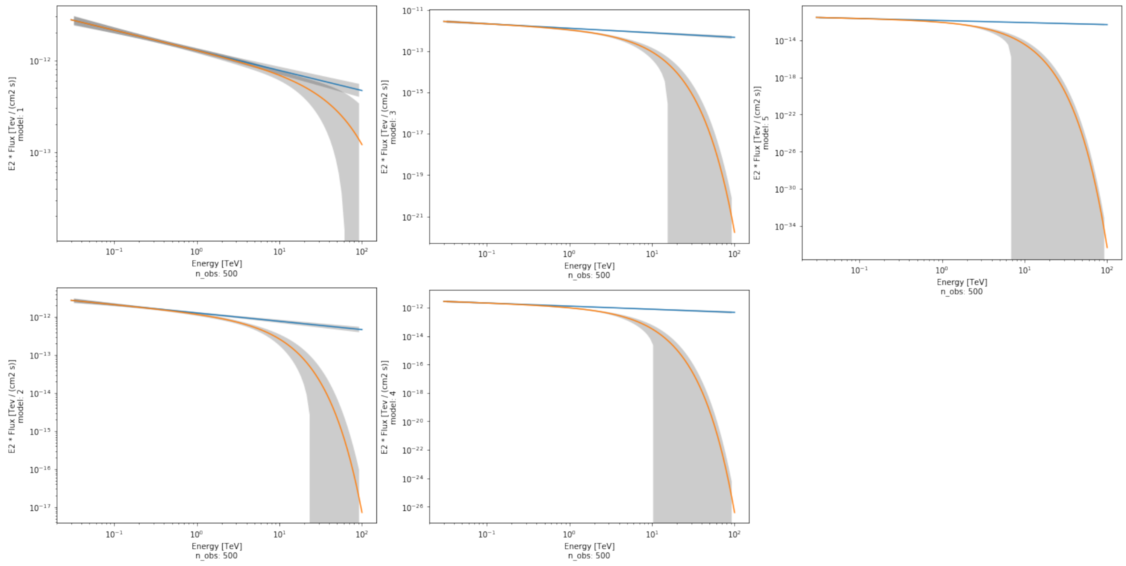
E2 * Flux [Tev / (cm2 s)]
model: 1
model: 2
model: 3
model: 4
model: 5
Energy [TeV]
n_obs: 50

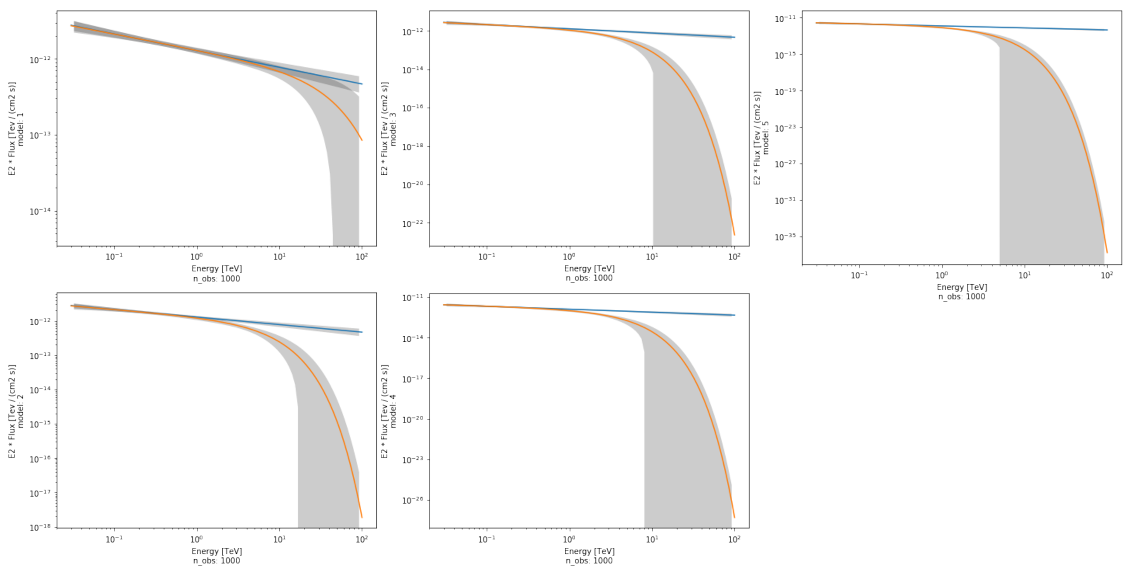

E2 * Flux [Tev / (cm2 s)]
model: 1
Energy [TeV]
n_obs: 100
E2 * Flux [Tev / (cm2 s)]
model: 3
Energy [TeV]
n_obs: 100
E2 * Flux [Tev / (cm2 s)]
model: 5
Energy [TeV]
n_obs: 100
E2 * Flux [Tev / (cm2 s)]
model: 2
Energy [TeV]
n_obs: 100
E2 * Flux [Tev / (cm2 s)]
model: 4
Energy [TeV]
n_obs: 100


E2 * Flux [Tev / (cm2 s)]
model: 1
Energy [TeV]
n_obs: 125
E2 * Flux [Tev / (cm2 s)]
model: 3
Energy [TeV]
n_obs: 125
E2 * Flux [Tev / (cm2 s)]
model: 5
Energy [TeV]
n_obs: 125
E2 * Flux [Tev / (cm2 s)]
model: 2
Energy [TeV]
n_obs: 125
E2 * Flux [Tev / (cm2 s)]
model: 4
Energy [TeV]
n_obs: 125


E2 * Flux [Tev / (cm2 s)]
model: 1
Energy [TeV]
n_obs: 200
E2 * Flux [Tev / (cm2 s)]
model: 3
Energy [TeV]
n_obs: 200
E2 * Flux [Tev / (cm2 s)]
model: 5
Energy [TeV]
n_obs: 200
E2 * Flux [Tev / (cm2 s)]
model: 2
Energy [TeV]
n_obs: 200
E2 * Flux [Tev / (cm2 s)]
model: 4
Energy [TeV]
n_obs: 200

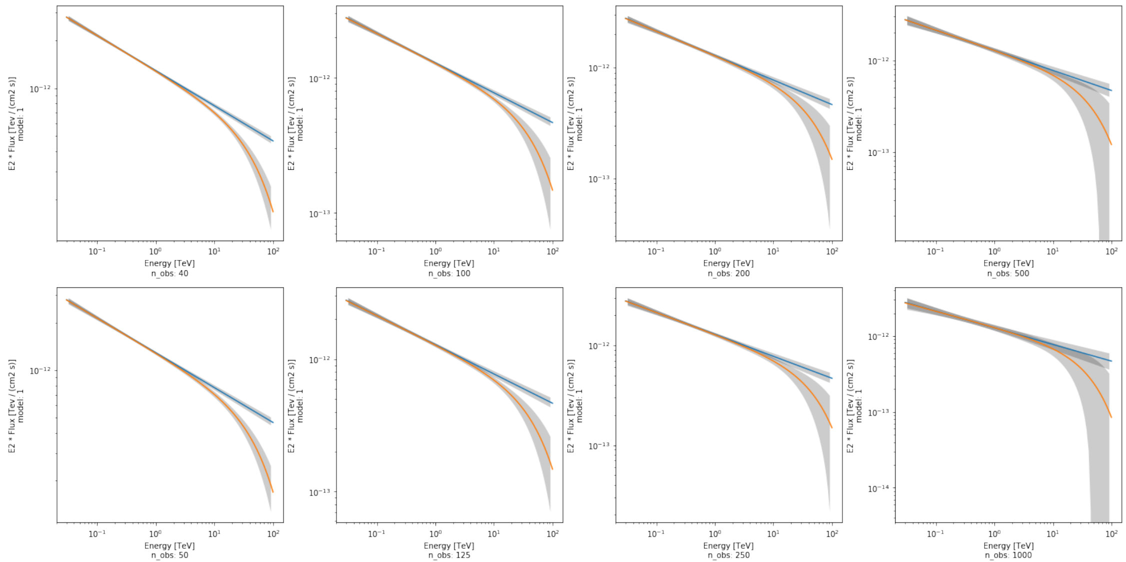

E2 * Flux [Tev / (cm2 s)]
model: 1
Energy [TeV]
n_obs: 250
E2 * Flux [Tev / (cm2 s)]
model: 3
Energy [TeV]
n_obs: 250
E2 * Flux [Tev / (cm2 s)]
model: 5
Energy [TeV]
n_obs: 250
E2 * Flux [Tev / (cm2 s)]
model: 2
Energy [TeV]
n_obs: 250
E2 * Flux [Tev / (cm2 s)]
model: 4
Energy [TeV]
n_obs: 250

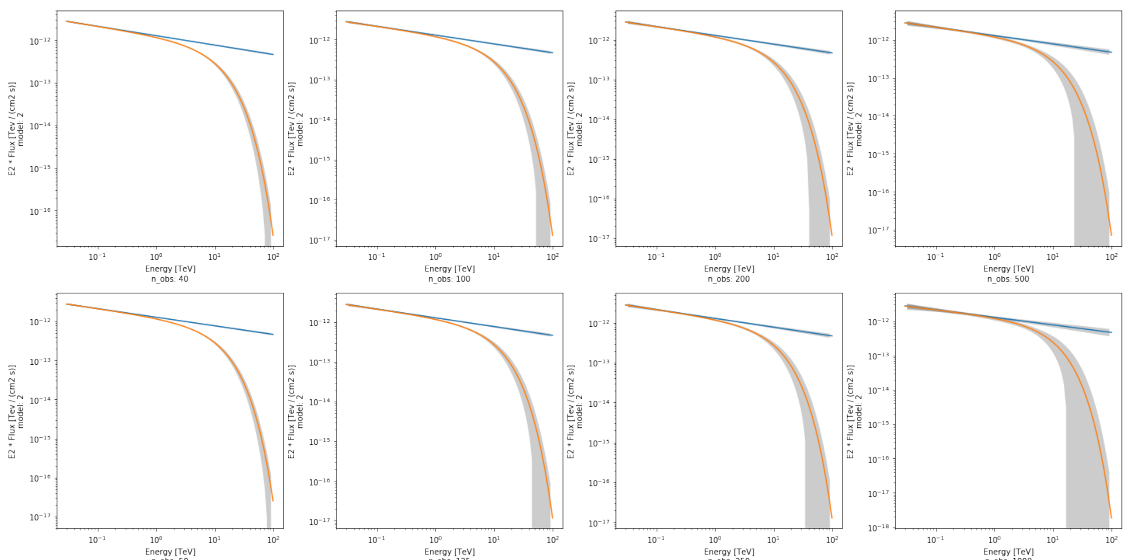


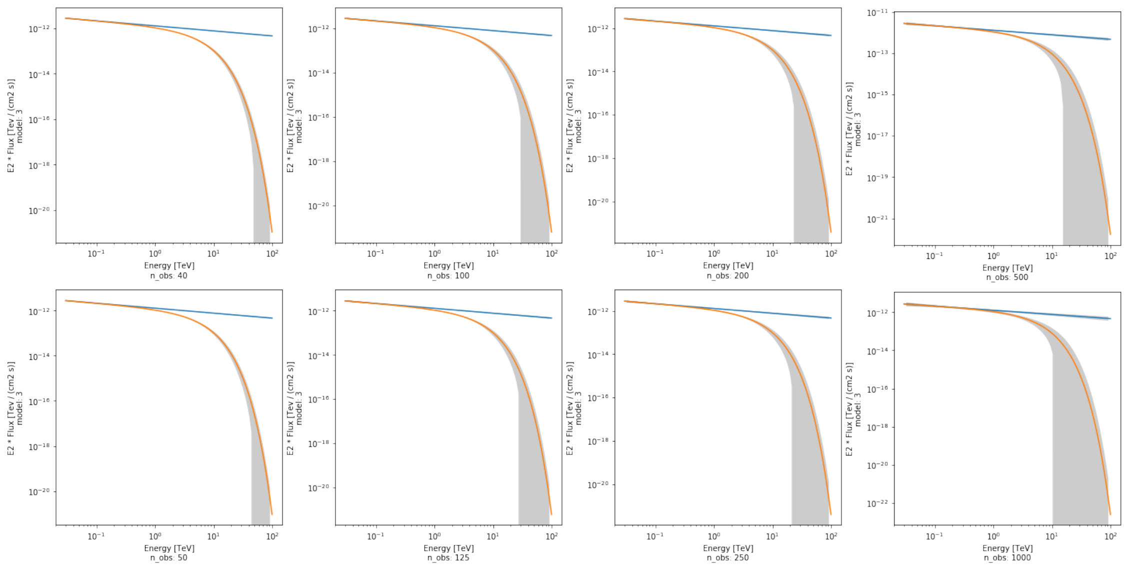


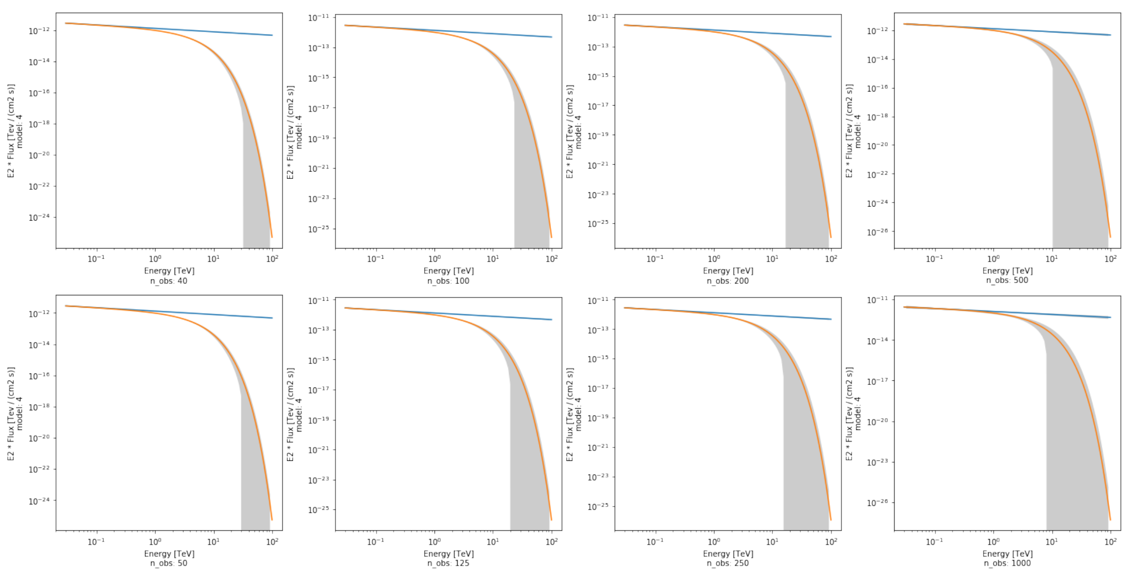
[38]:
```
for i in range(5):
    fig = plt.figure(figsize=[20,10],constrained_layout=True)

    import matplotlib.gridspec as gridspec

    gs0 = gridspec.GridSpec(1, 4, figure=fig)

    gs1 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[0])
    for n in range(2):
        ax = fig.add_subplot(gs1[n])
```

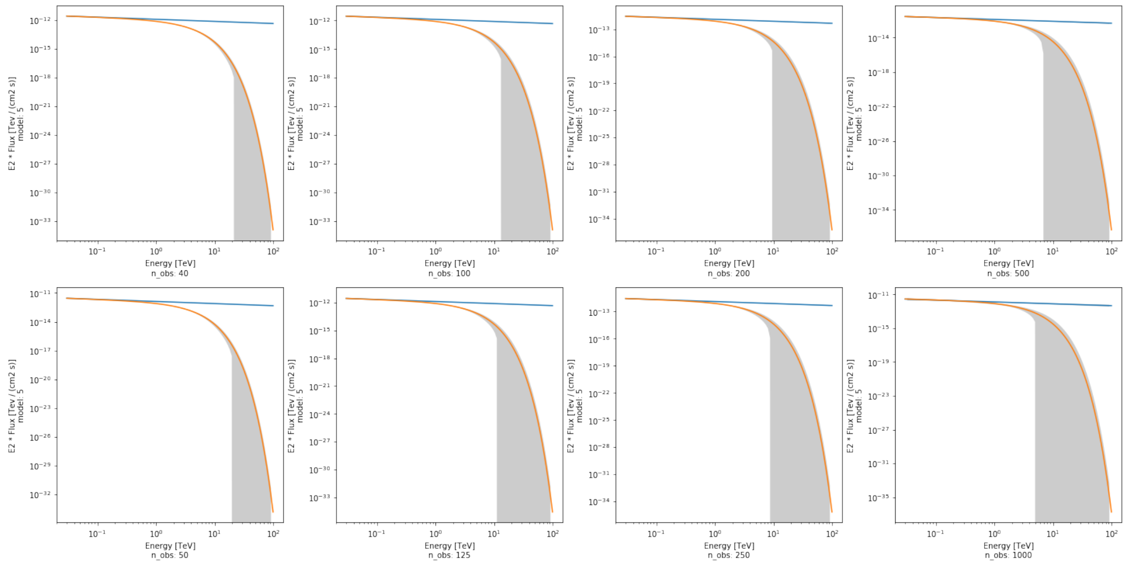
```
        sim[n][0].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[n][0], covar=covar[n][0],energy_range=energy_range,␣
↪energy_power=2)
        sim[n][i+1].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[n][i+1],␣
↪covar=covar[n][i+1],energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[n]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {i+1}')

    gs2 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[1])
    for n in range(2):
        ax = fig.add_subplot(gs2[n])
        sim[n+2][0].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[n+2][0],␣
↪covar=covar[n+2][0],energy_range=energy_range, energy_power=2)
        sim[n+2][i+1].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[n+2][i+1],␣
↪covar=covar[n+2][i+1],energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[n+2]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {i+1}')

    gs3 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[2])
    for n in range(2):
        ax = fig.add_subplot(gs3[n])
        sim[n+4][0].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[n+4][0],␣
↪covar=covar[n+4][0],energy_range=energy_range, energy_power=2)
        sim[n+4][i+1].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[n+4][i+1],␣
↪covar=covar[n+4][i+1],energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[n+4]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {i+1}')

    gs4 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[3])
    for n in range(2):
        ax = fig.add_subplot(gs4[n])
        sim[n+6][0].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[n+6][0],␣
↪covar=covar[n+6][0],energy_range=energy_range, energy_power=2)
        sim[n+6][i+1].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim[n+6][i+1],␣
↪covar=covar[n+6][i+1],energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[n+6]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {i+1}')

    plt.show()
```


E2 * Flux [Tev / (cm2 s)]
model: 1
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000


E2 * Flux [Tev / (cm2 s)]
model: 2
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000


E2 * Flux [Tev / (cm2 s)]
model: 3
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000


E2 * Flux [Tev / (cm2 s)]
model: 4
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000



[39]:

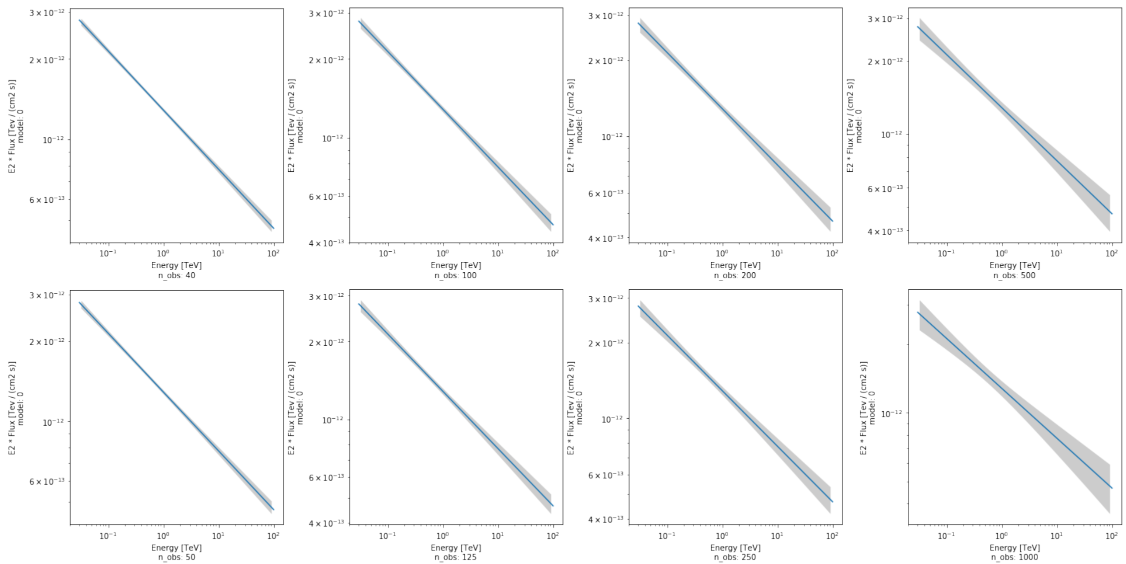
```
%%time
res_1 = [[0 for i in range(cols)] for j in range(rows)]
minuit_opts = {"tol": 0.001, "strategy": 1}
for i in range(8):
    results_1 = []
    for dataset in datas[i][0]:
        dataset.models = model[i][0].copy()
        fit_1 = Fit([dataset])
        result_1 = fit_1.run(optimize_opts=minuit_opts)
        results_1.append(
            {
                "index": result_1.parameters["index"].value,
                "amplitude": result_1.parameters["amplitude"].value,
                "reference":result_1.parameters["reference"].value,
                "covariance": result_1.parameters.covariance,
            }
        )
        print(result_1.parameters.to_table())
    res_1[i][0]=results_1
```

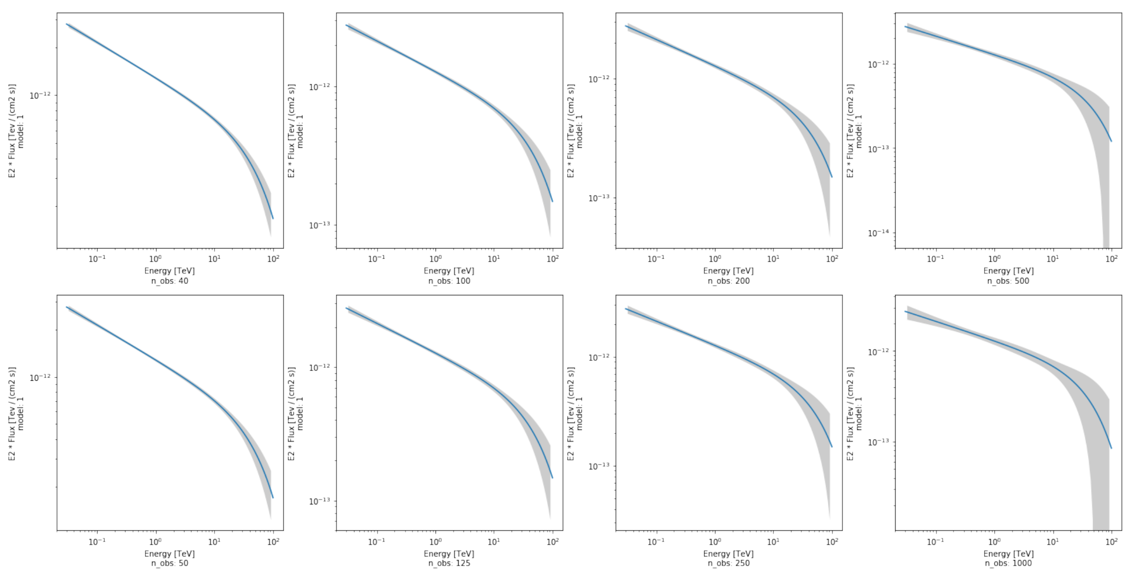
```
   name     value     error         unit       min max frozen
--------- --------- --------- -------------- --- --- ------
    index 2.210e+00 9.028e-03                nan nan  False
amplitude 1.318e-12 1.964e-14 cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00 0.000e+00            TeV nan nan   True
   name     value     error         unit       min max frozen
--------- --------- --------- -------------- --- --- ------
    index 2.222e+00 9.318e-03                nan nan  False
amplitude 1.258e-12 1.930e-14 cm-2 s-1 TeV-1 nan nan  False
```

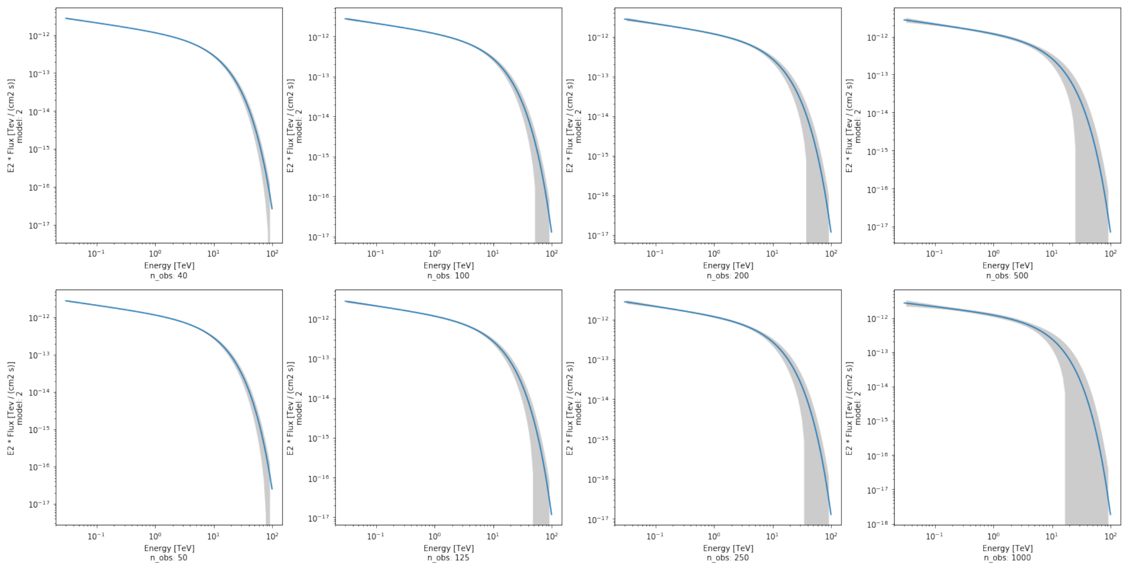
```
    alpha 1.000e+00   nan                nan nan   True
ExpCutoffPowerLawSpectralModel

   name     value   error      unit      min max frozen
--------- --------- ----- -------------- --- --- ------
    index 2.207e+00   nan                nan nan  False
amplitude 1.343e-12   nan cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00   nan            TeV nan nan   True
  lambda_ 5.286e-01   nan          TeV-1 nan nan  False
    alpha 1.000e+00   nan                nan nan   True
ExpCutoffPowerLawSpectralModel

   name     value   error      unit      min max frozen
--------- --------- ----- -------------- --- --- ------
    index 2.209e+00   nan                nan nan  False
amplitude 1.342e-12   nan cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00   nan            TeV nan nan   True
  lambda_ 5.268e-01   nan          TeV-1 nan nan  False
    alpha 1.000e+00   nan                nan nan   True
ExpCutoffPowerLawSpectralModel

   name     value   error      unit      min max frozen
--------- --------- ----- -------------- --- --- ------
    index 2.200e+00   nan                nan nan  False
amplitude 1.395e-12   nan cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00   nan            TeV nan nan   True
  lambda_ 5.540e-01   nan          TeV-1 nan nan  False
    alpha 1.000e+00   nan                nan nan   True
ExpCutoffPowerLawSpectralModel

   name     value   error      unit      min max frozen
--------- --------- ----- -------------- --- --- ------
    index 2.198e+00   nan                nan nan  False
amplitude 1.448e-12   nan cm-2 s-1 TeV-1 nan nan  False
reference 1.000e+00   nan            TeV nan nan   True
  lambda_ 5.646e-01   nan          TeV-1 nan nan  False
    alpha 1.000e+00   nan                nan nan   True
```

[43]:
```
for i in range(6):
    fig = plt.figure(figsize=[20,10],constrained_layout=True)

    import matplotlib.gridspec as gridspec

    gs0 = gridspec.GridSpec(1, 4, figure=fig)

    gs1 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[0])
    for n in range(2):
```

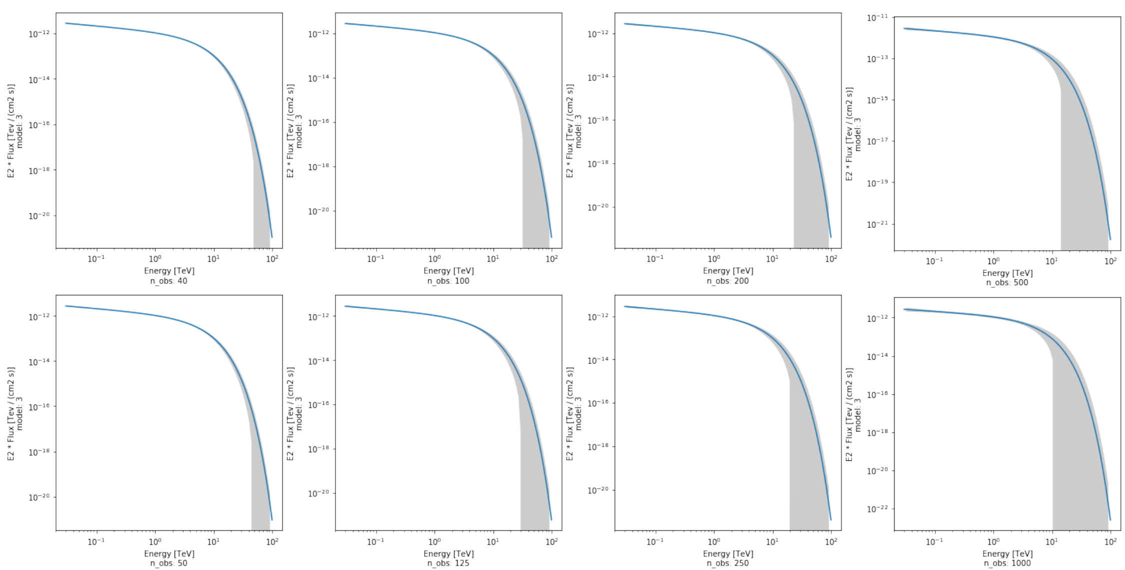
```
        ax = fig.add_subplot(gs1[n])
        sim_1[n][i].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim_1[n][i], covar=np.mean(covar_1[n][i], axis = 0),energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[n]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {i}')

    gs2 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[1])
    for n in range(2):
        ax = fig.add_subplot(gs2[n])
        sim_1[n+2][i].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim_1[n+2][i], covar=np.mean(covar_1[n+2][i], axis = 0),energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[n+2]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {i}')

    gs3 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[2])
    for n in range(2):
        ax = fig.add_subplot(gs3[n])
        sim_1[n+4][i].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim_1[n+4][i], covar=np.mean(covar_1[n+4][i], axis = 0),energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[n+4]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {i}')

    gs4 = gridspec.GridSpecFromSubplotSpec(2, 1, subplot_spec=gs0[3])
    for n in range(2):
        ax = fig.add_subplot(gs4[n])
        sim_1[n+6][i].plot(energy_range=energy_range, energy_power=2)
        plot_error(self=sim_1[n+6][i], covar=np.mean(covar_1[n+6][i], axis = 0),energy_range=energy_range, energy_power=2)
        plt.xlabel(f'Energy [TeV]\nn_obs: {n_obs[n+6]}')
        plt.ylabel(f'E2 * Flux [Tev / (cm2 s)]\nmodel: {i}')

    plt.show()
```


E2 * Flux [Tev / (cm2 s)]
model: 0
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000

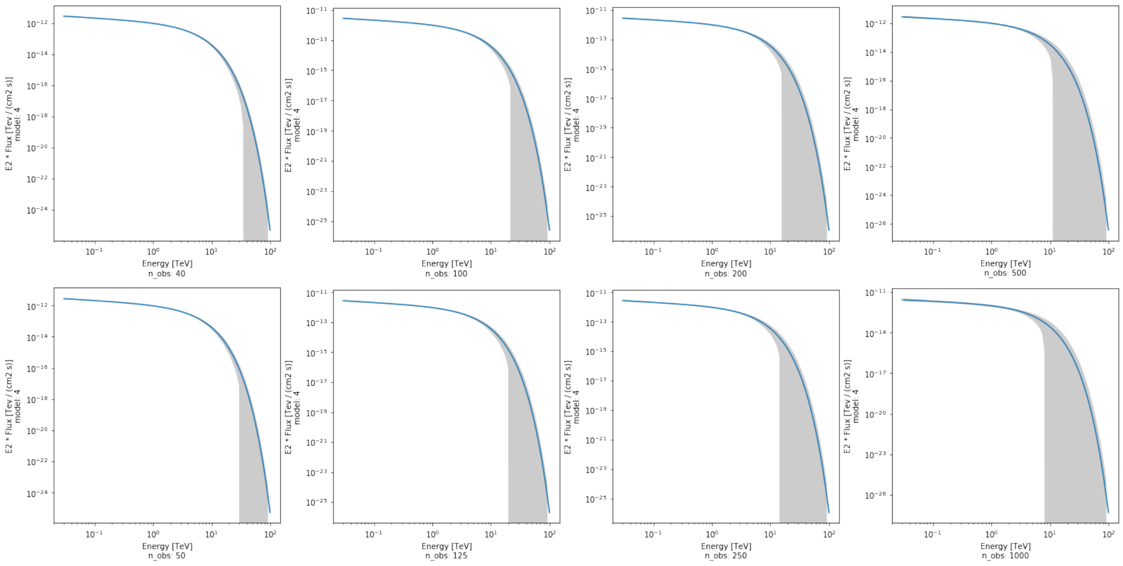

E2 * Flux [Tev / (cm2 s)]
model: 1
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000

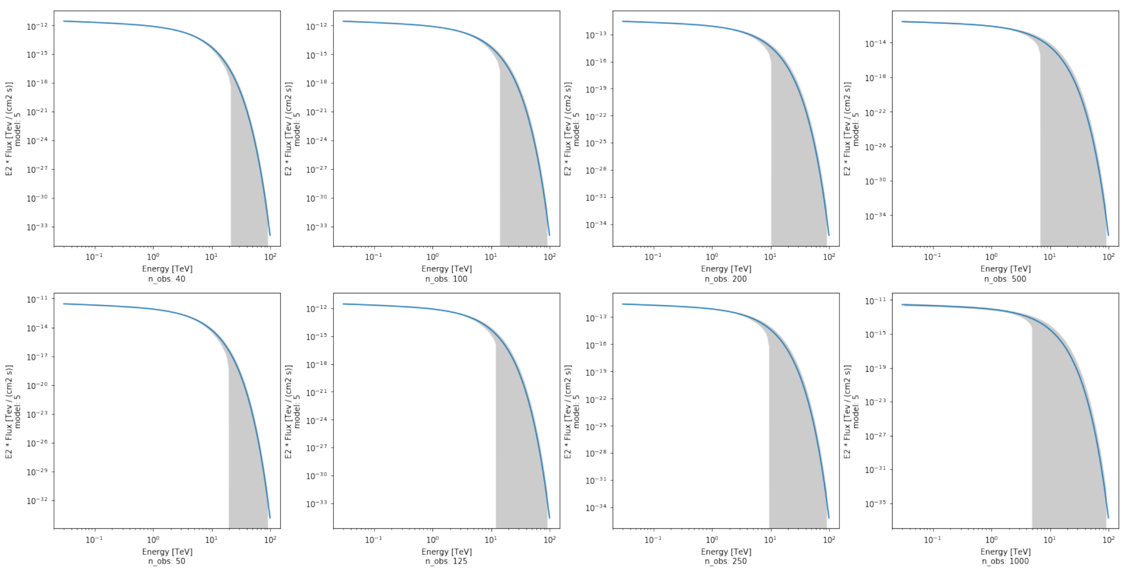


E2 * Flux [Tev / (cm2 s)]
model: 2
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000



E2 * Flux [Tev / (cm2 s)]
model: 3
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000


E2 * Flux [Tev / (cm2 s)]
model: 4
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000


E2 * Flux [Tev / (cm2 s)]
model: 5
Energy [TeV]
n_obs: 40
n_obs: 100
n_obs: 200
n_obs: 500
n_obs: 50
n_obs: 125
n_obs: 250
n_obs: 1000

[ ]: