pendulum

February 16, 2026

[46]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit

data_file = "pendulumData.csv"
df = pd.read_csv(data_file)
```

[47]:
```
df.head(10)
```

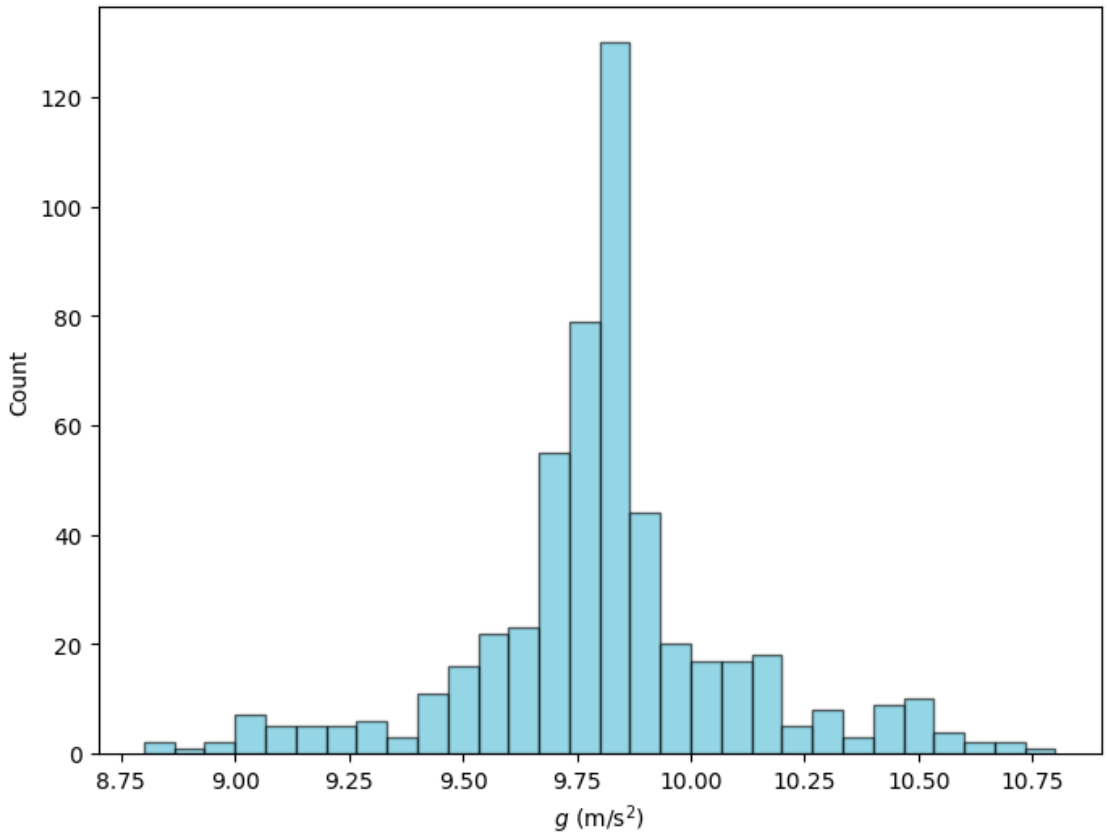
[47]:

	label	g (m/s2)	stat uncert	sys uncert	total uncert
0	20.Sep.X.B01.01	9.8300	0.0075	"+0.35,-0.44"	NaN
1	20.Sep.X.B01.02	9.0600	0.14	0.23	NaN
2	20.Sep.X.B01.03	9.7900	0.02	0.02	NaN
3	20.Sep.X.B01.04	9.8820	0.016	0.13	NaN
4	20.Sep.X.B01.05	978.0000	17	8	NaN
5	20.Sep.X.B02.01	9.8070	0.001	0.008	NaN
6	20.Sep.X.B02.02	9.2800	NaN	NaN	0.47
7	20.Sep.X.B02.03	9.8110	NaN	NaN	0.02
8	20.Sep.X.B02.04	9.2600	0.1	0.07	NaN
9	20.Sep.X.B03.01	9.8095	0.0035	0.0072	NaN

[48]:
```
xmin, xmax = 8.8, 10.8
df_cleaned = df[(df['g (m/s2)'] > xmin) & (df['g (m/s2)'] < xmax)]
len(df_cleaned)
```

[48]: 532

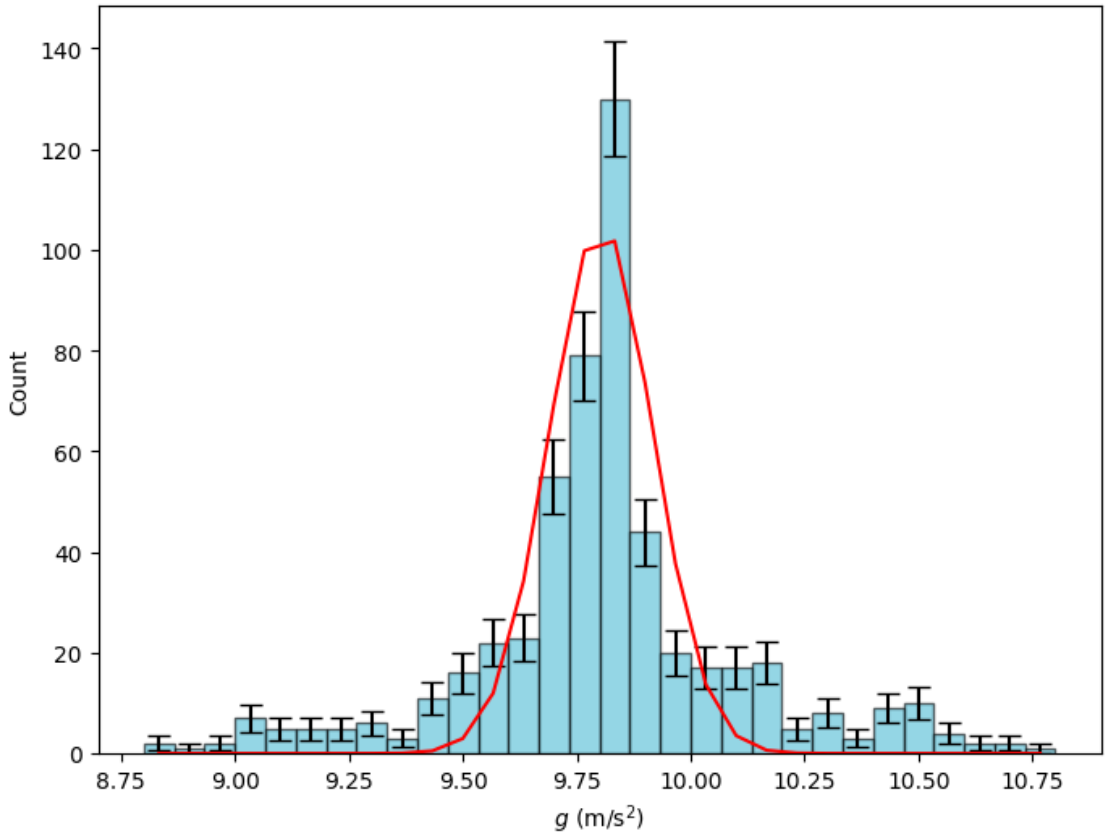
[49]:
```
g = df_cleaned['g (m/s2)'].values
plt.figure(figsize=(8, 6))
n, bins, _ = plt.hist(g, bins=30, range=(xmin, xmax), alpha=0.6, label=r'$g$ (m/
↪s$^2$)', color='#4dbbd5', edgecolor='black')
plt.xlabel(r'$g$ (m/s$^2$)')
plt.ylabel('Count')
plt.show()
```



[50]:
```
def gaussian(x, mu, sigma):
    return (1 / (sigma * np.sqrt(2 * np.pi))) * np.exp(-0.5 * ((x - mu) /␣
 ↪sigma) ** 2) * 30

uncertainty = np.sqrt(n)
bin_centers = (bins[:-1] + bins[1:]) / 2
popt, pcov = curve_fit(
    gaussian,
    bin_centers,
    n,
    p0=[9, 1],
    # sigma=uncertainty,
    absolute_sigma=True,
)

mu_fit, sigma_fit = popt
mu_err, sigma_err = np.sqrt(np.diag(pcov))
print(f"Fitted mean (mu): {mu_fit:.4f} ± {mu_err:.4f} m/s²")
print(f"Fitted standard deviation (sigma): {sigma_fit:.4f} ± {sigma_err:.4f} m/
 ↪s²")
```

```
Fitted mean (mu): 9.8037 ± 0.0009 m/s²
Fitted standard deviation (sigma): 0.1137 ± 0.0007 m/s²
```

[51]:
```
n_fit = gaussian(bin_centers, *popt)

plt.figure(figsize=(8, 6))
plt.hist(g, bins=30, range=(xmin, xmax), alpha=0.6, label=r'$g$ (m/s$^2$)',␣
↪color='#4dbbd5', edgecolor='black')
plt.errorbar(bin_centers, n, yerr=uncertainty, fmt='none', ecolor='black',␣
↪capsize=5, label='Data with Uncertainty')
plt.plot(bin_centers, n_fit, color='red', label='Gaussian Fit')
plt.xlabel(r'$g$ (m/s$^2$)')
plt.ylabel('Count')
plt.show()
```



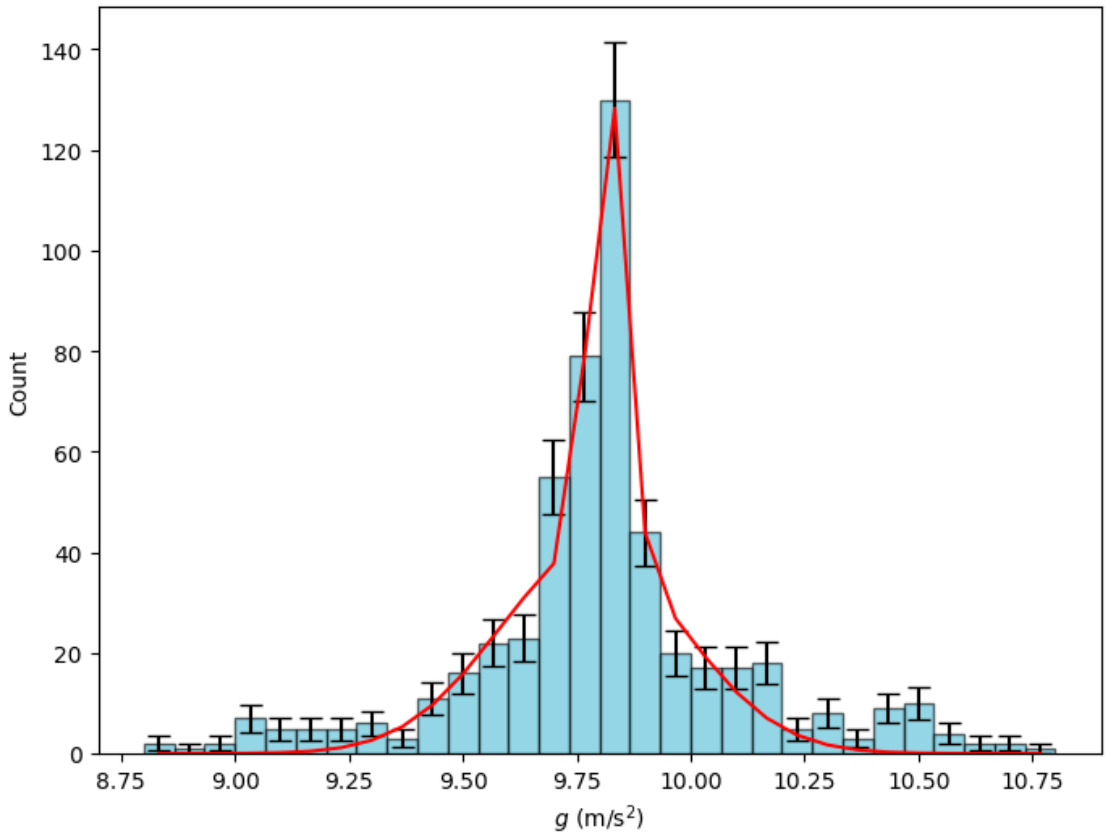
[52]:
```
chi_square = np.sum(((n - n_fit) ** 2) / n_fit)
ndf = len(n) - len(popt)
print(f"chi^2 / ndf: {chi_square:.2f} / {ndf}")
```

```
chi^2 / ndf: 290749890731430.44 / 28
```

[53]:
```
def model(x, mu_1, sigma_1, mu_2, sigma_2, p):
    """A mixture of two Gaussian distributions."""
    return p * gaussian(x, mu_1, sigma_1) + (1 - p) * gaussian(x, mu_2, sigma_2)

popt_mixture, pcov_mixture = curve_fit(
    model,
    bin_centers,
    n,
    p0=[9.8, 0.2, 9.8, 0.2, 0.1],
    absolute_sigma=True,
)
mu_1_fit, sigma_1_fit, mu_2_fit, sigma_2_fit, p_fit = popt_mixture
mu_1_err, sigma_1_err, mu_2_err, sigma_2_err, p_err = np.sqrt(np.
 ↪diag(pcov_mixture))
print(f"Fitted mean 1 (mu_1): {mu_1_fit:.4f} ± {mu_1_err:.4f} m/s²")
print(f"Fitted std dev 1 (sigma_1): {sigma_1_fit:.4f} ± {sigma_1_err:.4f} m/s²")
print(f"Fitted mean 2 (mu_2): {mu_2_fit:.4f} ± {mu_2_err:.4f} m/s²")
print(f"Fitted std dev 2 (sigma_2): {sigma_2_fit:.4f} ± {sigma_2_err:.4f} m/s²")
print(f"Fitted mixing parameter (p): {p_fit:.4f} ± {p_err:.4f}")
```

```
Fitted mean 1 (mu_1): 9.8184 ± 0.0008 m/s²
Fitted std dev 1 (sigma_1): 0.0382 ± 0.0009 m/s²
Fitted mean 2 (mu_2): 9.7819 ± 0.0034 m/s²
Fitted std dev 2 (sigma_2): 0.2067 ± 0.0032 m/s²
Fitted mixing parameter (p): 0.3088 ± 0.0074
```

[54]:
```
n_fit_mixture = model(bin_centers, *popt_mixture)

plt.figure(figsize=(8, 6))
plt.hist(g, bins=30, range=(xmin, xmax), alpha=0.6, label=r'$g$ (m/s$^2$)',␣
 ↪color='#4dbbd5', edgecolor='black')
plt.errorbar(bin_centers, n, yerr=uncertainty, fmt='none', ecolor='black',␣
 ↪capsize=5, label='Data with Uncertainty')
plt.plot(bin_centers, n_fit_mixture, color='red', label='Mixed Gaussian Fit')
plt.xlabel(r'$g$ (m/s$^2$)')
plt.ylabel('Count')
plt.show()
```



[55]:
```
chi_square_mixture = np.sum(((n - n_fit_mixture) ** 2) / n_fit_mixture)
ndf = len(n) - len(popt)
print(f"chi^2 / ndf: {chi_square_mixture:.2f} / {ndf}")
```

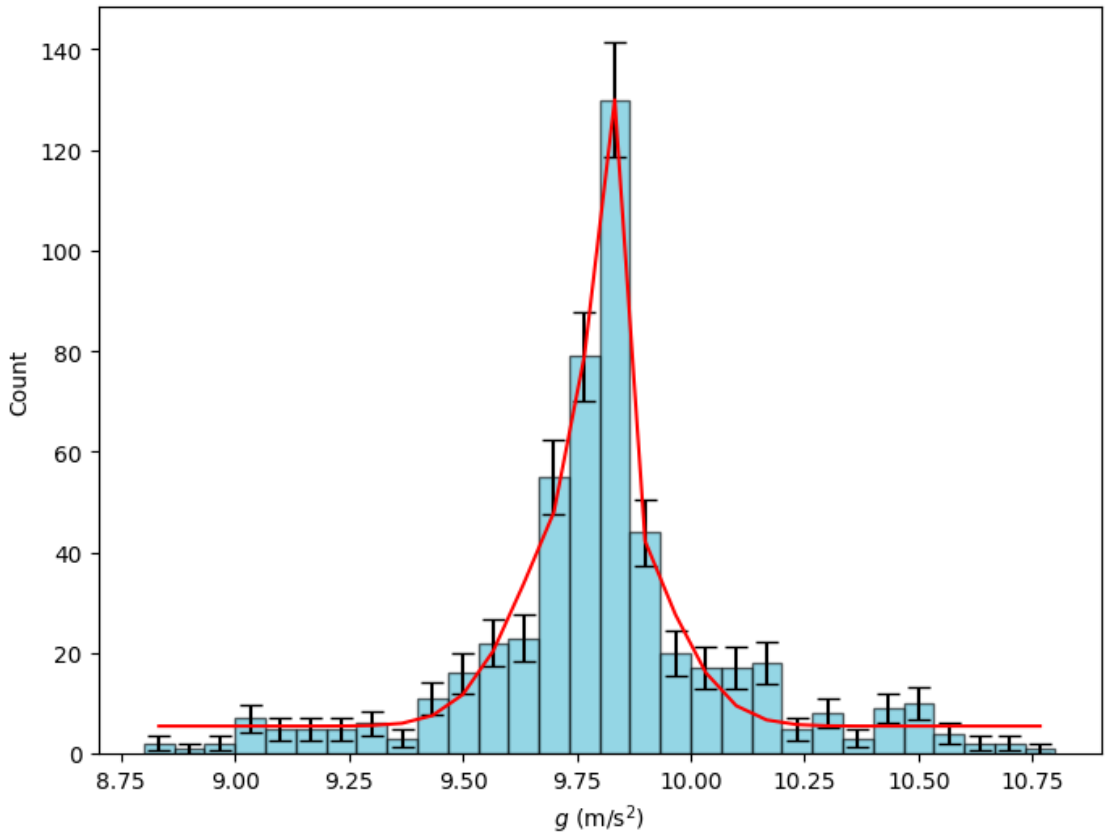
```
chi^2 / ndf: 11690.98 / 28
```

[56]:
```
def model_with_const(x, mu_1, sigma_1, mu_2, sigma_2, p, c):
    """A mixture of two Gaussian distributions, plus a constant background."""
    return p * gaussian(x, mu_1, sigma_1) + (1 - p) * gaussian(x, mu_2,␣
 ↪sigma_2) + c

popt_mixture_c, pcov_mixture_c = curve_fit(
    model_with_const,
    bin_centers,
    n,
    p0=[9.8, 0.1, 9.8, 0.1, 0.1, 2],
    absolute_sigma=True,
)
mu_1_fit, sigma_1_fit, mu_2_fit, sigma_2_fit, p_fit, c_fit = popt_mixture_c
```

```
mu_1_err, sigma_1_err, mu_2_err, sigma_2_err, p_err, c_err = np.sqrt(np.
  ↪diag(pcov_mixture_c))
print(f"Fitted mean 1 (mu_1): {mu_1_fit:.4f} ± {mu_1_err:.4f} m/s²")
print(f"Fitted std dev 1 (sigma_1): {sigma_1_fit:.4f} ± {sigma_1_err:.4f} m/s²")
print(f"Fitted mean 2 (mu_2): {mu_2_fit:.4f} ± {mu_2_err:.4f} m/s²")
print(f"Fitted std dev 2 (sigma_2): {sigma_2_fit:.4f} ± {sigma_2_err:.4f} m/s²")
print(f"Fitted mixing parameter (p): {p_fit:.4f} ± {p_err:.4f}")
print(f"Fitted constant (c): {c_fit:.4f} ± {c_err:.4f}")
```

```
Fitted mean 1 (mu_1): 9.8055 ± 0.0002 m/s²
Fitted std dev 1 (sigma_1): 0.0173 ± 0.0002 m/s²
Fitted mean 2 (mu_2): 9.7858 ± 0.0021 m/s²
Fitted std dev 2 (sigma_2): 0.1393 ± 0.0028 m/s²
Fitted mixing parameter (p): 0.4012 ± 0.0101
Fitted constant (c): 5.4658 ± 0.2322
```

[57]:
```
n_fit_mixture_c = model_with_const(bin_centers, *popt_mixture_c)

plt.figure(figsize=(8, 6))
plt.hist(g, bins=30, range=(xmin, xmax), alpha=0.6, label=r'$g$ (m/s$^2$)',␣
  ↪color='#4dbbd5', edgecolor='black')
plt.errorbar(bin_centers, n, yerr=uncertainty, fmt='none', ecolor='black',␣
  ↪capsize=5, label='Data with Uncertainty')
plt.plot(bin_centers, n_fit_mixture_c, color='red', label='Mixed Gaussian Fit␣
  ↪(with Const)')
plt.xlabel(r'$g$ (m/s$^2$)')
plt.ylabel('Count')
plt.show()
```



[58]:
```
chi_square_mixture_c = np.sum(((n - n_fit_mixture_c) ** 2) / n_fit_mixture_c)
ndf = len(n) - len(popt)
print(f"chi^2 / ndf: {chi_square_mixture_c:.2f} / {ndf}")
```

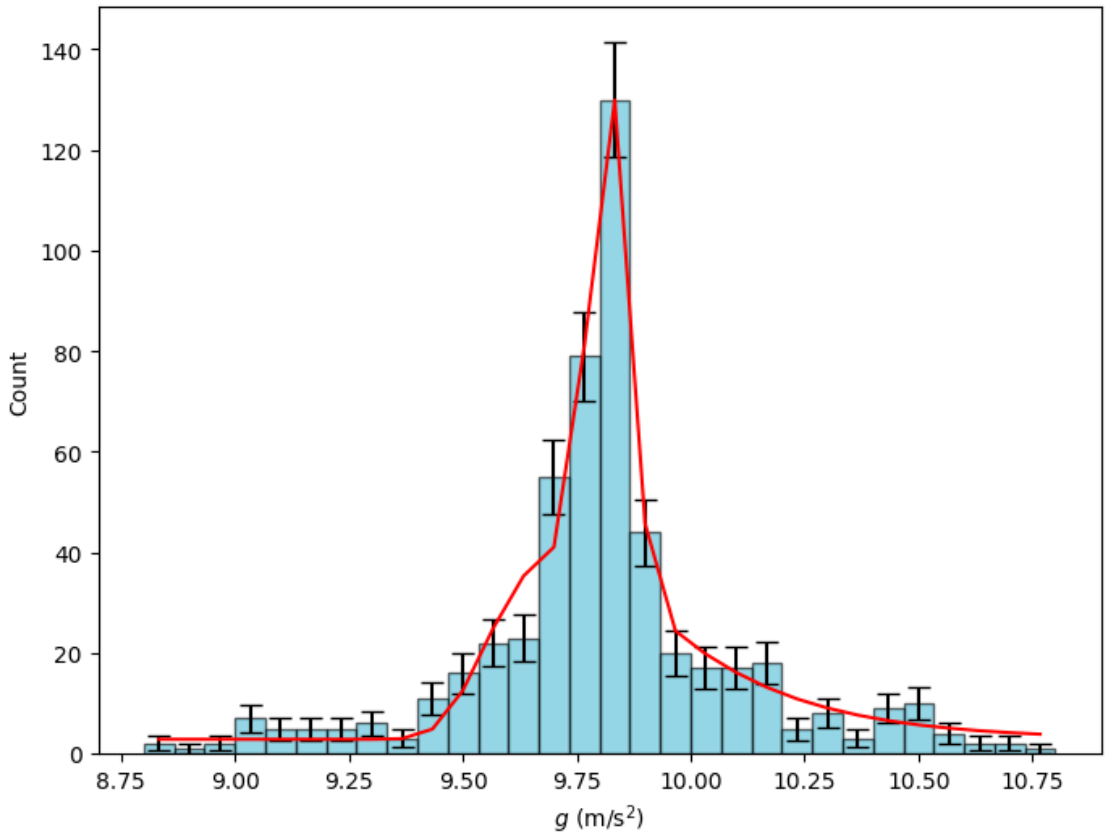
```
chi^2 / ndf: 61.88 / 28
```

[59]:
```
# Try Moyal approximation of Landau distribution
from scipy.stats import moyal
def model_with_moyal(x, mu_1, sigma_1, mu_2, sigma_2, p, c):
    """A mixture of a Gaussian and a Moyal distribution, plus a constant␣
 ↪background."""
    moyal_part = (1 - p) * moyal.pdf(x, loc=mu_2, scale=sigma_2) * 30
    gaussian_part = p * gaussian(x, mu_1, sigma_1)
    return gaussian_part + moyal_part + c

popt_mixture_m, pcov_mixture_m = curve_fit(
    model_with_moyal,
    bin_centers,
    n,
    p0=[9.8, 0.1, 9.2, 0.1, 0.1, 1],
```

```
    absolute_sigma=True,
)
mu_1_fit, sigma_1_fit, mu_2_fit, sigma_2_fit, p_fit, c_fit = popt_mixture_m
mu_1_err, sigma_1_err, mu_2_err, sigma_2_err, p_err, c_err = np.sqrt(np.
 ↪diag(pcov_mixture_m))
print(f"Fitted mean 1 (mu_1): {mu_1_fit:.4f} ± {mu_1_err:.4f} m/s²")
print(f"Fitted std dev 1 (sigma_1): {sigma_1_fit:.4f} ± {sigma_1_err:.4f} m/s²")
print(f"Fitted mean 2 (mu_2): {mu_2_fit:.4f} ± {mu_2_err:.4f} m/s²")
print(f"Fitted std dev 2 (sigma_2): {sigma_2_fit:.4f} ± {sigma_2_err:.4f} m/s²")
print(f"Fitted mixing parameter (p): {p_fit:.4f} ± {p_err:.4f}")
print(f"Fitted constant (c): {c_fit:.4f} ± {c_err:.4f}")
```

```
Fitted mean 1 (mu_1): 9.8210 ± 0.0007 m/s²
Fitted std dev 1 (sigma_1): 0.0416 ± 0.0007 m/s²
Fitted mean 2 (mu_2): 9.7155 ± 0.0035 m/s²
Fitted std dev 2 (sigma_2): 0.1284 ± 0.0020 m/s²
Fitted mixing parameter (p): 0.3472 ± 0.0065
Fitted constant (c): 2.8456 ± 0.1827
```

[60]:
```
n_fit_mixture_m = model_with_moyal(bin_centers, *popt_mixture_m)

plt.figure(figsize=(8, 6))
plt.hist(g, bins=30, range=(xmin, xmax), alpha=0.6, label=r'$g$ (m/s$^2$)',␣
 ↪color='#4dbbd5', edgecolor='black')
plt.errorbar(bin_centers, n, yerr=uncertainty, fmt='none', ecolor='black',␣
 ↪capsize=5, label='Data with Uncertainty')
plt.plot(bin_centers, n_fit_mixture_m, color='red', label='Gaussian + Moyal Fit␣
 ↪(with Const)')
plt.xlabel(r'$g$ (m/s$^2$)')
plt.ylabel('Count')
plt.show()
```



[61]:
```
chi_square_mixture_m = np.sum(((n - n_fit_mixture_m) ** 2) / n_fit_mixture_m)
ndf = len(n) - len(popt)
print(f"chi^2 / ndf: {chi_square_mixture_m:.2f} / {ndf}")
```

```
chi^2 / ndf: 52.45 / 28
```

[ ]:
```
from scipy.stats import chi2

p_value_gaussian = 1 - chi2.cdf(chi_square, ndf)
p_value_mixture = 1 - chi2.cdf(chi_square_mixture, ndf)
p_value_mixture_c = 1 - chi2.cdf(chi_square_mixture_c, ndf)
p_value_mixture_m = 1 - chi2.cdf(chi_square_mixture_m, ndf)
print(f"Gaussian Fit p-value: {p_value_gaussian:.4e}")
print(f"Mixed Gaussian Fit p-value: {p_value_mixture:.4e}")
print(f"Mixed Gaussian + Const Fit p-value: {p_value_mixture_c:.4e}")
print(f"Gaussian + Moyal + Const Fit p-value: {p_value_mixture_m:.4e}")
```

```
Gaussian Fit p-value: 0.0000e+00
Mixed Gaussian Fit p-value: 0.0000e+00
Mixed Gaussian + Const Fit p-value: 2.3323e-04
Gaussian + Moyal + Const Fit p-value: 3.4009e-03
```