Building on the con- figuration of exercise 7.2 ($\Delta t = 0.01$ s, $x_0 = 0$, $L = 100$), we now introduce a multi- plicative noise:

$$\sigma(x) = \sigma_0 + \frac{\Delta\sigma}{L}x,$$

where $x \in \left[-\frac{L}{2}, \frac{L}{2}\right]$, and σ_0 and $\delta\sigma$ are such that $\sigma(x) > 0$ all over the domain (e.g., take $\sigma_0 = 1$ and $\Delta\sigma = 1.8$, so that at the extremes of the domain we have $\sigma\left(-\frac{L}{2}\right) = 0.1$ and $\sigma\left(-\frac{L}{2}\right) = 1.9$). In a similar way to exercise 7.2, we aim to build the probability distribution $p(x)$ of finding the particle at a given position of the interval—and we expect to find a flat distribution, because no deterministic external forces are acting on the particle.

a. Using equation (7.7), simulate $N = 104$ independent trajectories for various durations (e.g., $T = 10, 10^2, 10^3, 10^4$, and 10^5 s) and plot the histograms of the final positions. Compare your results with figures 7.3(c) and 7.3(d).

In [4]:

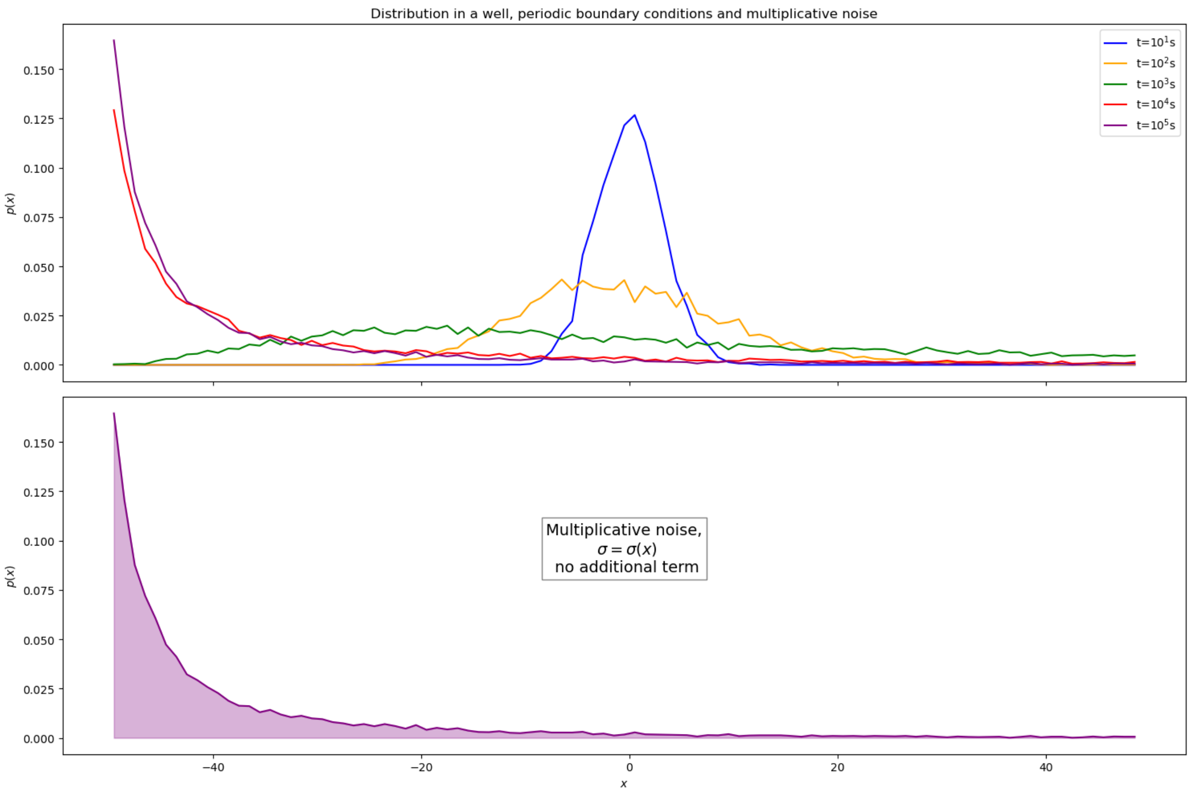
```
import numpy as np
from matplotlib import pyplot as plt
from tqdm import trange

A = 7
steps = 10**A+1          # Number of iterations
N = 10000
x = np.zeros((1,N))           # Pre allocation of the positions
L = 100
sigma0 = 1
sigma = 1.8
beta = sigma/L
dt = np.sqrt(0.01)
results = np.zeros((5,N))
counter = 0

for i in trange(steps):
    x = x + (sigma0 + beta * x) * np.random.choice([-dt, dt], size=
    x = np.where(x > L/2, L-x, x)
    x = np.where(x < -L/2, -L-x, x)

    if i in [10**(A-4),10**(A-3),10**(A-2),10**(A-1),10**(A)]:
        results[counter, :] = x
        counter += 1
```

```
100%|██████████| 10000001/10000001 [13:32<00:00, 12301.19it/s]
```

In [5]:

```
l = np.arange(100) - 50
h10 = np.histogram(results[0, :], l, density=True)
h100 = np.histogram(results[1, :], l, density=True)
h1000 = np.histogram(results[2, :], l, density=True)
h10000 = np.histogram(results[3, :], l, density=True)
```

```
h100000 = np.histogram(results[4, :], l, density=True)

# Create a figure with subplots
fig, axs = plt.subplots(2, 1, figsize=(15, 10), sharex=True)

# Plot for the first subplot
axs[0].plot(h10[1][:-1] + 1/2, h10[0], color='blue')
axs[0].plot(h100[1][:-1] + 1/2, h100[0], color='orange')
axs[0].plot(h1000[1][:-1] + 1/2, h1000[0], color='green')
axs[0].plot(h10000[1][:-1] + 1/2, h10000[0], color='red')
axs[0].plot(h100000[1][:-1] + 1/2, h100000[0], color='purple')

axs[0].legend(['t=$10^1$s', 't=$10^2$s', 't=$10^3$s', 't=$10^4$s',
axs[0].set_ylabel('$p(x)$')
axs[0].set_title('Distribution in a well, periodic boundary conditi

# Plot for the second subplot (same as the last histogram)
axs[1].plot(h100000[1][:-1] + 1/2, h100000[0], color='purple')
axs[1].fill_between(h100000[1][:-1] + 1/2, h100000[0], color='purpl
axs[1].set_ylabel('$p(x)$')
axs[1].set_xlabel('$x$')
axs[1].set_ylim(axs[0].get_ylim())
axs[1].text(0.5, 0.65, 'Multiplicative noise,\n $\sigma = \sigma(x)
            verticalalignment='top', horizontalalignment='center', 
            fontsize=14, bbox=dict(facecolor='white', alpha=0.5))

plt.tight_layout()
plt.show()
```



b)

Why is the final distribution non-uniform?

"Because the steps from the rigth to the left are bigger, compared the jumps from the left to the right due to the changing sigma(x). The result is that the particles spend more time in the left half and thus the distribution is asymmetric."

Is this compatible with a Brownian particle at thermodynamic equilibrium with its environment?

No, under thermodynamic equlibrium we expect the particles uniformly distributed.