task2

September 21, 2024

0.1 Laplacian of Gaussian

The Laplacian of Gaussian (LoG) is the second derivative of the Gaussian function

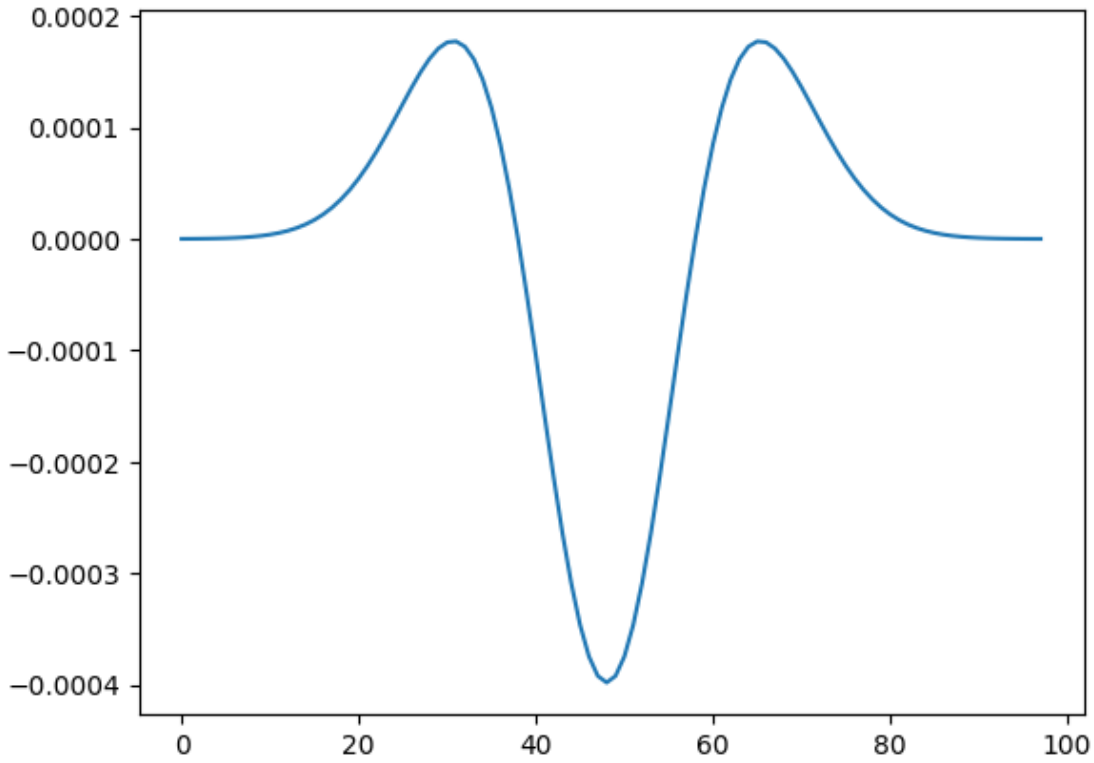
Gaussian function: $G_\sigma = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}$

LoG: $\nabla^2 G_\sigma = \frac{\partial^2 f}{\partial x^2}$

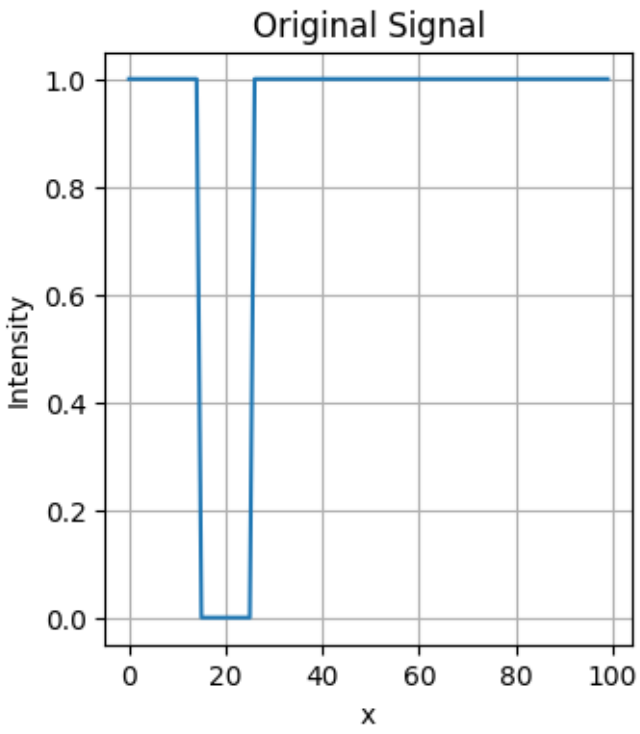
[66]:
```
import numpy as np
import matplotlib.pyplot as plt
```

[67]:
```
def laplacian_of_gaussian(sigma: float, size: int) -> np.ndarray:
    x = np.arange(-size // 2 + 1, size // 2 + 1)
    gaussian = np.exp(-0.5 * (x / sigma) ** 2) / (sigma * np.sqrt(2 * np.pi))
    laplacian = np.diff(gaussian, n=2)
    return laplacian
```

[68]:
```
log = laplacian_of_gaussian(10, 100)
plt.plot(log)
plt.show()
```

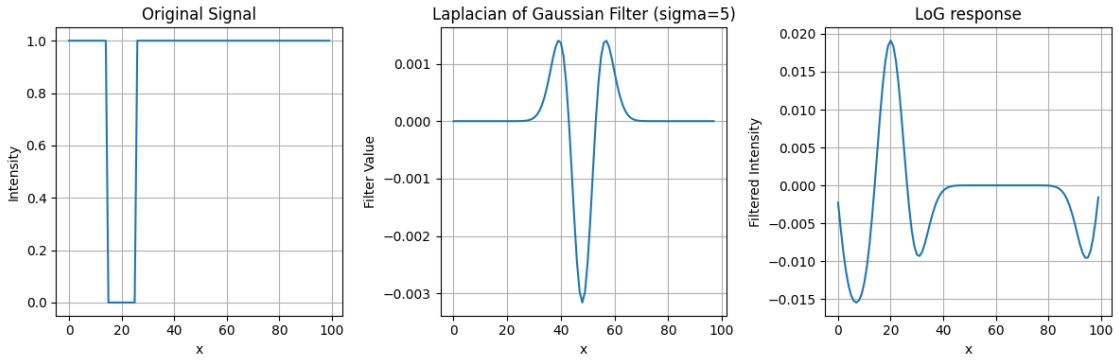


[69]:
```
# Create a 1D blob signal
blob_size = 10
blob_center = 20
signal_length = 100
signal = np.ones(signal_length)
signal[blob_center - blob_size // 2 : blob_center + blob_size // 2 + 1] = 0
plt.figure(figsize=(12, 4))
plt.subplot(1, 3, 1)
plt.plot(signal)
plt.title("Original Signal")
plt.xlabel("x")
plt.ylabel("Intensity")
plt.grid(True)
plt.show()
```



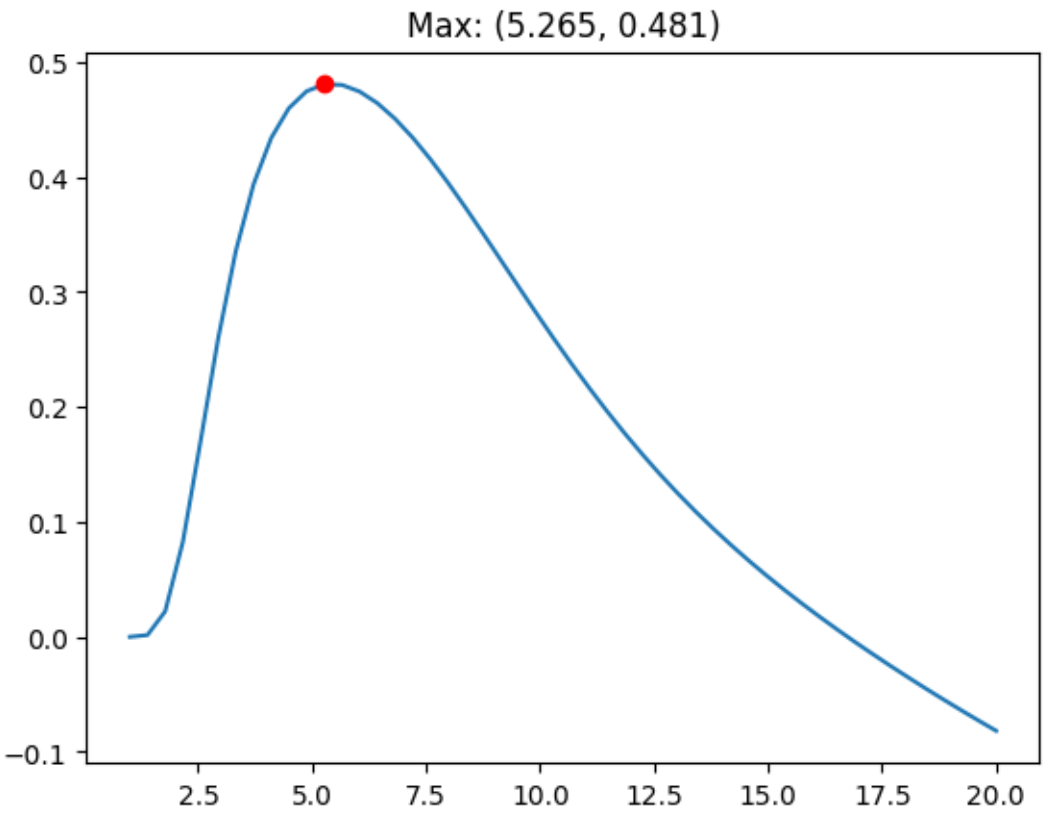
[70]:
```python
# Apply the LoG to the above signal at various sigma values and plot the result.
# use np.convolve
sigma = 5
size = signal_length
log_filter = laplacian_of_gaussian(sigma, size)
filtered_signal = np.convolve(signal, log_filter, mode="same")

# Plot the original signal, the LoG filter, and the filtered signal
plt.figure(figsize=(12, 4))
plt.subplot(1, 3, 1)
plt.plot(signal)
plt.title("Original Signal")
plt.xlabel("x")
plt.ylabel("Intensity")
plt.grid(True)

plt.subplot(1, 3, 2)
plt.plot(log_filter)
plt.title(f"Laplacian of Gaussian Filter (sigma={sigma})")
plt.xlabel("x")
plt.ylabel("Filter Value")
plt.grid(True)
```

```
plt.subplot(1, 3, 3)
plt.plot(filtered_signal)
plt.title("LoG response")
plt.xlabel("x")
plt.ylabel("Filtered Intensity")
plt.grid(True)

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



[71]:
```
# multiplying by sigma^2 to normalize the response
def normalized_laplacian_of_gaussian(sigma: float, size: int) -> np.ndarray:
    return sigma**2 * laplacian_of_gaussian(sigma, size)
```

[92]:
```
sigmas = np.linspace(1, 20)
results = []
for sigma in sigmas:
    log_filter = normalized_laplacian_of_gaussian(sigma, size)
    result = np.convolve(signal, log_filter, mode="same")
    results.append(result[blob_center])

sigma_max_index = np.argmax(results)
sigma_max = sigmas[sigma_max_index]
result_max = results[sigma_max_index]
plt.plot(sigmas, results)
plt.plot(sigma_max, result_max, "ro")
plt.title(f"Max: ({sigma_max:.3f}, {result_max:.3f})")
plt.show()
```


Max: (5.265, 0.481)
0.5
0.4
0.3
0.2
0.1
0.0
−0.1
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0