```
import numpy as np
import matplotlib.pyplot as plt

# Simulate energy consumption data
np.random.seed(42)
hours = np.arange(0, 24, 1)  # 24-hour period
base_consumption = 100 + 50 * np.sin(2 * np.pi * hours / 24)  # Baseline energy usage (kWh)
noise = np.random.normal(0, 10, len(hours))  # Random noise
consumption = base_consumption + noise

# Optimization function: Reduce energy usage during peak hours
def optimize_energy(consumption, peak_hours=(8, 20), reduction_factor=0.8):
    optimized = consumption.copy()
    for i, hour in enumerate(hours):
        if peak_hours[0] <= hour <= peak_hours[1]:
            optimized[i] *= reduction_factor  # Reduce consumption during peak hours
    return optimized
```

```
# Calculate optimized consumption
optimized_consumption = optimize_energy(consumption)

# Calculate energy savings
savings = consumption - optimized_consumption
total_savings = np.sum(savings)

# Visualization
plt.figure(figsize=(10, 6))
plt.plot(hours, consumption, label='Baseline Consumption
(kWh)', color='red', linestyle='--')
plt.plot(hours, optimized_consumption, label='Optimized
Consumption (kWh)', color='green')
plt.fill_between(hours, consumption,
optimized_consumption, color='green', alpha=0.2,
label='Savings')
plt.title(f'Energy Efficiency Optimization\nTotal
Savings: {total_savings:.2f} kWh')
plt.xlabel('Hour of Day')
plt.ylabel('Energy Consumption (kWh)')
plt.grid(True)
plt.legend()
```

```
plt.legend()
plt.tight_layout()

# Display plot
plt.show()

# Print summary
print(f"Total Energy Savings: {total_savings:.2f} kWh")
print(f"Peak Hours (Reduced): {hours[8]}:00 to
{hours[20]}:00")
```