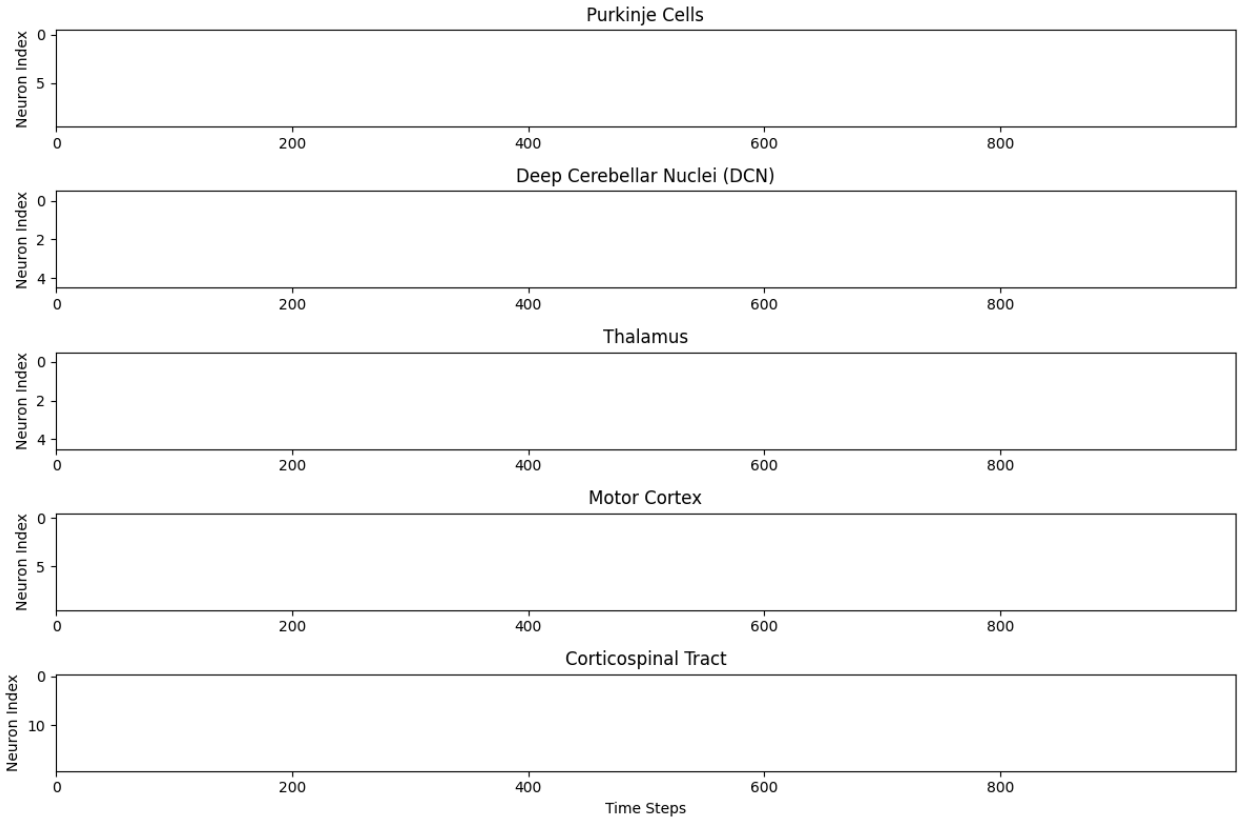
Code [1]

Output:

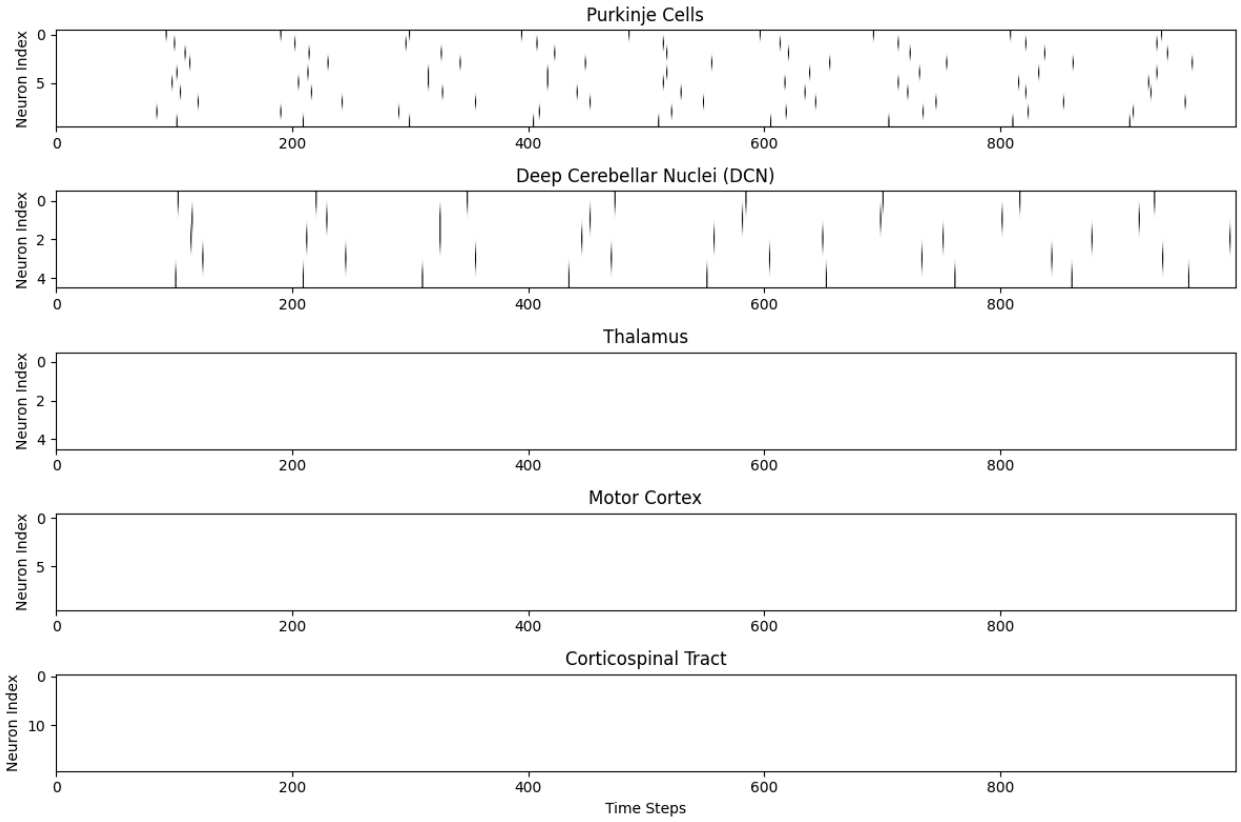
default



Code [2]

Output:

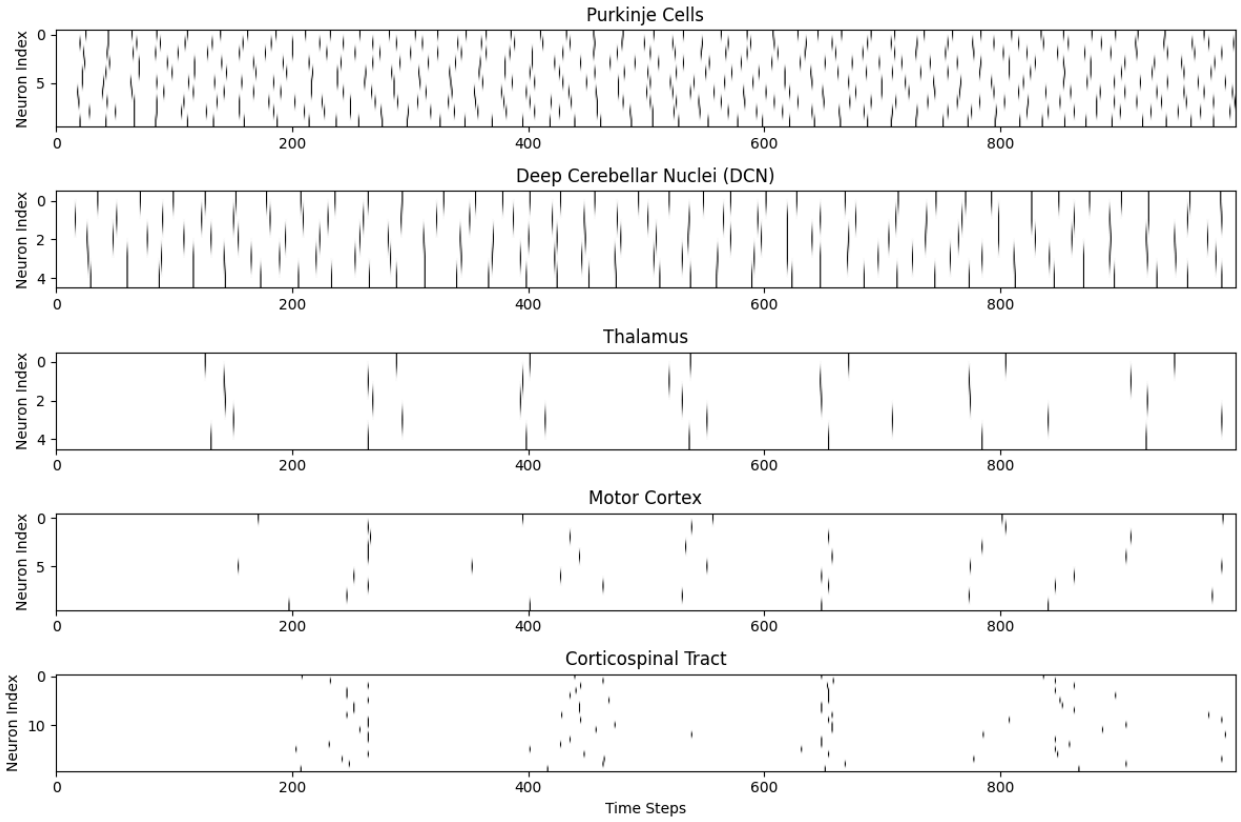
#Added Noise and increased weights



Code [3]

Output:

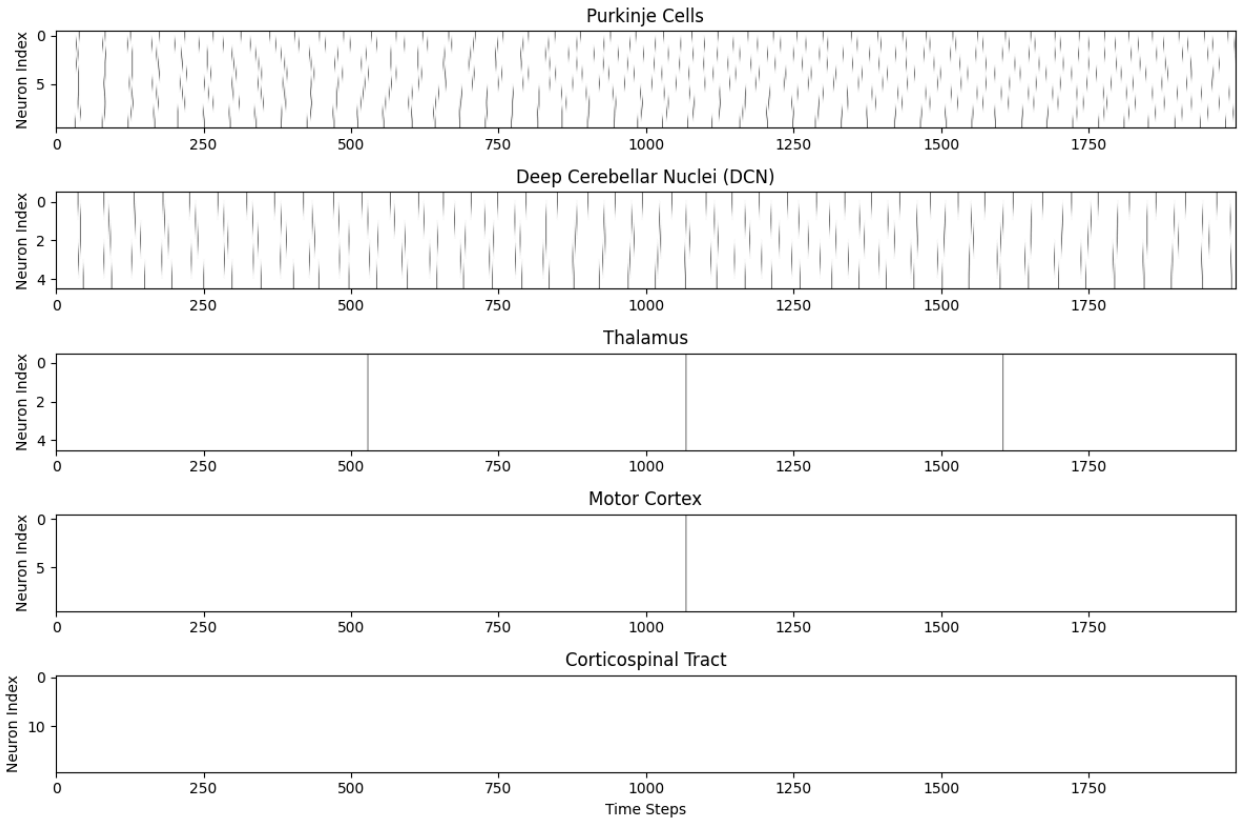
#increased noise and weights



Code [4]

Output:

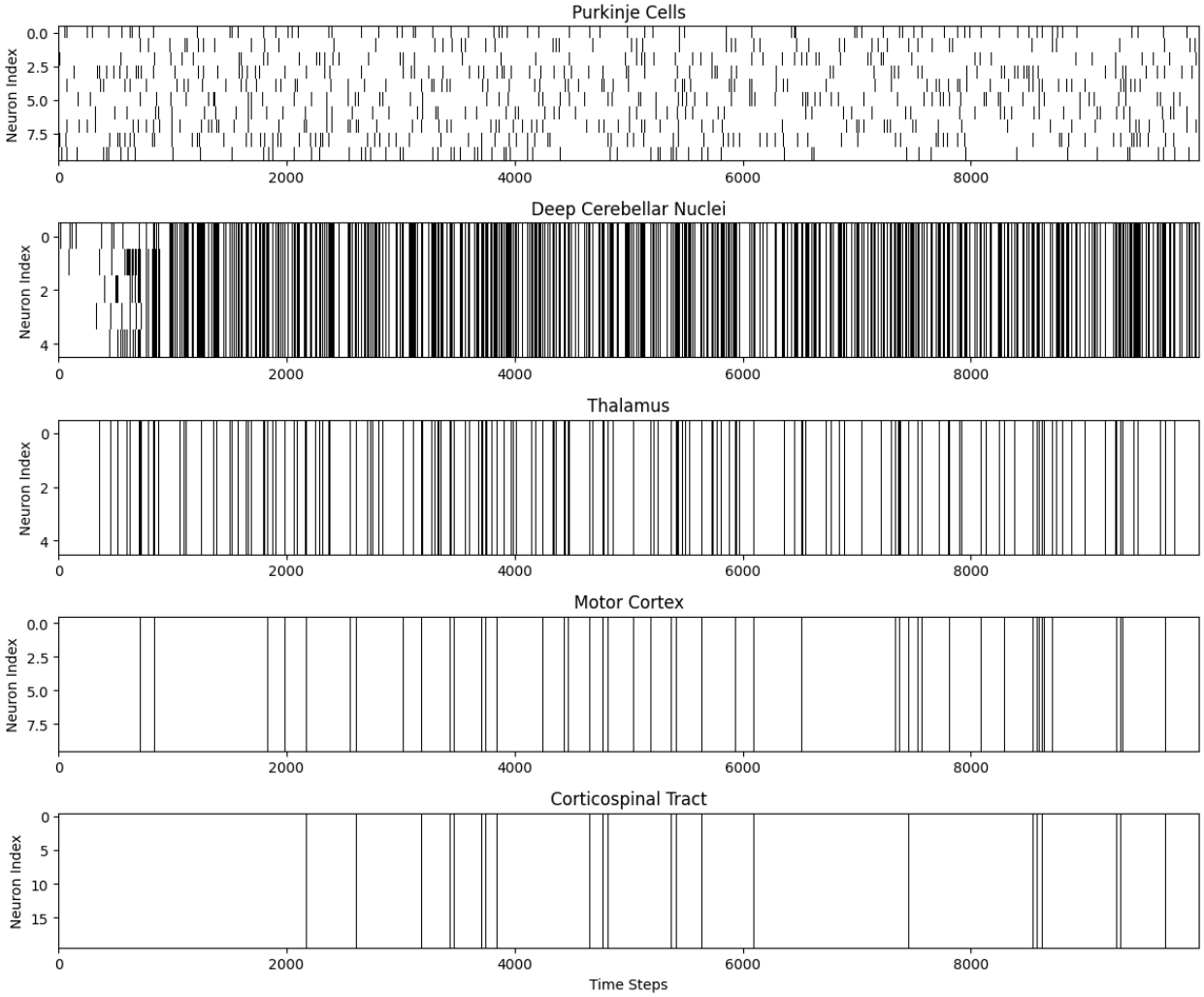
#added the extensions



Code [5]

Output:

#improved extensions



[1]

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


# Parameters
time_steps = 1000  # Total simulation time steps
dt = 0.1  # Time step size (ms)
num_purkinje = 10  # Number of Purkinje cells
num_dcn = 5  # Number of deep cerebellar nuclei (DCN) neurons
num_thalamus = 5  # Number of thalamic neurons
num_motor_cortex = 10  # Number of motor cortex neurons
num_corticospinal = 20  # Number of corticospinal tract neurons


# Neuron model parameters
tau_mem = 20.0  # Membrane time constant (ms) for faster dynamics
threshold = 1.0  # Spiking threshold
resting_potential = 0.0  # Resting membrane potential
reset_potential = 0.0  # Reset potential after spike


# Synaptic weights (adjusted for stronger signal propagation)
w_purkinje_dcn = -1 * np.ones((num_dcn, num_purkinje))             # Inhibitory
weight from Purkinje to DCN
w_dcn_thalamus = 1 * np.ones((num_thalamus, num_dcn))              # Excitatory
weight from DCN to thalamus
w_thalamus_motor = 1 * np.ones((num_motor_cortex, num_thalamus))   # Excitatory
weight from thalamus to motor cortex
w_motor_corticospinal = 1 * np.ones((num_corticospinal, num_motor_cortex))  #
Excitatory weight from motor cortex to corticospinal tract


# Inputs (scaled for sufficient excitation)
climbing_fiber_input = np.random.rand(num_purkinje, time_steps)   # Climbing
fiber input to Purkinje cells
mossy_fiber_input = np.random.rand(num_dcn, time_steps)           # Mossy fiber
input to DCN


# Neuron states
purkinje_membrane = np.zeros((num_purkinje, time_steps))              #
Membrane potential of Purkinje cells
```

```
dcn_membrane = np.zeros((num_dcn, time_steps))                          #
Membrane potential of DCN neurons
thalamus_membrane = np.zeros((num_thalamus, time_steps))                #
Membrane potential of thalamic neurons
motor_membrane = np.zeros((num_motor_cortex, time_steps))               #
Membrane potential of motor cortex neurons
corticospinal_membrane = np.zeros((num_corticospinal, time_steps))      #
Membrane potential of corticospinal neurons


# Spikes
purkinje_spikes = np.zeros((num_purkinje, time_steps))                  # Spikes
of Purkinje cells
dcn_spikes = np.zeros((num_dcn, time_steps))                            # Spikes
of DCN neurons
thalamus_spikes = np.zeros((num_thalamus, time_steps))                  # Spikes
of thalamic neurons
motor_spikes = np.zeros((num_motor_cortex, time_steps))                 # Spikes
of motor cortex neurons
corticospinal_spikes = np.zeros((num_corticospinal, time_steps))        # Spikes
of corticospinal neurons


# Noise (background activity)
noise_amplitude = 0.5


# Simulation loop
for t in range(1, time_steps):
    # Purkinje cells
    purkinje_input = climbing_fiber_input[:, t]                         #
Climbing fiber input
    purkinje_membrane[:, t] = purkinje_membrane[:, t - 1] + dt / tau_mem *
(-purkinje_membrane[:, t - 1] + purkinje_input)
    purkinje_spikes[:, t] = (purkinje_membrane[:, t] >=
threshold).astype(float)
    purkinje_membrane[purkinje_spikes[:, t] == 1, t] = reset_potential


    # Deep cerebellar nuclei (DCN)
    dcn_input = mossy_fiber_input[:, t] + np.dot(w_purkinje_dcn,
purkinje_spikes[:, t])   # Mossy fiber + Purkinje input
    dcn_membrane[:, t] = dcn_membrane[:, t - 1] + dt / tau_mem *
(-dcn_membrane[:, t - 1] + dcn_input)
```

```
    dcn_spikes[:, t] = (dcn_membrane[:, t] >= threshold).astype(float)
    dcn_membrane[dcn_spikes[:, t] == 1, t] = reset_potential


    # Thalamus
    thalamus_input = np.dot(w_dcn_thalamus, dcn_spikes[:, t]) + noise_amplitude
* np.random.rand(num_thalamus)
    thalamus_membrane[:, t] = thalamus_membrane[:, t - 1] + dt / tau_mem *
(-thalamus_membrane[:, t - 1] + thalamus_input)
    thalamus_spikes[:, t] = (thalamus_membrane[:, t] >=
threshold).astype(float)
    thalamus_membrane[thalamus_spikes[:, t] == 1, t] = reset_potential


    # Motor cortex
    motor_input = np.dot(w_thalamus_motor, thalamus_spikes[:, t]) +
noise_amplitude * np.random.rand(num_motor_cortex)
    motor_membrane[:, t] = motor_membrane[:, t - 1] + dt / tau_mem *
(-motor_membrane[:, t - 1] + motor_input)
    motor_spikes[:, t] = (motor_membrane[:, t] >= threshold).astype(float)
    motor_membrane[motor_spikes[:, t] == 1, t] = reset_potential


    # Corticospinal tract
    corticospinal_input = np.dot(w_motor_corticospinal, motor_spikes[:, t]) +
noise_amplitude * np.random.rand(num_corticospinal)
    corticospinal_membrane[:, t] = corticospinal_membrane[:, t - 1] + dt /
tau_mem * (-corticospinal_membrane[:, t - 1] + corticospinal_input)
    corticospinal_spikes[:, t] = (corticospinal_membrane[:, t] >=
threshold).astype(float)
    corticospinal_membrane[corticospinal_spikes[:, t] == 1, t] =
reset_potential


# Plotting results as raster plots for each layer
plt.figure(figsize=(12, 8))


# Purkinje cells raster plot
plt.subplot(5, 1, 1)
plt.title("Purkinje Cells")
plt.imshow(purkinje_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")
```

```
# DCN raster plot
plt.subplot(5, 1, 2)
plt.title("Deep Cerebellar Nuclei (DCN)")
plt.imshow(dcn_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")


# Thalamus raster plot
plt.subplot(5, 1, 3)
plt.title("Thalamus")
plt.imshow(thalamus_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")


# Motor cortex raster plot
plt.subplot(5, 1, 4)
plt.title("Motor Cortex")
plt.imshow(motor_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")


# Corticospinal tract raster plot
plt.subplot(5, 1, 5)
plt.title("Corticospinal Tract")
plt.imshow(corticospinal_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")
plt.xlabel("Time Steps")


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

[2]

Python

```
#Initialization and Parameters
import numpy as np
import matplotlib.pyplot as plt


# Simulation parameters
time_steps = 1000  # Total simulation time steps
dt = 0.1  # Time step size (ms)
num_purkinje = 10
num_dcn = 5
num_thalamus = 5
num_motor_cortex = 10
num_corticospinal = 20


# Adjusted Synaptic Weights
w_purkinje_dcn = -2.0 * np.ones((num_dcn, num_purkinje))             # Stronger
inhibition from Purkinje to DCN
w_dcn_thalamus = 2.0 * np.ones((num_thalamus, num_dcn))              # Stronger
excitation from DCN to thalamus
w_thalamus_motor = 2.0 * np.ones((num_motor_cortex, num_thalamus))   # Stronger
excitation from thalamus to motor cortex
w_motor_corticospinal = 2.0 * np.ones((num_corticospinal, num_motor_cortex))  #
Stronger excitation from motor cortex to corticospinal tract


# Adjusted Neuron Model Parameters
tau_mem = 20.0    # Faster membrane dynamics
threshold = 1.0   # Keep threshold at 1.0
resting_potential = 0.0
reset_potential = 0.0


#Inputs and State Variables
# Adjusted Inputs
climbing_fiber_input = 5 * np.random.rand(num_purkinje, time_steps)  # Scale
climbing fiber input
mossy_fiber_input = 5 * np.random.rand(num_dcn, time_steps)          # Scale
mossy fiber input


# Membrane potentials
```

```
purkinje_membrane = np.zeros((num_purkinje, time_steps))
dcn_membrane = np.zeros((num_dcn, time_steps))
thalamus_membrane = np.zeros((num_thalamus, time_steps))
motor_membrane = np.zeros((num_motor_cortex, time_steps))
corticospinal_membrane = np.zeros((num_corticospinal, time_steps))


# Spike recorders
purkinje_spikes = np.zeros((num_purkinje, time_steps))
dcn_spikes = np.zeros((num_dcn, time_steps))
thalamus_spikes = np.zeros((num_thalamus, time_steps))
motor_spikes = np.zeros((num_motor_cortex, time_steps))
corticospinal_spikes = np.zeros((num_corticospinal, time_steps))


#Simulation Loop
for t in range(1, time_steps):
    # Purkinje cells
    purkinje_input = climbing_fiber_input[:, t]
    purkinje_membrane[:, t] = (
        purkinje_membrane[:, t - 1]
        + dt / tau_mem * (-purkinje_membrane[:, t - 1] + purkinje_input)
    )
    purkinje_spikes[:, t] = (purkinje_membrane[:, t] >=
threshold).astype(float)
    purkinje_membrane[purkinje_spikes[:, t] == 1, t] = reset_potential


    # Deep cerebellar nuclei (DCN)
    dcn_input = mossy_fiber_input[:, t] + np.dot(w_purkinje_dcn,
purkinje_spikes[:, t])
    dcn_membrane[:, t] = (
        dcn_membrane[:, t - 1]
        + dt / tau_mem * (-dcn_membrane[:, t - 1] + dcn_input)
    )
    dcn_spikes[:, t] = (dcn_membrane[:, t] >= threshold).astype(float)
    dcn_membrane[dcn_spikes[:, t] == 1, t] = reset_potential


    # Thalamus
    thalamus_input = np.dot(w_dcn_thalamus, dcn_spikes[:, t])
    thalamus_membrane[:, t] = (
        thalamus_membrane[:, t - 1]
        + dt / tau_mem * (-thalamus_membrane[:, t - 1] + thalamus_input)
```

```
    )
    thalamus_spikes[:, t] = (thalamus_membrane[:, t] >=
threshold).astype(float)
    thalamus_membrane[thalamus_spikes[:, t] == 1, t] = reset_potential


    # Motor cortex
    motor_input = np.dot(w_thalamus_motor, thalamus_spikes[:, t])
    motor_membrane[:, t] = (
        motor_membrane[:, t - 1]
        + dt / tau_mem * (-motor_membrane[:, t - 1] + motor_input)
    )
    motor_spikes[:, t] = (motor_membrane[:, t] >= threshold).astype(float)
    motor_membrane[motor_spikes[:, t] == 1, t] = reset_potential


    # Corticospinal tract
    corticospinal_input = np.dot(w_motor_corticospinal, motor_spikes[:, t])
    corticospinal_membrane[:, t] = (
        corticospinal_membrane[:, t - 1]
        + dt / tau_mem * (-corticospinal_membrane[:, t - 1] +
corticospinal_input)
    )
    corticospinal_spikes[:, t] = (corticospinal_membrane[:, t] >=
threshold).astype(float)
    corticospinal_membrane[corticospinal_spikes[:, t] == 1, t] =
reset_potential


#Visualization
plt.figure(figsize=(12, 8))


# Purkinje
plt.subplot(5, 1, 1)
plt.title("Purkinje Cells")
plt.imshow(purkinje_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")


# DCN
plt.subplot(5, 1, 2)
plt.title("Deep Cerebellar Nuclei (DCN)")
plt.imshow(dcn_spikes, aspect='auto', cmap='binary')
```

```
plt.ylabel("Neuron Index")


# Thalamus
plt.subplot(5, 1, 3)
plt.title("Thalamus")
plt.imshow(thalamus_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")


# Motor Cortex
plt.subplot(5, 1, 4)
plt.title("Motor Cortex")
plt.imshow(motor_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")


# Corticospinal
plt.subplot(5, 1, 5)
plt.title("Corticospinal Tract")
plt.imshow(corticospinal_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")
plt.xlabel("Time Steps")


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

[3]

Python

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


# Simulation parameters
time_steps = 1000  # Total simulation time steps
dt = 0.1  # Time step size (ms)
num_purkinje = 10  # Number of Purkinje cells
num_dcn = 5  # Number of deep cerebellar nuclei (DCN) neurons
num_thalamus = 5  # Number of thalamic neurons
num_motor_cortex = 10  # Number of motor cortex neurons
num_corticospinal = 20  # Number of corticospinal tract neurons


# Neuron model parameters
tau_mem = 10.0  # Membrane time constant (ms)
threshold = 1.0  # Spiking threshold
resting_potential = 0.0  # Resting membrane potential
reset_potential = 0.0  # Reset potential after spike


# Best configuration parameters (from optimization results)
w_purkinje_dcn_val = 2.3 # Inhibitory weight from Purkinje to DCN
w_dcn_thalamus_val = 2.1  # Excitatory weight from DCN to thalamus
w_thalamus_motor_val = 3 # Excitatory weight from thalamus to motor cortex
w_motor_corticospinal_val = 2.6 # Excitatory weight from motor cortex to
corticospinal tract
noise_amplitude_val = 2    # Noise amplitude for random inputs


# Synaptic weights
w_purkinje_dcn = -w_purkinje_dcn_val * np.ones((num_dcn, num_purkinje))
w_dcn_thalamus = w_dcn_thalamus_val * np.ones((num_thalamus, num_dcn))
w_thalamus_motor = w_thalamus_motor_val * np.ones((num_motor_cortex,
num_thalamus))
w_motor_corticospinal = w_motor_corticospinal_val * np.ones((num_corticospinal,
num_motor_cortex))


# Inputs: scaled to increase excitability
climbing_fiber_input = 10.0 * np.random.rand(num_purkinje, time_steps)
mossy_fiber_input = 10.0 * np.random.rand(num_dcn, time_steps)
```

```
# Neuron states: membrane potentials and spikes
purkinje_membrane = np.zeros((num_purkinje, time_steps))
dcn_membrane = np.zeros((num_dcn, time_steps))
thalamus_membrane = np.zeros((num_thalamus, time_steps))
motor_membrane = np.zeros((num_motor_cortex, time_steps))
corticospinal_membrane = np.zeros((num_corticospinal, time_steps))


purkinje_spikes = np.zeros((num_purkinje, time_steps))
dcn_spikes = np.zeros((num_dcn, time_steps))
thalamus_spikes = np.zeros((num_thalamus, time_steps))
motor_spikes = np.zeros((num_motor_cortex, time_steps))
corticospinal_spikes = np.zeros((num_corticospinal, time_steps))


# Simulation loop
for t in range(1, time_steps):
    # Purkinje cells
    purkinje_input = climbing_fiber_input[:, t]
    purkinje_membrane[:, t] = purkinje_membrane[:, t - 1] + dt / tau_mem *
(-purkinje_membrane[:, t - 1] + purkinje_input)
    purkinje_spikes[:, t] = (purkinje_membrane[:, t] >=
threshold).astype(float)
    purkinje_membrane[purkinje_spikes[:, t] == 1, t] = reset_potential


    # Deep cerebellar nuclei (DCN)
    dcn_input = mossy_fiber_input[:, t] + np.dot(w_purkinje_dcn,
purkinje_spikes[:, t])
    dcn_membrane[:, t] = dcn_membrane[:, t - 1] + dt / tau_mem *
(-dcn_membrane[:, t - 1] + dcn_input)
    dcn_spikes[:, t] = (dcn_membrane[:, t] >= threshold).astype(float)
    dcn_membrane[dcn_spikes[:, t] == 1, t] = reset_potential


    # Thalamus
    thalamus_input = np.dot(w_dcn_thalamus, dcn_spikes[:, t]) +
noise_amplitude_val * np.random.rand(num_thalamus)
    thalamus_membrane[:, t] = thalamus_membrane[:, t - 1] + dt / tau_mem *
(-thalamus_membrane[:, t - 1] + thalamus_input)
    thalamus_spikes[:, t] = (thalamus_membrane[:, t] >=
threshold).astype(float)
    thalamus_membrane[thalamus_spikes[:, t] == 1, t] = reset_potential
```

```
    # Motor cortex
    motor_input = np.dot(w_thalamus_motor, thalamus_spikes[:, t]) +
noise_amplitude_val * np.random.rand(num_motor_cortex)
    motor_membrane[:, t] = motor_membrane[:, t - 1] + dt / tau_mem *
(-motor_membrane[:, t - 1] + motor_input)
    motor_spikes[:, t] = (motor_membrane[:, t] >= threshold).astype(float)
    motor_membrane[motor_spikes[:, t] == 1, t] = reset_potential


    # Corticospinal tract
    corticospinal_input = np.dot(w_motor_corticospinal, motor_spikes[:, t]) +
noise_amplitude_val * np.random.rand(num_corticospinal)
    corticospinal_membrane[:, t] = corticospinal_membrane[:, t - 1] + dt /
tau_mem * (-corticospinal_membrane[:, t - 1] + corticospinal_input)
    corticospinal_spikes[:, t] = (corticospinal_membrane[:, t] >=
threshold).astype(float)
    corticospinal_membrane[corticospinal_spikes[:, t] == 1, t] =
reset_potential


# Plotting results as raster plots using imshow()
plt.figure(figsize=(12, 8))


plt.subplot(5, 1, 1)
plt.title("Purkinje Cells")
plt.imshow(purkinje_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")


plt.subplot(5, 1, 2)
plt.title("Deep Cerebellar Nuclei (DCN)")
plt.imshow(dcn_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")


plt.subplot(5, 1, 3)
plt.title("Thalamus")
plt.imshow(thalamus_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")
```

```
plt.subplot(5, 1, 4)
plt.title("Motor Cortex")
plt.imshow(motor_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")


plt.subplot(5, 1, 5)
plt.title("Corticospinal Tract")
plt.imshow(corticospinal_spikes, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")
plt.xlabel("Time Steps")


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

[4]

Python

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

def simulate_model(
    w_purkinje_dcn_val,
    w_dcn_thalamus_val,
    w_thalamus_motor_val,
    w_motor_corticospinal_val,
    # Plasticity parameters
    alpha_ltp=0.0005,
    alpha_ltd=0.001,
    # Feedback weight scale
    w_motor_purkinje_val=0.01
):
    """
    A simplified integrate-and-fire model of cerebellar, thalamic,
    and cortical circuits with added plasticity and a motor-to-cerebellum
feedback loop.
    """
    time_steps = 2000
    dt = 0.1

    # Neuron group sizes
    num_purkinje = 10
    num_dcn = 5
    num_thalamus = 5
    num_motor_cortex = 10
    num_corticospinal = 20

    # Membrane constants
    tau_mem = 20.0
    threshold = 1.0
    reset_potential = -0.2

    # Synaptic weights (initial)
    w_purkinje_dcn = -np.ones((num_dcn, num_purkinje)) * w_purkinje_dcn_val
    w_dcn_thalamus = np.ones((num_thalamus, num_dcn)) * w_dcn_thalamus_val
    w_thalamus_motor = np.ones((num_motor_cortex, num_thalamus)) *
w_thalamus_motor_val
    w_motor_corticospinal = np.ones((num_corticospinal, num_motor_cortex)) *
w_motor_corticospinal_val
```

```
    # Feedback loop from motor cortex back to Purkinje cells
    w_motor_purkinje = np.ones((num_purkinje, num_motor_cortex)) *
w_motor_purkinje_val

    # Random external inputs
    climbing_fiber_input = 8.0 * np.random.rand(num_purkinje, time_steps)
    mossy_fiber_input = 8.0 * np.random.rand(num_dcn, time_steps)

    # Membrane potentials
    purkinje_membrane = np.zeros((num_purkinje, time_steps))
    dcn_membrane = np.zeros((num_dcn, time_steps))
    thalamus_membrane = np.zeros((num_thalamus, time_steps))
    motor_membrane = np.zeros((num_motor_cortex, time_steps))
    corticospinal_membrane = np.zeros((num_corticospinal, time_steps))

    # Spike arrays
    purkinje_spikes = np.zeros((num_purkinje, time_steps))
    dcn_spikes = np.zeros((num_dcn, time_steps))
    thalamus_spikes = np.zeros((num_thalamus, time_steps))
    motor_spikes = np.zeros((num_motor_cortex, time_steps))
    corticospinal_spikes = np.zeros((num_corticospinal, time_steps))

    # Simple placeholder for a more detailed neuron model
    # For Hodgkin-Huxley, integrate gating variables, conductances, etc.

    for t in range(1, time_steps):
        #-----------------------------
        # Purkinje Cells
        #-----------------------------
        # Add baseline and climbing fiber input plus feedback from the motor
cortex
        purkinje_input = climbing_fiber_input[:, t] + 2.0 \
                         + np.dot(w_motor_purkinje, motor_spikes[:, t-1])
        purkinje_membrane[:, t] = purkinje_membrane[:, t-1] + dt/tau_mem * (
            -purkinje_membrane[:, t-1] + purkinje_input
        )
        purkinje_spikes[:, t] = (purkinje_membrane[:, t] >=
threshold).astype(float)
        purkinje_membrane[purkinje_spikes[:, t] == 1, t] = reset_potential

        #-----------------------------
        # Deep Cerebellar Nuclei (DCN)
        #-----------------------------
        dcn_input = mossy_fiber_input[:, t] \
```

```
                    + np.dot(w_purkinje_dcn, purkinje_spikes[:, t]) \
                    + 1.5
        dcn_membrane[:, t] = dcn_membrane[:, t-1] + dt/tau_mem * (
            -dcn_membrane[:, t-1] + dcn_input
        )
        dcn_spikes[:, t] = (dcn_membrane[:, t] >= threshold).astype(float)
        dcn_membrane[dcn_spikes[:, t] == 1, t] = reset_potential

        #----------------------------
        # Thalamus
        #----------------------------
        thalamus_input = np.dot(w_dcn_thalamus, dcn_spikes[:, t]) + 1.0
        thalamus_membrane[:, t] = thalamus_membrane[:, t-1] + dt/tau_mem * (
            -thalamus_membrane[:, t-1] + thalamus_input
        )
        thalamus_spikes[:, t] = (thalamus_membrane[:, t] >=
threshold).astype(float)
        thalamus_membrane[thalamus_spikes[:, t] == 1, t] = reset_potential

        #----------------------------
        # Motor Cortex
        #----------------------------
        motor_input = np.dot(w_thalamus_motor, thalamus_spikes[:, t]) + 1.0
        motor_membrane[:, t] = motor_membrane[:, t-1] + dt/tau_mem * (
            -motor_membrane[:, t-1] + motor_input
        )
        motor_spikes[:, t] = (motor_membrane[:, t] >= threshold).astype(float)
        motor_membrane[motor_spikes[:, t] == 1, t] = reset_potential

        #----------------------------
        # Corticospinal Tract
        #----------------------------
        corticospinal_input = np.dot(w_motor_corticospinal, motor_spikes[:, t])
+ 0.5
        corticospinal_membrane[:, t] = corticospinal_membrane[:, t-1] +
dt/tau_mem * (
            -corticospinal_membrane[:, t-1] + corticospinal_input
        )
        corticospinal_spikes[:, t] = (corticospinal_membrane[:, t] >=
threshold).astype(float)
        corticospinal_membrane[corticospinal_spikes[:, t] == 1, t] =
reset_potential

        #----------------------------
```

```
        # Simple Synaptic Plasticity Example (LTD/LTP)
        #----------------------------
        # If a Purkinje cell fires while the climbing fiber input is high,
reduce (LTD)
        # If a Purkinje cell fires with low climbing fiber input, increase
(LTP)
        high_climbing = (climbing_fiber_input[:, t] > 5.0).astype(float)
        low_climbing = (climbing_fiber_input[:, t] < 2.0).astype(float)

        for j in range(num_dcn):
            for i in range(num_purkinje):
                if purkinje_spikes[i, t] == 1.0:
                    if high_climbing[i] == 1.0:
                        # LTD - decrease magnitude of inhibition
                        w_purkinje_dcn[j, i] -= alpha_ltd * w_purkinje_dcn_val
                    elif low_climbing[i] == 1.0:
                        # LTP - make inhibition slightly stronger (more
negative)
                        w_purkinje_dcn[j, i] -= alpha_ltp * w_purkinje_dcn_val

    return (
        purkinje_spikes, dcn_spikes, thalamus_spikes,
        motor_spikes, corticospinal_spikes,
        purkinje_membrane, dcn_membrane,
        thalamus_membrane, motor_membrane,
        corticospinal_membrane, threshold
    )


# Example usage with adjusted parameters
(
    p_spk, d_spk, th_spk, m_spk,
    c_spk, p_mem, d_mem, th_mem,
    mot_mem, cort_mem, threshold
) = simulate_model(
    w_purkinje_dcn_val=0.5,
    w_dcn_thalamus_val=0.8,
    w_thalamus_motor_val=0.8,
    w_motor_corticospinal_val=1.0,
    alpha_ltp=0.0005,
    alpha_ltd=0.001,
    w_motor_purkinje_val=0.01
)
```

```
# Plot raster plots for spikes across layers
plt.figure(figsize=(12, 8))

plt.subplot(5, 1, 1)
plt.title("Purkinje Cells")
plt.imshow(p_spk, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")

plt.subplot(5, 1, 2)
plt.title("Deep Cerebellar Nuclei (DCN)")
plt.imshow(d_spk, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")

plt.subplot(5, 1, 3)
plt.title("Thalamus")
plt.imshow(th_spk, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")

plt.subplot(5, 1, 4)
plt.title("Motor Cortex")
plt.imshow(m_spk, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")

plt.subplot(5, 1, 5)
plt.title("Corticospinal Tract")
plt.imshow(c_spk, aspect='auto', cmap='binary')
plt.ylabel("Neuron Index")
plt.xlabel("Time Steps")

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

[5]

Python

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

# ========================
# Neuron Model
# ========================
def lif_neuron(mem_prev, input_current, dt, tau_mem, threshold,
reset_potential):
    """Leaky Integrate-and-Fire neuron model"""
    new_mem = mem_prev + dt/tau_mem * (-mem_prev + input_current)
    spike = (new_mem >= threshold).astype(float)
    new_mem = np.where(spike, reset_potential, new_mem)
    return new_mem, spike

# ========================
# STDP Plasticity
# ========================
class STDP:
    def __init__(self, pre_dim, post_dim, tau_plus=20.0, tau_minus=20.0,
a_plus=0.1, a_minus=0.1):
        self.tau_plus = tau_plus
        self.tau_minus = tau_minus
        self.a_plus = a_plus
        self.a_minus = a_minus
        self.tr_pre = np.zeros(pre_dim)   # Trace for presynaptic neurons
        self.tr_post = np.zeros(post_dim) # Trace for postsynaptic neurons

    def update(self, pre_spike, post_spike, dt):
        # Update traces
        self.tr_pre = self.tr_pre * (1 - dt/self.tau_plus) + pre_spike
        self.tr_post = self.tr_post * (1 - dt/self.tau_minus) + post_spike

        # Weight update matrix (post_dim x pre_dim)
        delta_w = (self.a_plus * np.outer(post_spike, self.tr_pre) -
                   self.a_minus * np.outer(self.tr_post, pre_spike))
        return delta_w

# ========================
# Simulation Core
# ========================
def simulate_model():
```

```
time_steps = 1000
dt = 0.1
num_purkinje = 10
num_dcn = 5
num_thalamus = 5
num_motor = 10
num_corticospinal = 20

# Neuron parameters
tau_mem = 5.0
threshold = 1
reset_potential = 0.0

# Initialize STDP
stdp = STDP(pre_dim=num_purkinje, post_dim=num_dcn)

# Inputs
climbing_fiber = 8 * np.random.rand(num_purkinje, time_steps)
mossy_fiber = 8 * np.random.rand(num_dcn, time_steps)

# Initialize membranes and spikes (2D arrays)
purk_mem = np.zeros((num_purkinje, time_steps))
dcn_mem = np.zeros((num_dcn, time_steps))
thal_mem = np.zeros((num_thalamus, time_steps))
motor_mem = np.zeros((num_motor, time_steps))
cortico_mem = np.zeros((num_corticospinal, time_steps))

purk_spk = np.zeros((num_purkinje, time_steps))
dcn_spk = np.zeros((num_dcn, time_steps))
thal_spk = np.zeros((num_thalamus, time_steps))
motor_spk = np.zeros((num_motor, time_steps))
cortico_spk = np.zeros((num_corticospinal, time_steps))

# Synaptic weights
w_purkinje_dcn = -1.2 * np.ones((num_dcn, num_purkinje))
w_dcn_thalamus = 6.0 * np.ones((num_thalamus, num_dcn))
w_thalamus_motor = 6.0 * np.ones((num_motor, num_thalamus))
w_motor_cortico = 6.0 * np.ones((num_corticospinal, num_motor))
w_motor_dcn = 0.5 * np.ones((num_dcn, num_motor))

motor_spk_prev = np.zeros(num_motor)

for t in range(1, time_steps):
    # Purkinje cells
```

```
        purk_mem[:, t], purk_spk[:, t] = lif_neuron(
            purk_mem[:, t-1],
            climbing_fiber[:, t],
            dt, tau_mem, threshold, reset_potential
        )

        # DCN with feedback
        feedback = np.dot(w_motor_dcn, motor_spk_prev)
        dcn_input = mossy_fiber[:, t] + np.dot(w_purkinje_dcn, purk_spk[:, t])
+ feedback
        dcn_mem[:, t], dcn_spk[:, t] = lif_neuron(
            dcn_mem[:, t-1],
            dcn_input,
            dt, tau_mem, threshold, reset_potential
        )

        # STDP weight update
        delta_w = stdp.update(purk_spk[:, t], dcn_spk[:, t], dt)
        w_purkinje_dcn += delta_w

        # Thalamus
        thal_input = np.dot(w_dcn_thalamus, dcn_spk[:, t])
        thal_mem[:, t], thal_spk[:, t] = lif_neuron(
            thal_mem[:, t-1],
            thal_input,
            dt, tau_mem, threshold, reset_potential
        )

        # Motor cortex
        motor_input = np.dot(w_thalamus_motor, thal_spk[:, t])
        motor_mem[:, t], motor_spk[:, t] = lif_neuron(
            motor_mem[:, t-1],
            motor_input,
            dt, tau_mem, threshold, reset_potential
        )
        motor_spk_prev = motor_spk[:, t].copy()

        # Corticospinal
        cortico_input = np.dot(w_motor_cortico, motor_spk[:, t])
        cortico_mem[:, t], cortico_spk[:, t] = lif_neuron(
            cortico_mem[:, t-1],
            cortico_input,
            dt, tau_mem, threshold, reset_potential
        )
```

```
    return purk_spk, dcn_spk, thal_spk, motor_spk, cortico_spk

# ========================
# Plotting
# ========================
p_spk, d_spk, th_spk, m_spk, c_spk = simulate_model()

plt.figure(figsize=(12, 10))
layers = [
    ("Purkinje Cells", p_spk),
    ("Deep Cerebellar Nuclei", d_spk),
    ("Thalamus", th_spk),
    ("Motor Cortex", m_spk),
    ("Corticospinal Tract", c_spk)
]

for idx, (name, spk_data) in enumerate(layers):
    plt.subplot(5, 1, idx+1)
    plt.title(name)
    plt.imshow(spk_data, aspect='auto', cmap='binary', interpolation='none')
    plt.ylabel("Neuron Index")
    if idx == 4:
        plt.xlabel("Time Steps")

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