DSP Lab Demo 52 Exercise 1

Major Code Changes

1. Set up matplotlib animation with two subplots

```
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(10, 6))
line1, = ax1.plot(x_axis, input_buffer, 'b-', linewidth=0.5)
line2, = ax2.plot(x_axis, output_buffer, 'r-', linewidth=0.5)

ani = animation.FuncAnimation(
    fig,
    animate,
    interval=int(1000 * BLOCKLEN / RATE),
    blit=True,
    cache_frame_data=False)
```

2. Implemented block-based filter

```
BLOCKLEN = 512

for n in range(0, BLOCKLEN):
    x = float(input_tuple[n])

    # canonical form filter
    acc = x
    for k in range(1, M + 1):
        acc -= a[k] * w[k]
    w0 = acc

    y = b[0] * w0
    for k in range(1, M + 1):
        y += b[k] * w[k]

    for k in range(M, 0, -1):
        w[k] = w[k - 1] if k - 1 >= 1 else w0
```

3. Rolling display buffers

```
DISPLAY_SAMPLES = 2048

# shift old samples left, add new samples on right
input_buffer[:-BLOCKLEN] = input_buffer[BLOCKLEN:]
input_buffer[-BLOCKLEN:] = input_tuple

output_buffer[:-BLOCKLEN] = output_buffer[BLOCKLEN:]
output_buffer[-BLOCKLEN:] = output_block
```

Outcome

Implemented a real-time bandpass filter with animated visualization. The top subplot displays the input signal, the bottom subplot shows the filtered output. The animation updates at approximately 512 samples / 44100 Hz ≈ 11.6 ms per frame, synchronized with audio playback. The filter uses a canonical form structure requiring only M delay states instead of 2M, reducing memory usage.