

w03c — Signal and power: solution

$$f = [1, 3, 4, 0, 3, 4, 4, 4], \quad k = [1, 0, -1].$$

Since $k = [1, 0, -1]$, convolution gives $(k * f)[n] = f[n] - f[n - 2]$:

n	0	1	2	3	4	5	6	7	8	9
$(k * f)[n]$	1	3	3	-3	-1	4	1	0	-4	-4

Power.

$$\sum f^2 = 1 + 9 + 16 + 0 + 9 + 16 + 16 + 16 = \boxed{83}$$

$$\sum (k * f)^2 = 1 + 9 + 9 + 9 + 1 + 16 + 1 + 0 + 16 + 16 = \boxed{78}$$

$$78 \neq 83 \implies \text{power is not preserved.}$$

Why. By Parseval, filtering scales each frequency component by $|K(\omega)|$, so power is preserved only if $|K(\omega)| = 1$ everywhere (an all-pass filter). Here $K(\omega) = 1 - e^{-2i\omega}$, which is **zero at** $\omega = 0$: the filter is a discrete derivative and annihilates the DC (mean) component entirely. Since f has a large mean, a lot of power is removed.

A complete answer (i) computes both powers correctly and (ii) attributes the loss to the filter removing low frequencies. Saying only “no, because the filter is a high-pass/derivative filter” captures the intuition but skips the arithmetic that pins it down.

One watch-out: truncating the convolution to length 8 instead of 10 changes the number. Keeping only the first eight outputs gives $\sum = 46$; numpy’s `same` mode keeps the central eight and gives $\sum = 61$. Both are still well below 83, so the conclusion holds either way; what matters is the reasoning.