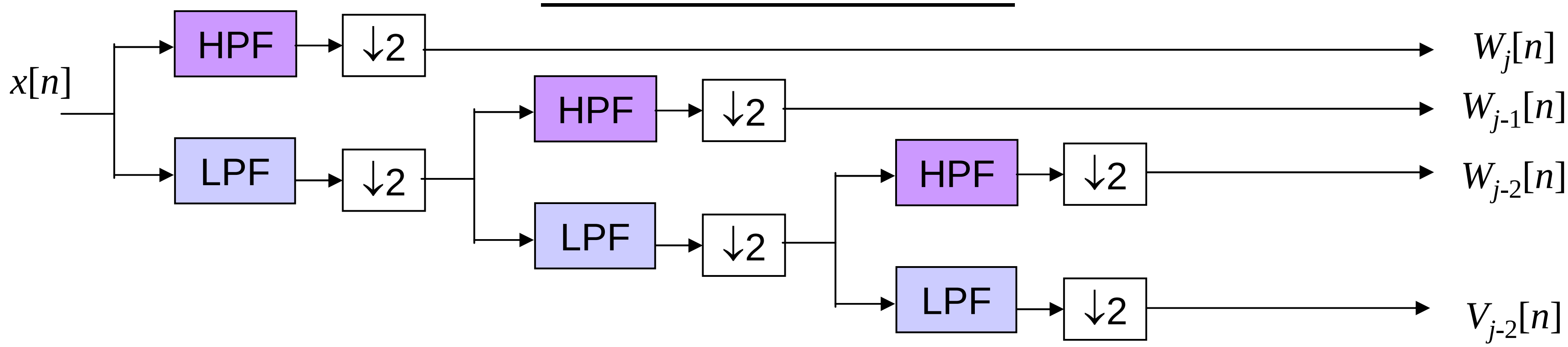
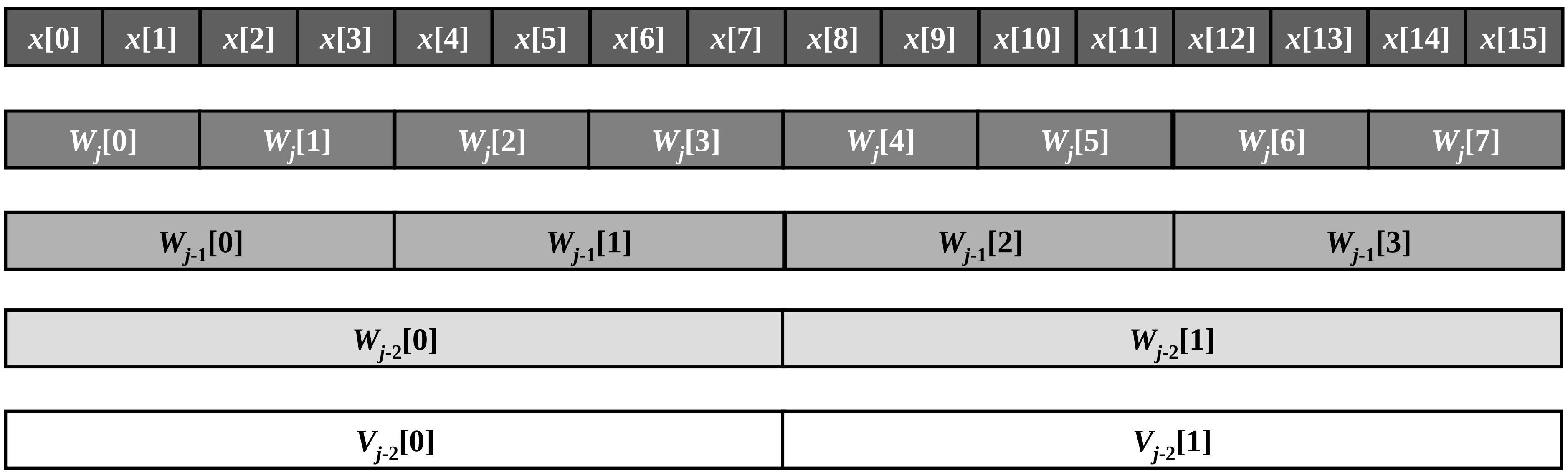
WT of Images

These use some material from:

1. J. S. Walker and T. Q. Nguyen, “Wavelet-Based Image Compression,” Ch. 6 in *The Transform and Data Compression Handbook*, edited by K. R. Rao and P. C. Yip, CRC Press, 2001.
2. X. Wu, “Compression of Wavelet Transform Coefficients,” Ch. 8 in *The Transform and Data Compression Handbook*, edited by K. R. Rao and P. C. Yip, CRC Press, 2001.

Recall 1-D WT





This shows the true time "duration" covered by each coefficient
The widening duration of lower lines is due to the decimation

Recall 1-D WT (cont.)

But… if we ignore the true “duration” coverage and just look at the coefficients as a series of numbers we can get the following

$x[0]$	$x[1]$	$x[2]$	$x[3]$	$x[4]$	$x[5]$	$x[6]$	$x[7]$	$x[8]$	$x[9]$	$x[10]$	$x[11]$	$x[12]$	$x[13]$	$x[14]$	$x[15]$

$W_j[0]$	$W_j[1]$	$W_j[2]$	$W_j[3]$	$W_j[4]$	$W_j[5]$	$W_j[6]$	$W_j[7]$

$W_{j-1}[0]$	$W_{j-1}[1]$	$W_{j-1}[2]$	$W_{j-1}[3]$

$W_{j-2}[0]$	$W_{j-2}[1]$

$V_{j-2}[0]$	$V_{j-2}[1]$

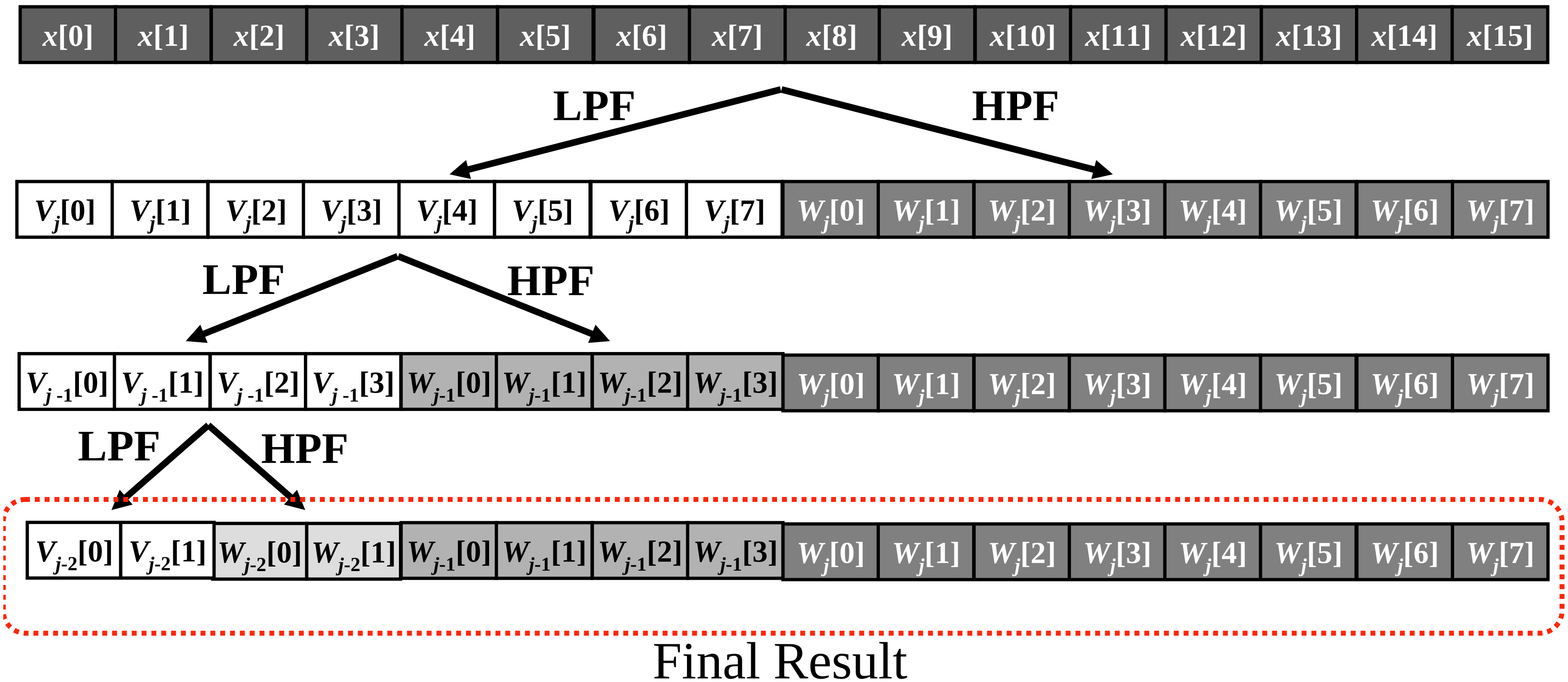
Which can be re-ordered as:

$V_{j-2}[0]$	$V_{j-2}[1]$	$W_{j-2}[0]$	$W_{j-2}[1]$	$W_{j-1}[0]$	$W_{j-1}[1]$	$W_{j-1}[2]$	$W_{j-1}[3]$	$W_j[0]$	$W_j[1]$	$W_j[2]$	$W_j[3]$	$W_j[4]$	$W_j[5]$	$W_j[6]$	$W_j[7]$

Recall: these are computed from the LPF output… so..

Recall 1-D WT (cont.)

So… another view is:



Recall: at each stage the LPF gives the next-lower resolution approximation… which then gets split further into into the next-lower resolution and the next level's details…. etc. etc. etc.

2-D WT

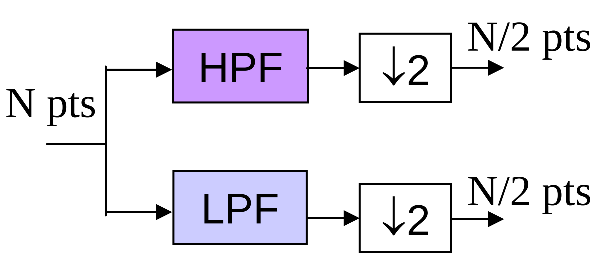
There are general 2-D WT methods but the most commonly used methods are those that use so-called “separable” 2-D filters….

…. That means that we

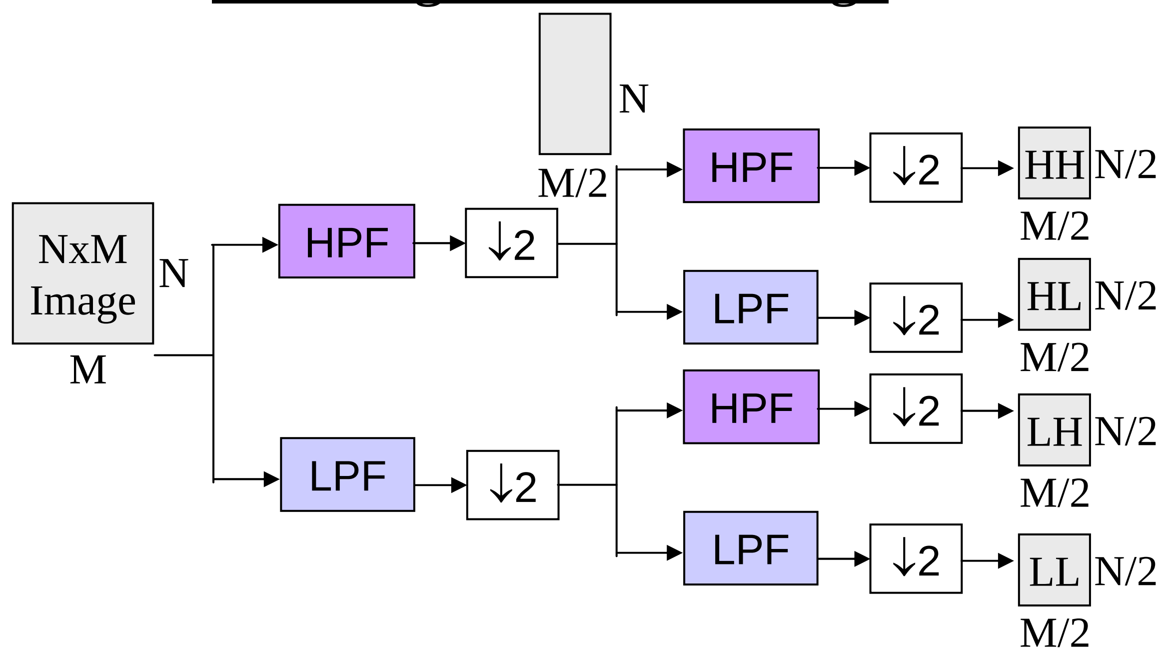
- First apply our filters to each row of the image
- Then apply our filters to each column of the row-filtered result

The result is that each subband stage of the 1-D filter bank gets replaced by what appears to be two stages, but is really one row-stage and one column-stage (which together back up one subband stage):

1-D Single Subband Stage



2-D Single Subband Stage



Third Stage
Subband Results

Second Stage
Subband Results

First Stage
Subband Results

Original Image

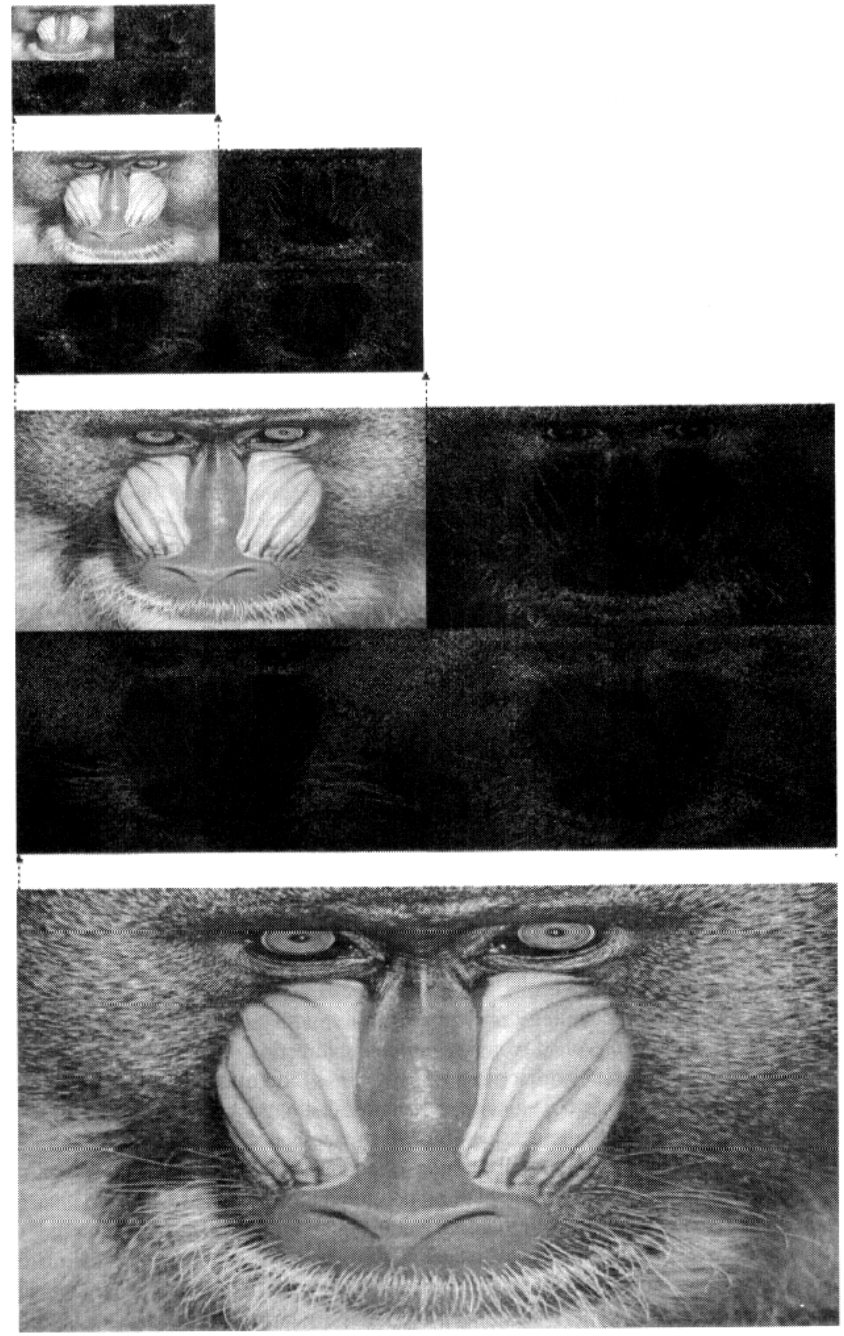


FIGURE 8.2
Dyadic wavelet decomposition of a test image.

(Figure from [2])

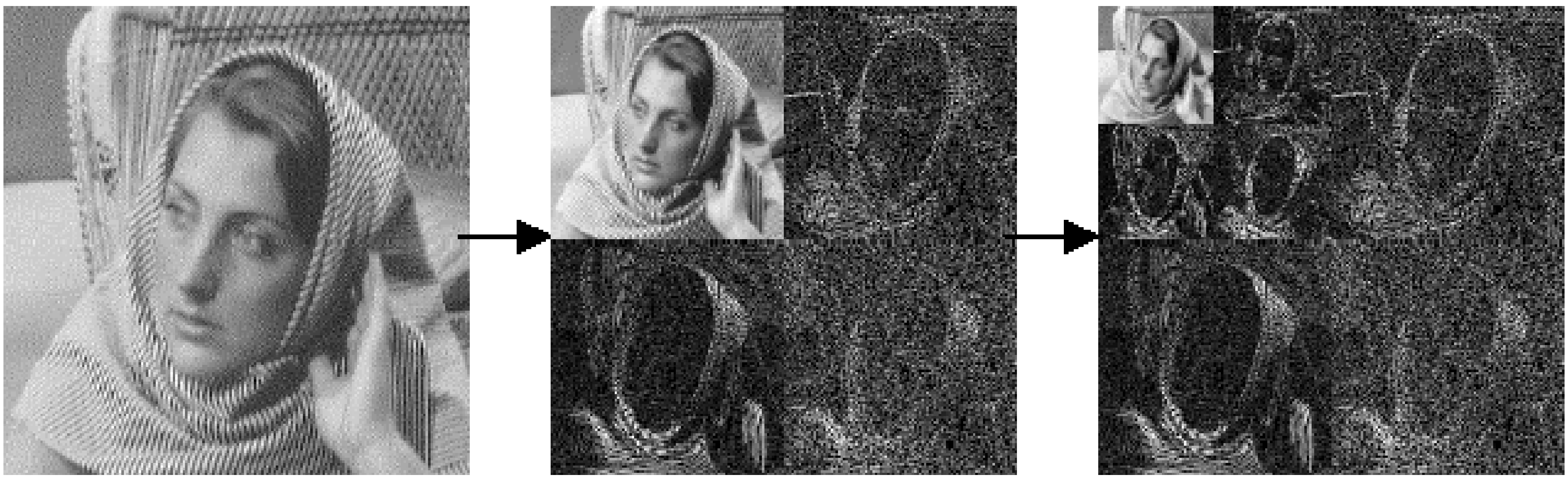


Fig. 5. Example of dyadic decomposition into subbands for the test image 'barbara'

(Figure from [1])

Choice of Wavelet for Image Compression

Wavelet reconstruction formula w/o quantization:

$$x(t) = \sum_{k=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} d_{j,k} 2^{j/2} \psi(2^j t - k)$$

$$= \sum_{j=j_0}^{\infty} \sum_{k=-\infty}^{\infty} d_{j,k} 2^{j/2} \psi(2^j t - k) + \sum_{k=-\infty}^{\infty} c_{j_0,k} 2^{j_0/2} \phi(2^{j_0} t - k)$$

Now consider <u>with</u> quantization:

$$\hat{x}(t) = \sum_{j=j_0}^{\infty} \sum_{k=-\infty}^{\infty} \left[d_{j,k} + \varepsilon_{j,k}^{d} \right] 2^{j/2} \psi(2^j t - k) + \sum_{k=-\infty}^{\infty} \left[c_{j_0,k} + \varepsilon_{j_0,k}^{c} \right] 2^{j_0/2} \phi(2^{j_0} t - k)$$

$$= x(t) + \underbrace{\sum_{j=j_0}^{\infty} \sum_{k=-\infty}^{\infty} \varepsilon_{j,k}^{d} 2^{j/2} \psi(2^j t - k) + \sum_{k=-\infty}^{\infty} \varepsilon_{j_0,k}^{c} 2^{j_0/2} \phi(2^{j_0} t - k)}_{\text{error term is sum of wavelets and scaling function}}$$

- Thus, if the wavelet and scaling functions are rough, then the error is rough. So we want to make them smooth
  - There are various results that show how to design wavelet systems with specific degrees of smoothness: see the wavelet literature for details
  - One such means is: $\int t^k \psi(t) dt = 0 \quad \text{for} \quad 0 \leq k \leq N$
  - This is called imposing *N* vanishing moments and imposes that the wavelet will be *N*-times continuously differentiable
- Another aspect of vanishing moments:
  - If a wavelet system has *N* vanishing moments, then polynomials of degree less than *N* can be represented as a linear combination of translates of the scaling function
  - Thus, any locally-polynomial component of an image having degree less than *N* gets zeroed out by the high-pass filter because it can be completely handled by the low-pass filter
  - This results in lots of zero values for the wavelet coefficients, which leads to efficient coding (via zerotrees, as we will see).