# Deep Learning

## Chapter 5: Mathematical Prespective of CNNs

**Mina Rezaei**

Department of Statistics – LMU Munich

Winter Semester 2020

# Convolutions — mathematical perspective

# CONVOLUTIONS : A DEEPER LOOK

- CNNs borrow their name from a mathematical operation termed **convolution** that originates in Signal Processing.
- Basic understanding of this concept and related operations improves the understanding of the CNN functionality.
- Still, there are successful practitioners that never heard of these concepts.
- The following should provide exactly this fundamental understanding of convolutions.

# CONVOLUTIONS : A DEEPER LOOK

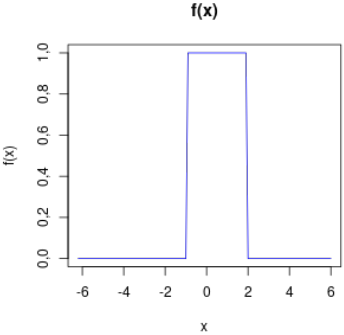
- Definition:

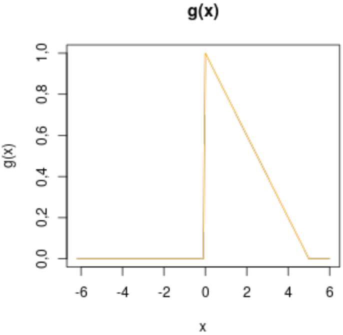
$$h(i) = (f * g)(i) = \int f(x)g(i - x)dx$$

$$\begin{aligned} \text{where } f(x) &: \text{input function} \\ \text{and } g(x) &: \text{weighting function, kernel} \\ \text{and } h(i) &: \text{output function, feature map elements} \end{aligned}$$

- Intuition 1: weighted smoothing of $f(x)$ with weighting function $g(x)$.
- Intuition 2: filter function $g(x)$ filters features $h(i)$ from input signal $f(x)$.

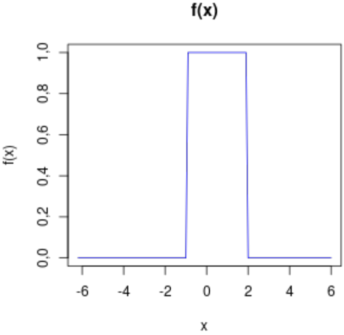
# 1D CONVOLUTION ANIMATION

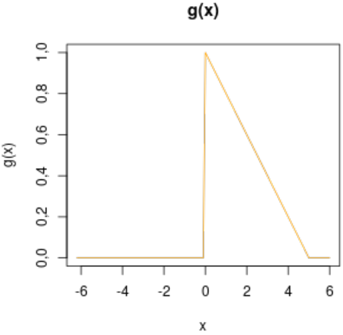



$$f(x) = \begin{cases} 1, & \text{if } x \in [-1, 2] \\ 0, & \text{otherwise} \end{cases} \qquad g(x) = \begin{cases} 1 - 0.2 * |x|, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases}$$

# 1D CONVOLUTION ANIMATION

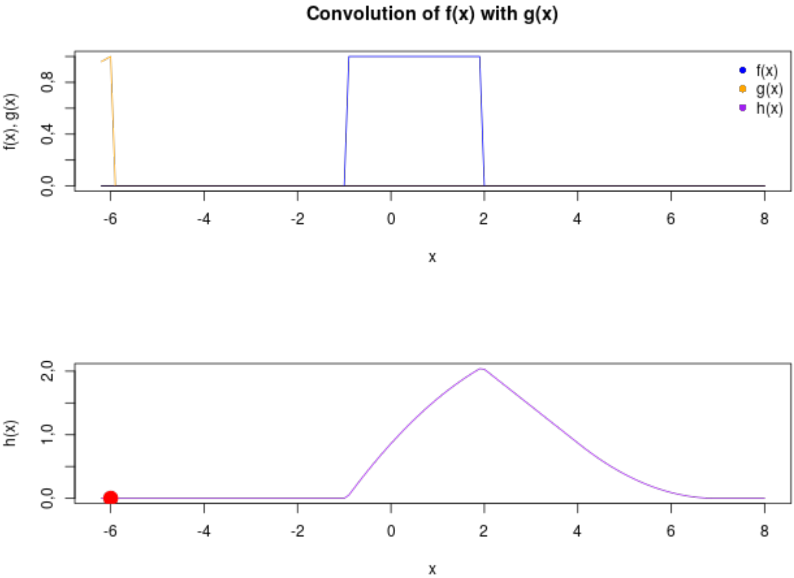


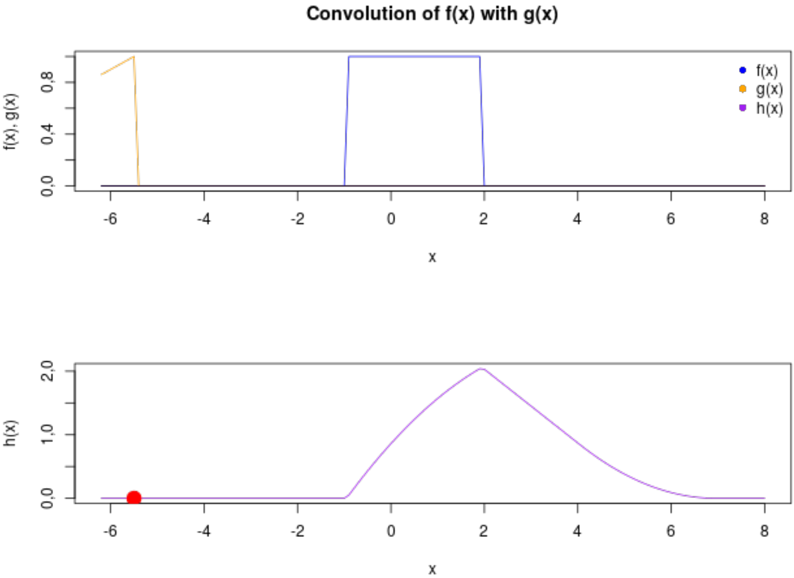

Kernel is flipped due to the negative iterator in

$$h(i) = \int_{x=-\infty}^{\infty} f(x)g(i-x)$$

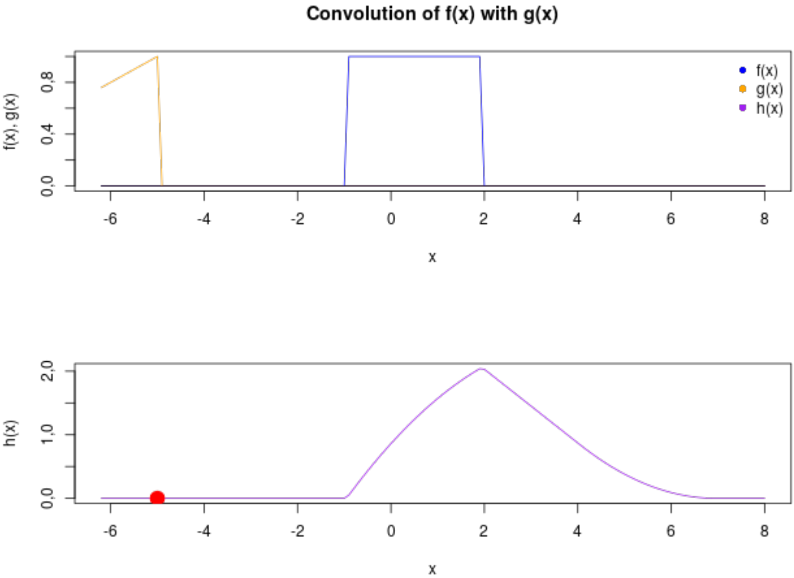
# 1D CONVOLUTION ANIMATION



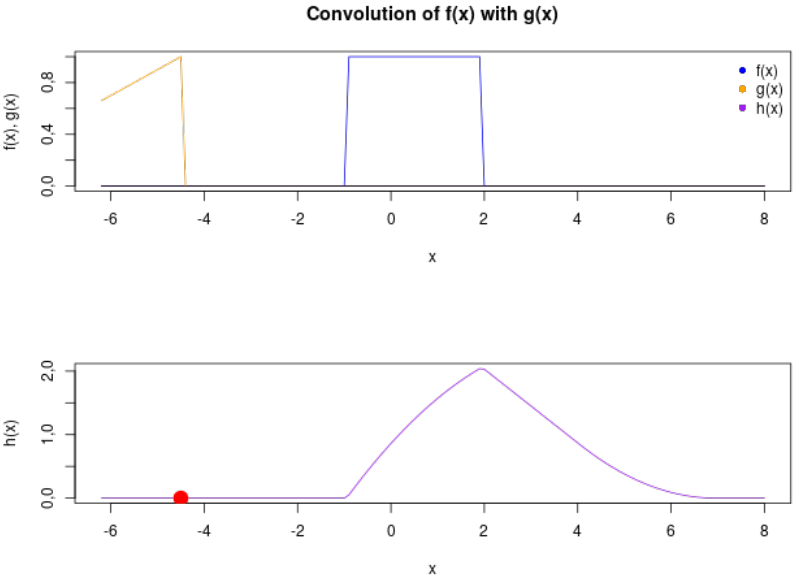
# 1D CONVOLUTION ANIMATION



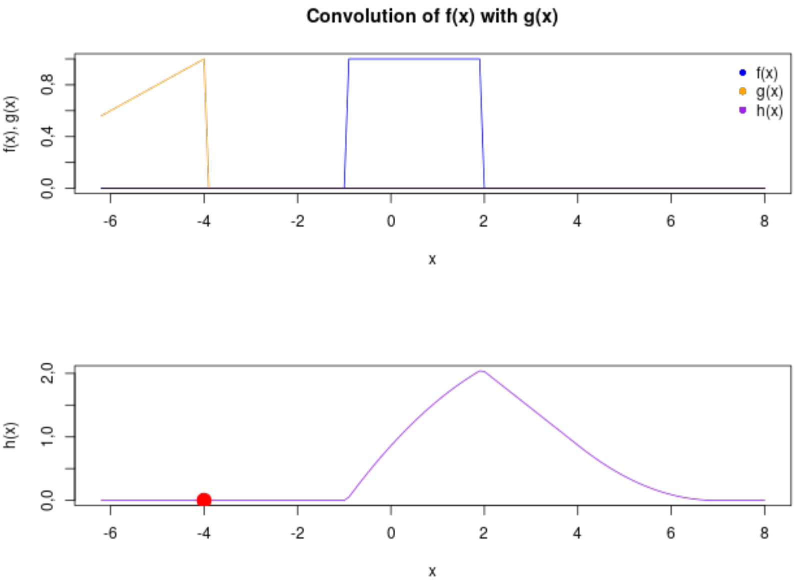
# 1D CONVOLUTION ANIMATION



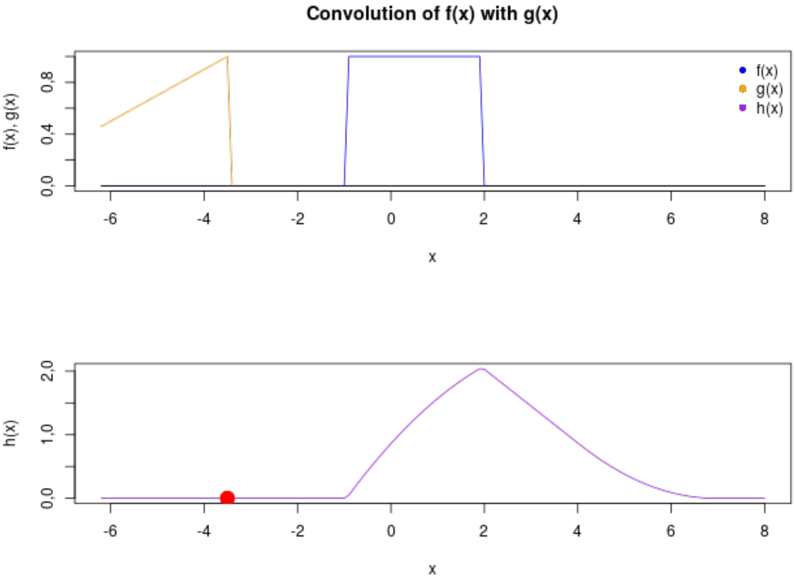
# 1D CONVOLUTION ANIMATION



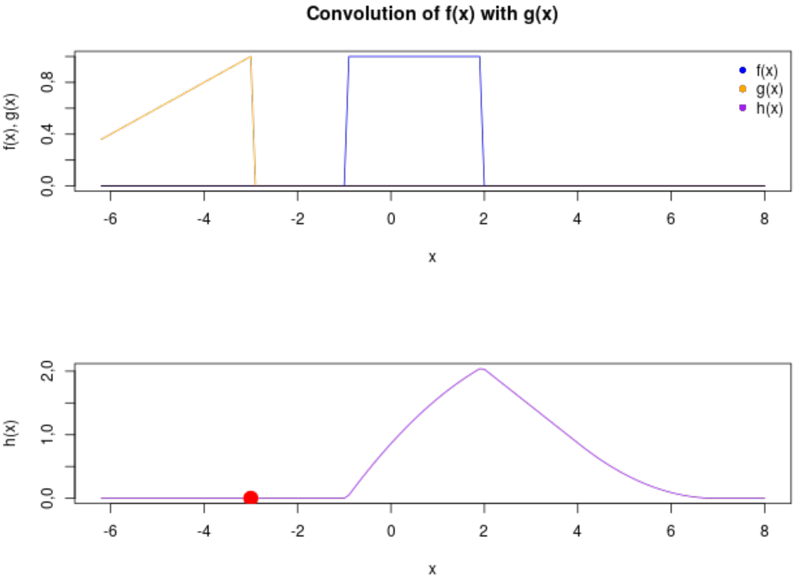
# 1D CONVOLUTION ANIMATION



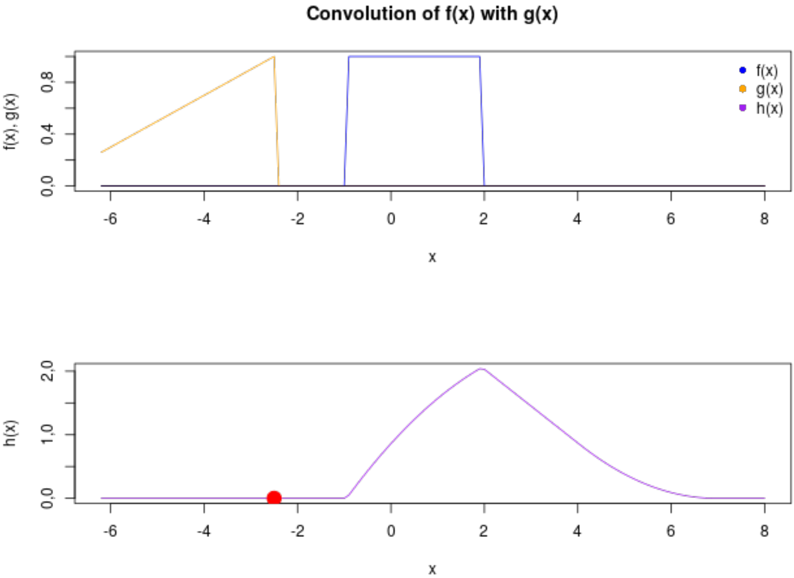
# 1D CONVOLUTION ANIMATION



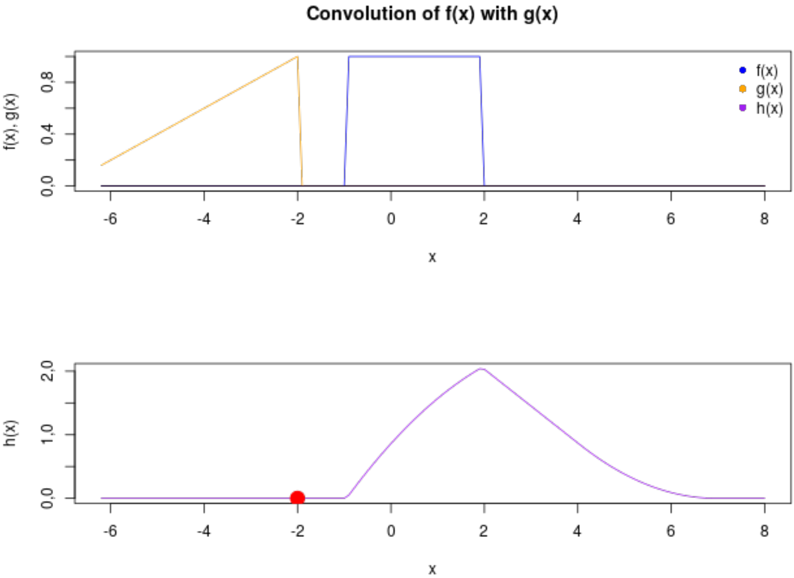
# 1D CONVOLUTION ANIMATION



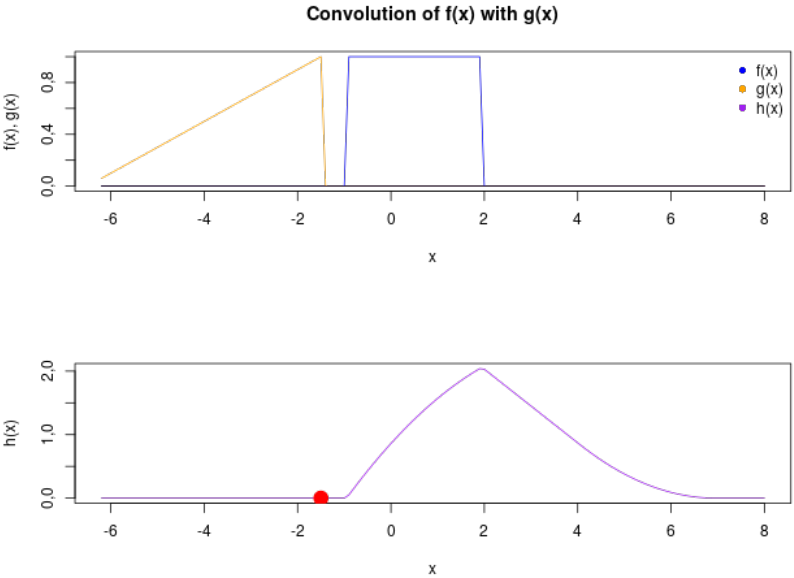
# 1D CONVOLUTION ANIMATION



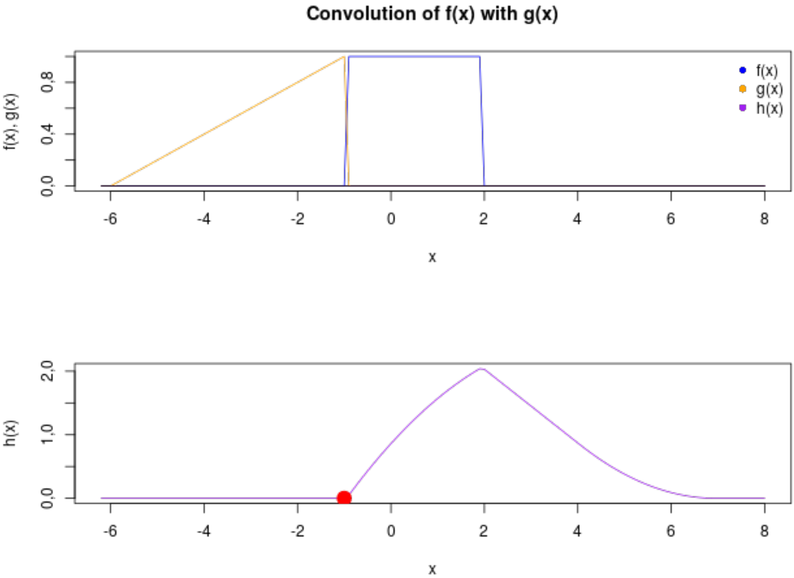
# 1D CONVOLUTION ANIMATION



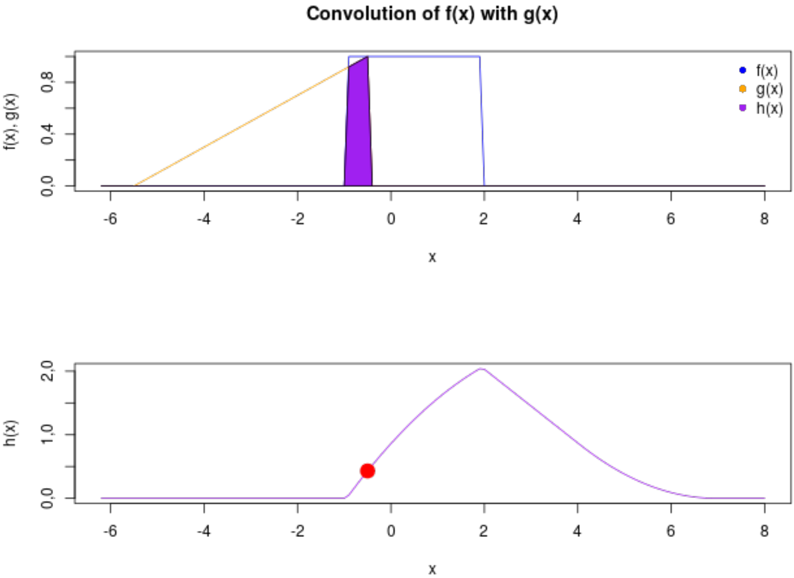
# 1D CONVOLUTION ANIMATION



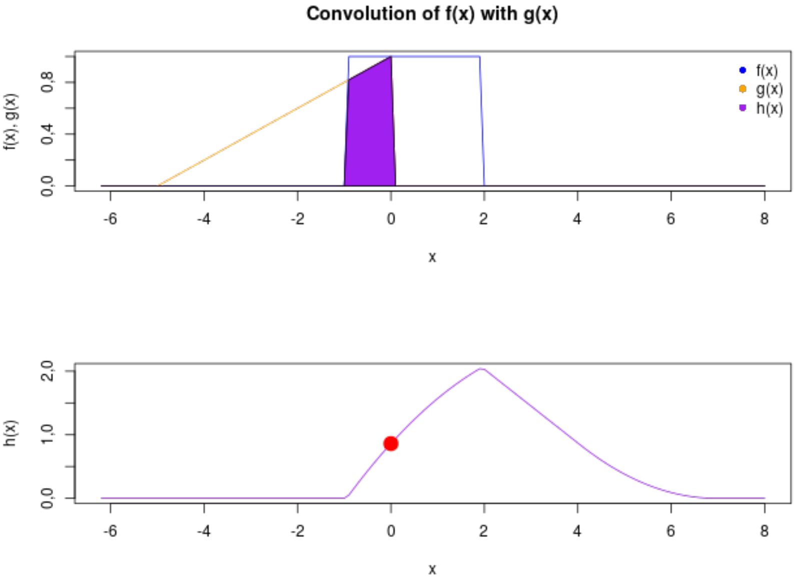
# 1D CONVOLUTION ANIMATION



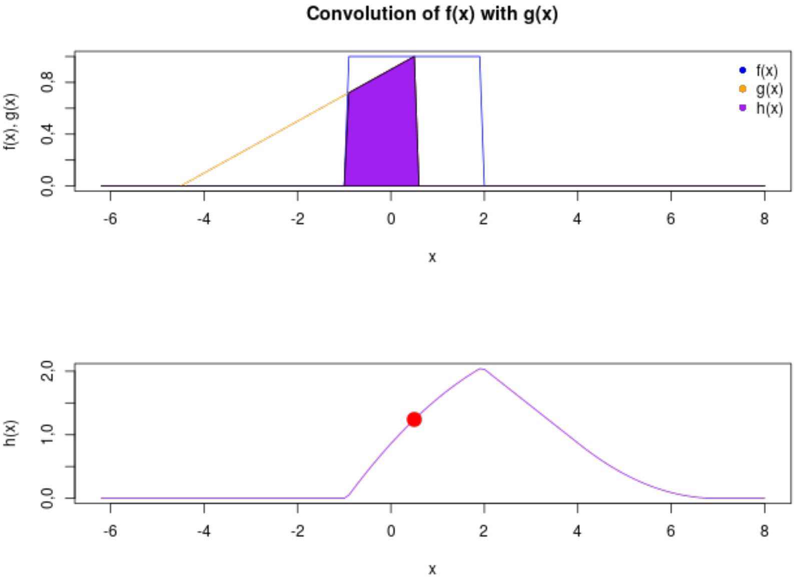
# 1D CONVOLUTION ANIMATION



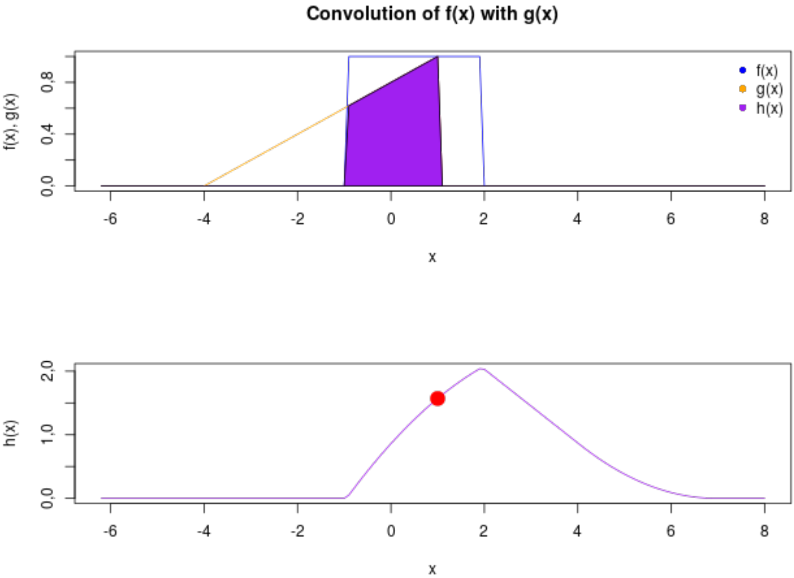
# 1D CONVOLUTION ANIMATION



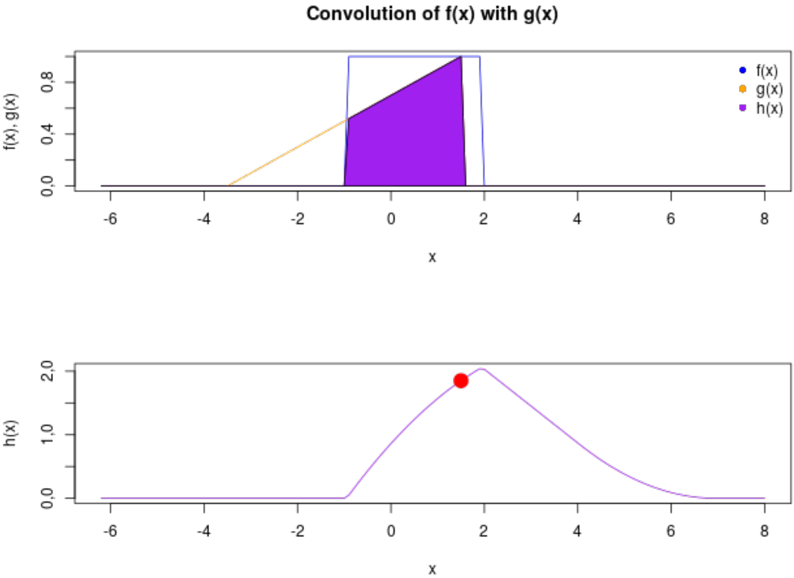
# 1D CONVOLUTION ANIMATION



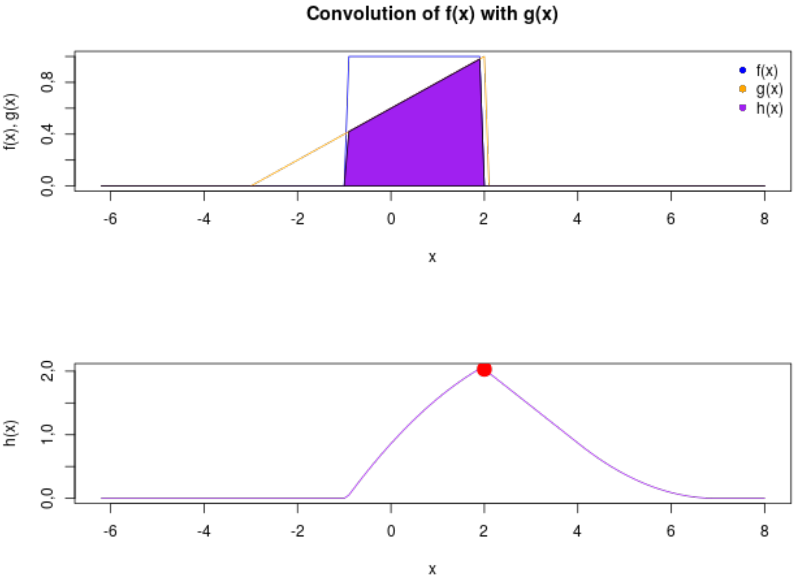
# 1D CONVOLUTION ANIMATION



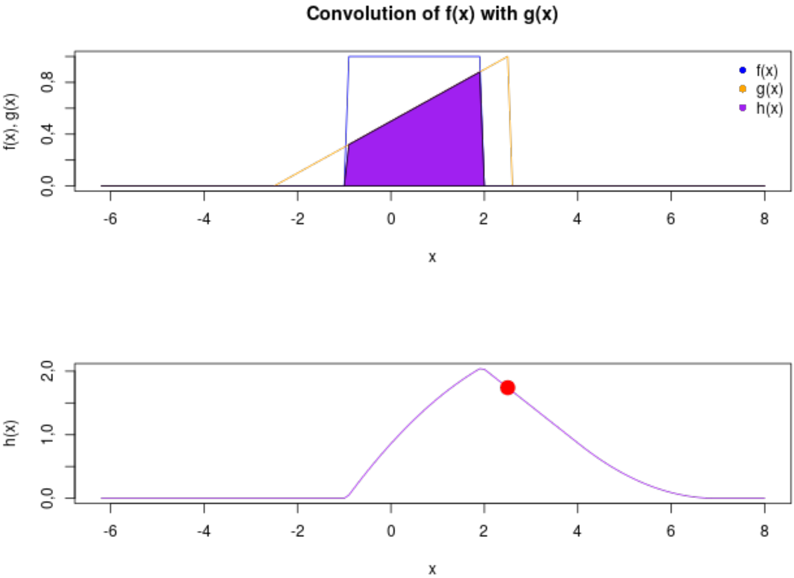
# 1D CONVOLUTION ANIMATION



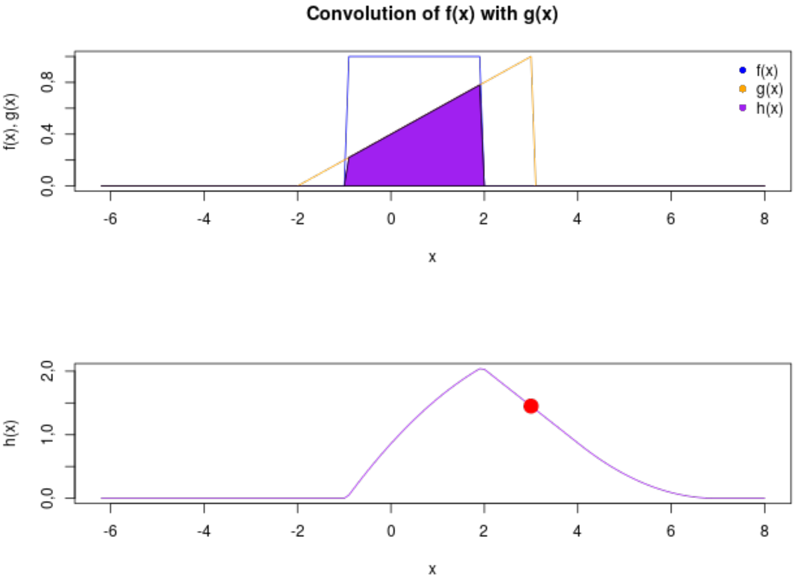
# 1D CONVOLUTION ANIMATION



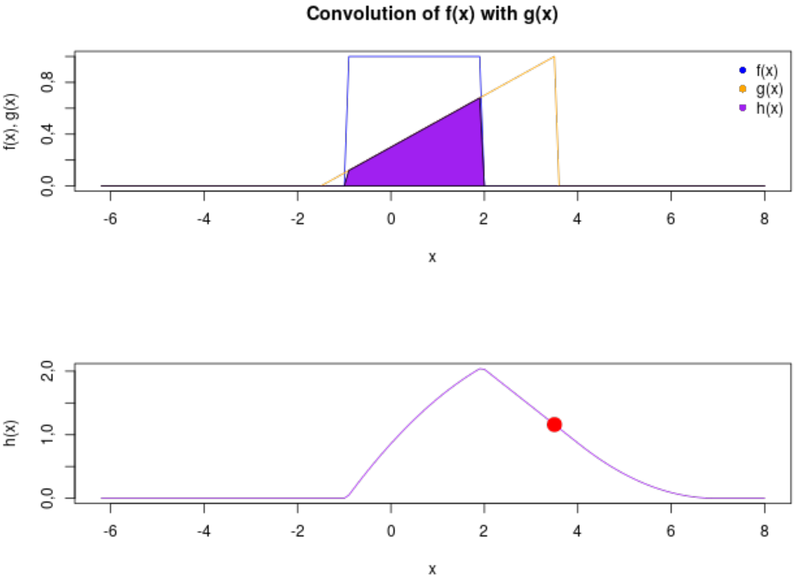
# 1D CONVOLUTION ANIMATION



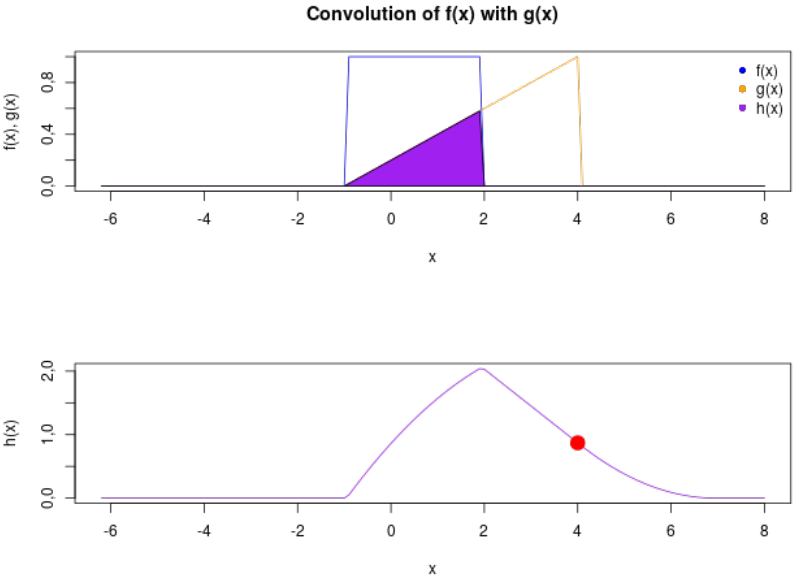
# 1D CONVOLUTION ANIMATION



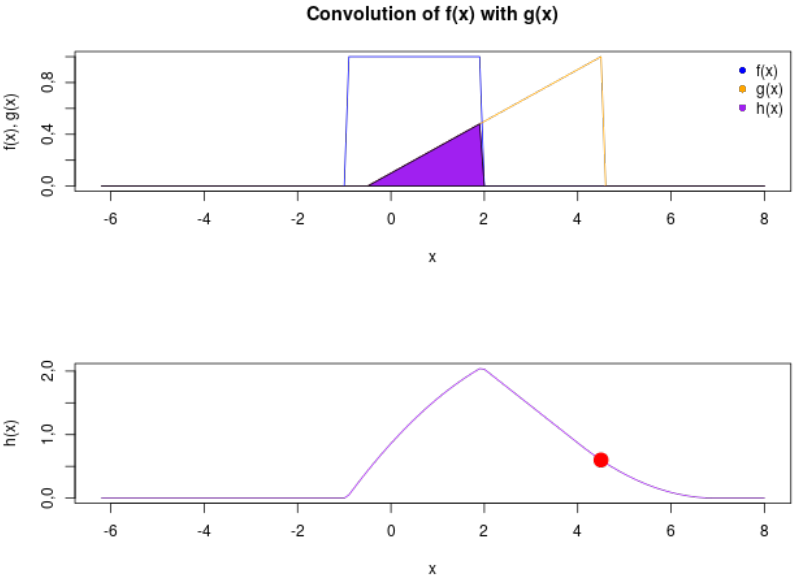
# 1D CONVOLUTION ANIMATION



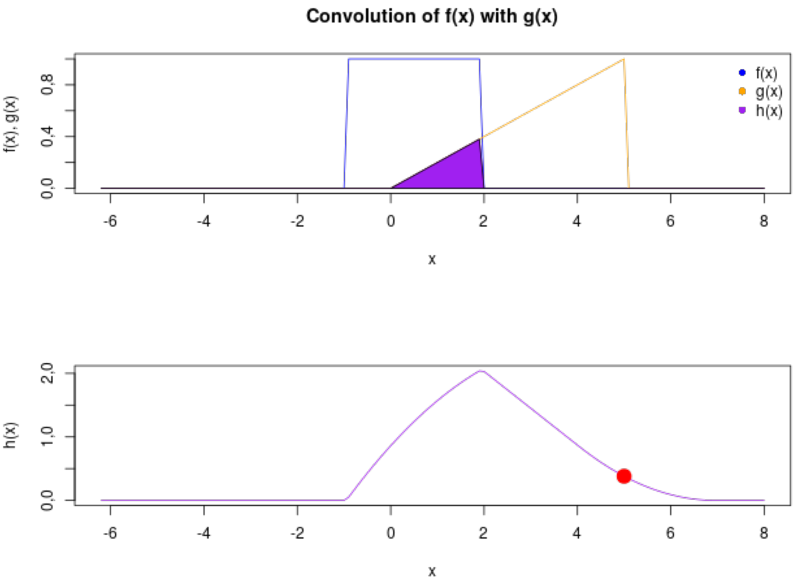
# 1D CONVOLUTION ANIMATION



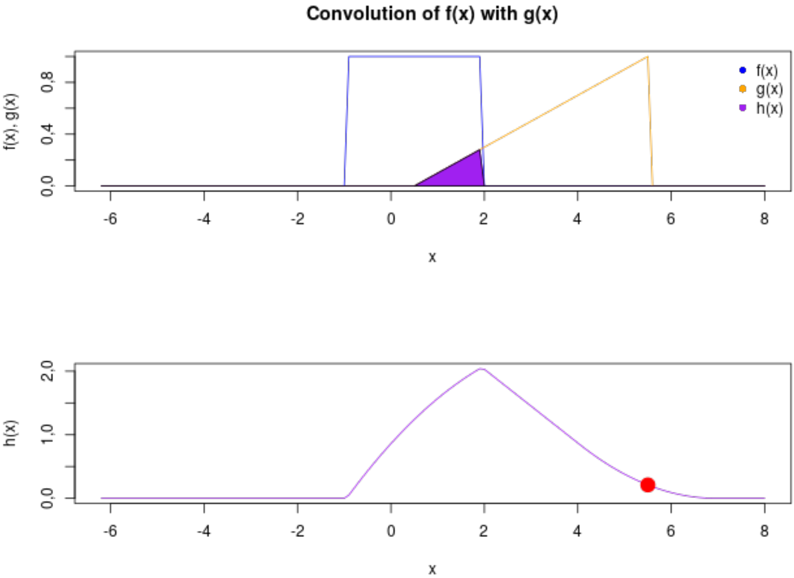
# 1D CONVOLUTION ANIMATION



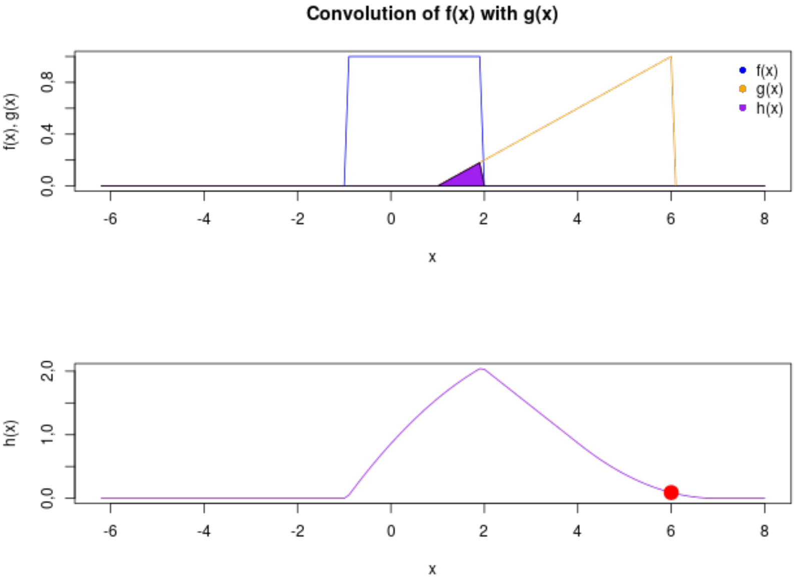
# 1D CONVOLUTION ANIMATION



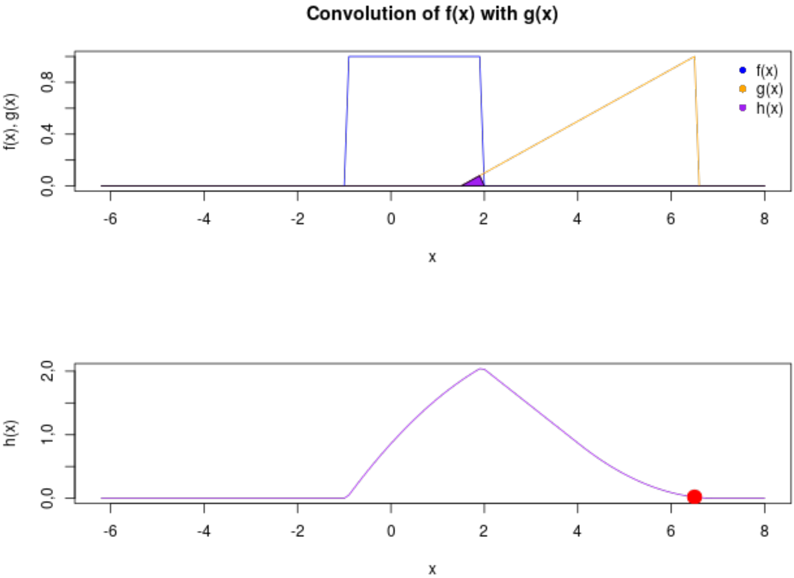
# 1D CONVOLUTION ANIMATION



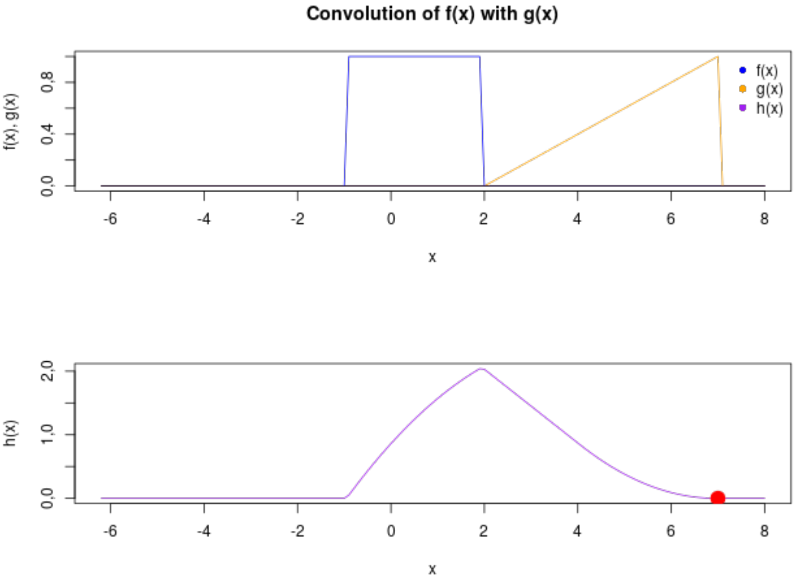
# 1D CONVOLUTION ANIMATION



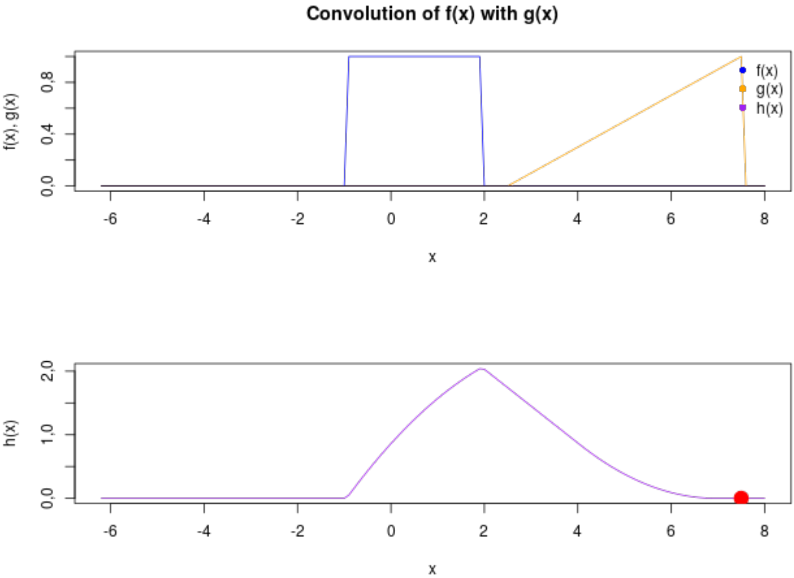
# 1D CONVOLUTION ANIMATION



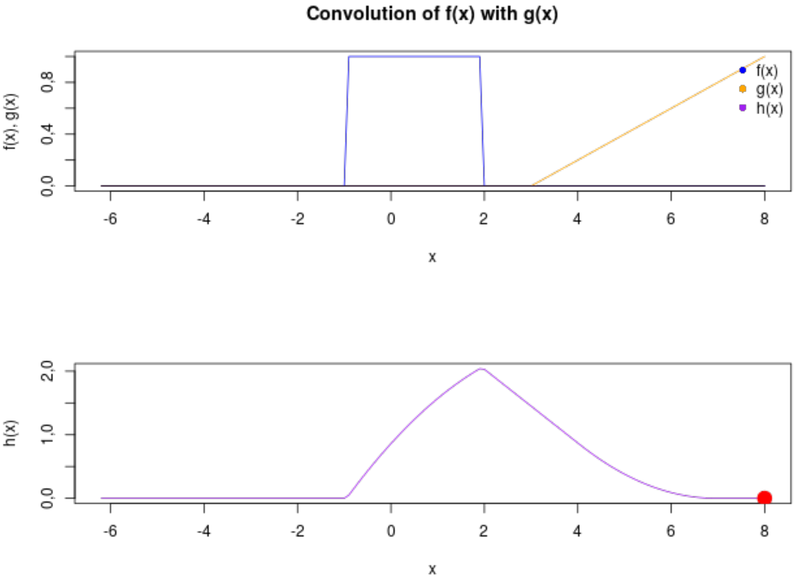
# 1D CONVOLUTION ANIMATION



# 1D CONVOLUTION ANIMATION



# 1D CONVOLUTION ANIMATION



# DISCRETIZATION

- Discretization for one-dimensional input:

$$h(i) = (f * g)(i) = \sum_{x} f(x)g(i - x)$$

- Discretization for 2D images:
  - $\mathcal{I} \in \mathcal{R}^2$ contains two dimensions
  - Use 2D Kernel $\mathcal{G}$ as well to yield feature map $\mathcal{H}$:

$$H(i,j) = (\mathcal{I} * \mathcal{G})(i,j) = \sum_{x} \sum_{y} \mathcal{I}(x,y)\mathcal{G}(i - x, j - y)$$

where $x, y :=$ indices $\mathcal{I}$ and $\mathcal{G}$

and $i, j :=$ indices elements in $\mathcal{H}$

# PROPERTIES OF THE CONVOLUTION

- Commutativity:

$$f * g = g * f$$

- Associativity:

$$(f * g) * h = f * (g * h)$$

- Distributivity:

$$f * (g + h) = f * g + f * h$$

$$\alpha(f * g) = (\alpha f) * g \text{ for scalar } \alpha$$

- Differentiability:

$$\frac{\partial (f * g)(x)}{\partial x_i} = \frac{\partial f(x)}{\partial x_i} * g(x) = \frac{\partial g(x)}{\partial x_i} * f(x)$$

$\rightarrow$ $(f * g)(x)$ is as many times differentiable as the max of $g(x)$ and $f(x)$.

# RELATED OPERATIONS

- Convolution is strongly related to two other mathematical operators:
  1. Fourier transform via the Convolution Theorem
  2. Cross correlation

# CONVOLUTION THEOREM

- Fourier transform of the convolution of two functions can be expressed as the product of their Fourier transforms:

$$\mathcal{F}\{f * g\} = \mathcal{F}\{f\}\mathcal{F}\{g\}$$

- Transformation of a signal from time to frequency domain.
- Convolution in the time domain is equivalent to multiplication in frequency domain.
- The computationally fastest way to compute a convolution is therefore taking the Fourier inverse of the multiplication of the Fourier-transformed input and filter function :

$$(f * g)(t) = \mathcal{F}^{-1}\{\mathcal{F}\{f(t)\}\mathcal{F}\{g(t)\}\}$$

# CONVOLUTION THEOREM - PROOF

$$
\begin{aligned}
\widehat{(f * g)}(t) &= \int_{-\infty}^{\infty} exp(-2\pi i\omega t)\Big[\int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau\Big] \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} exp(-2\pi i\omega t)f(\tau)g(t-\tau)d\tau dt \\
&\overset{Fubini}{=} \int_{-\infty}^{\infty}\Big[\int_{-\infty}^{\infty} exp(-2\pi i\omega t)f(\tau)g(t-\tau)dt\Big]d\tau \\
&\overset{f(\tau)\perp t}{=} \int_{-\infty}^{\infty} f(\tau)\Big[\int_{-\infty}^{\infty} exp(-2\pi i\omega t)g(t-\tau)dt\Big]d\tau \\
&\overset{u=t-\tau}{=} \int_{-\infty}^{\infty} f(\tau)\Big[\int_{-\infty}^{\infty} exp(-2\pi i\omega\tau)exp(-2\pi i\omega u)g(u)du\Big]d\tau \\
&= \int_{-\infty}^{\infty} exp(-2\pi i\omega\tau)f(\tau)\Big[\int_{-\infty}^{\infty} exp(-2\pi i\omega u)g(u)du\Big]d\tau \\
&\overset{Fubini}{=} \dots
\end{aligned}
$$

# CONVOLUTION THEOREM - PROOF

$$... \int_{-\infty}^{\infty} exp(-2\pi i\omega\tau)f(\tau)d\tau \int_{-\infty}^{\infty} exp(-2\pi i\omega u)g(u)du$$
$$= f(\hat{t})g(\hat{t})$$

# CROSS CORRELATION

- Measurement for similarity of two functions $f(x), g(x)$.
- More specifically, at which position are the two functions most similar to each other? Where does the pattern of $g(x)$ match $f(x)$ the best?
- Intuition:
  - Slide with $g(x)$ over $f(x)$ and at each discrete step compute the sum of the product of their elements.
  - When peaks of both functions are aligned, the product of high (positive or negative) values will lead to high sums.
  - Thus, both functions are most similar at points with equal peaks.

# CROSS CORRELATION

- Definition:

$$h(i) = (f \star g)(i) = \int_{-\infty}^{\infty} f(x)g(i+x)dx$$

where $f(x)$ : input function

and $g(x)$ : weighting function, kernel

and $h(i)$ : output function, feature map elements

for $f, g \in \mathcal{R}^d \mapsto h \in \mathcal{R}^d$

# CROSS CORRELATION

- Discrete formulation:

$$h(i) = (f \star g)(i) = \sum_{x=-\infty}^{\infty} f(x)g(i+x)$$

- Thus:

$$f(i) \star g(i) = f(-i) * g(i)$$

- Remember: $*$ is used for convolution and $\star$ for cross correlation.
- Similar formulation as the convolution despite the flipped filter function in the convolutional kernel .

# CROSS CORRELATION

- This operation also works in 2 dimensions
- The difference w.r.t. the convolution are the positive iterators in the sum:

$$H(i,j) = (\mathcal{I} \star \mathcal{G})(i,j) = \sum_x \sum_y \mathcal{I}(x,y)\mathcal{G}(i+x, j+y)$$

where $x, y :=$ indices $\mathcal{I}$ and $\mathcal{G}$

and $i, j :=$ indices elements in $\mathcal{H}$

# CROSS CORRELATION

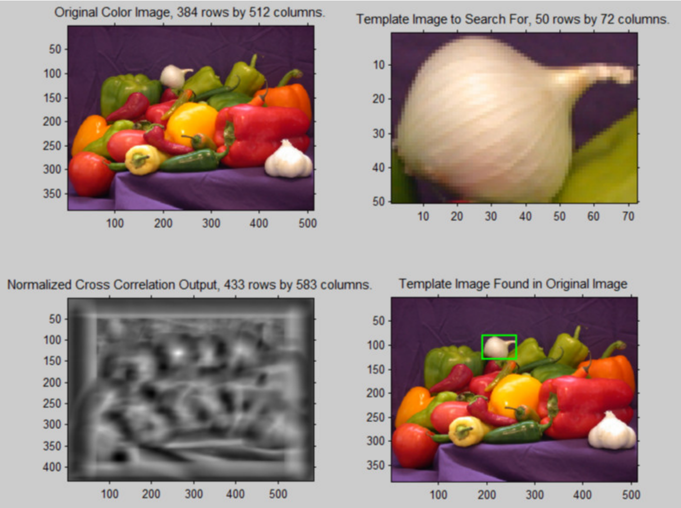
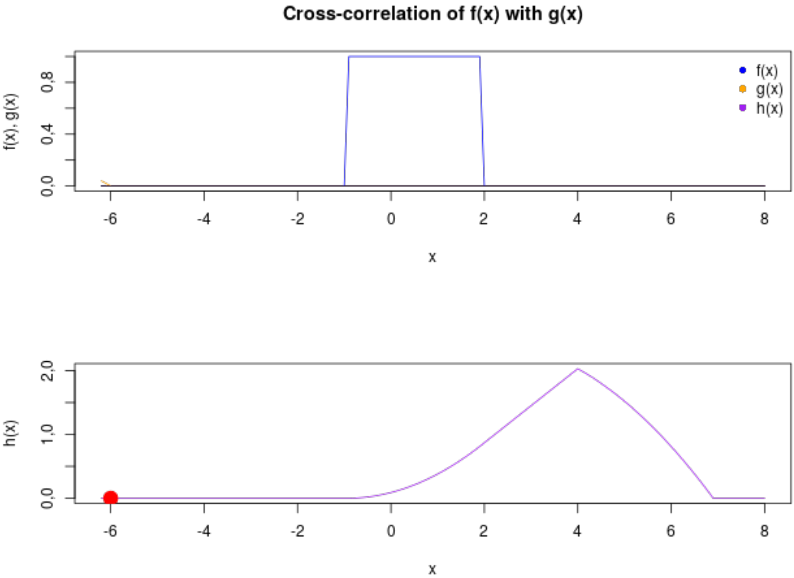



**Figure:** Cross-correlation used to detect a template (onion) in an image. Cross correlation peaks (white) at the position where template and input match best.

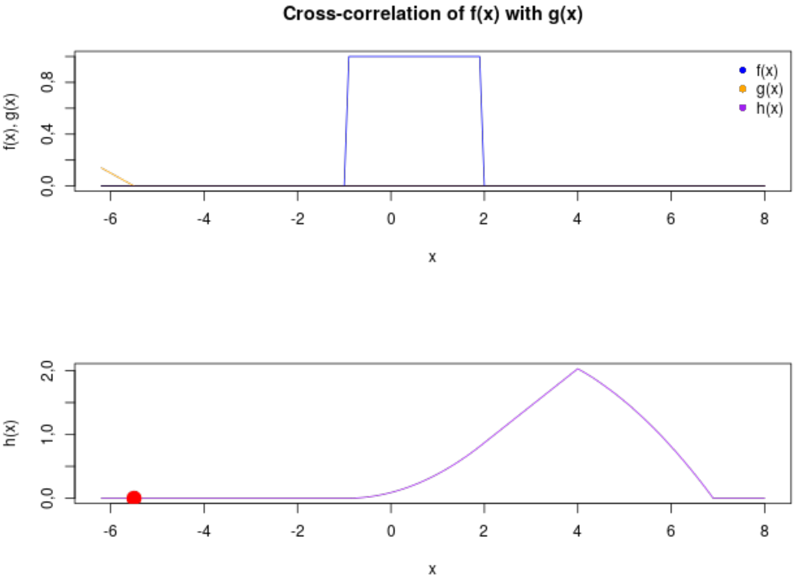
# CROSS CORRELATION

- From the following animation we see that
  - Kernel is not flipped as opposed to the convolution.
  - Cross-Correlation peaks, where the filter matches the signal the most.
- In some frameworks, Cross-Correlation is implemented instead of the convolution due to
  - better computational performance.
  - similar properties, as the kernel weights are learned throughout the training process.

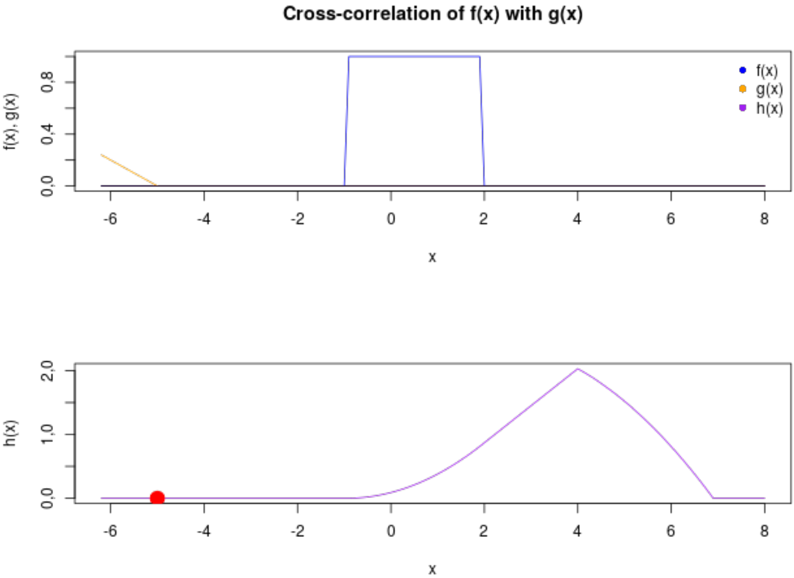
# 1D CROSS CORRELATION ANIMATION



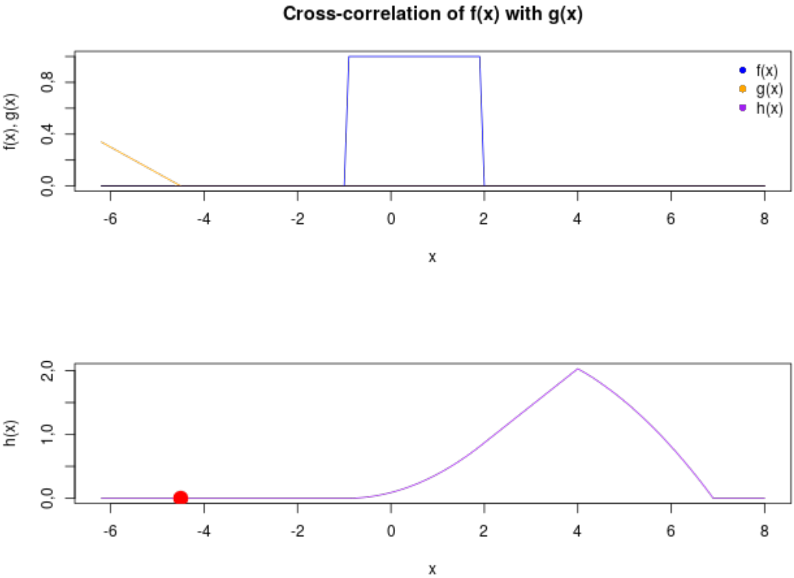
# 1D CROSS CORRELATION ANIMATION



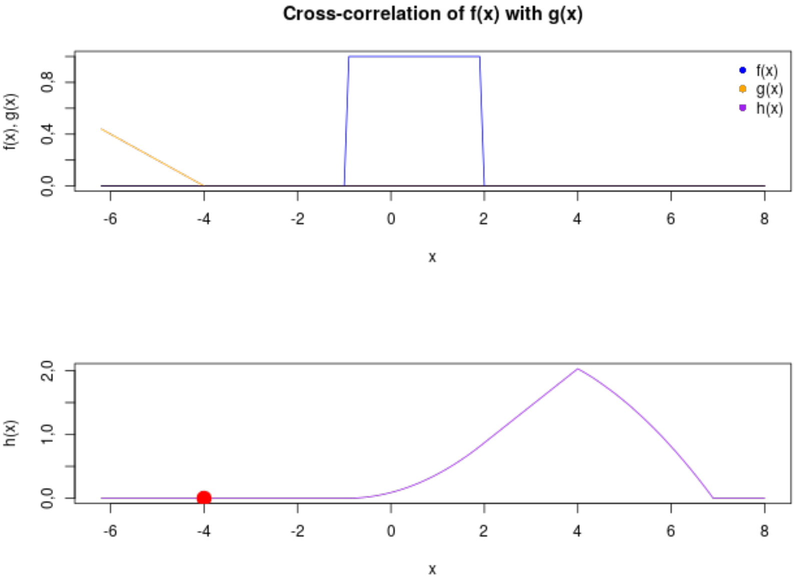
# 1D CROSS CORRELATION ANIMATION



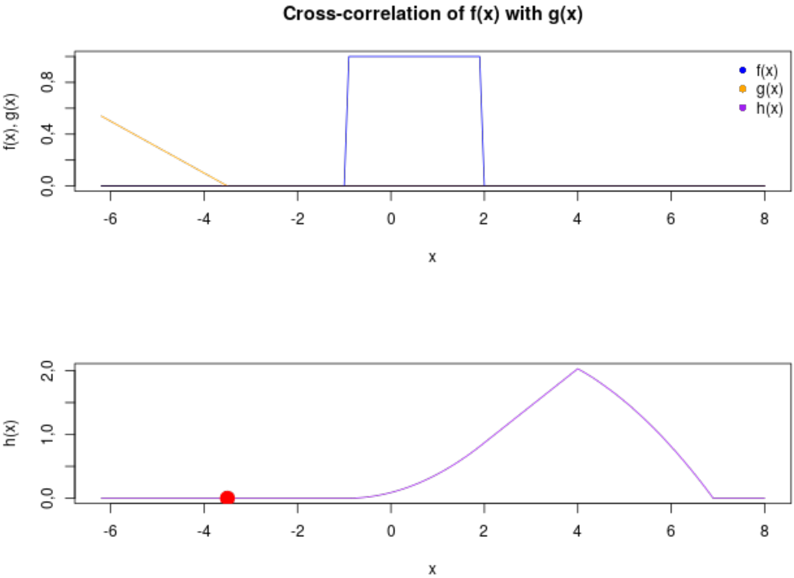
# 1D CROSS CORRELATION ANIMATION



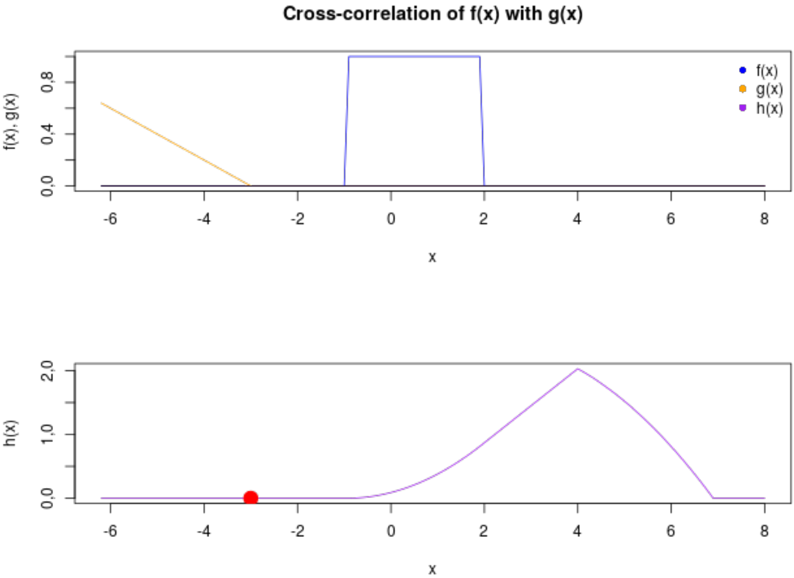
# 1D CROSS CORRELATION ANIMATION



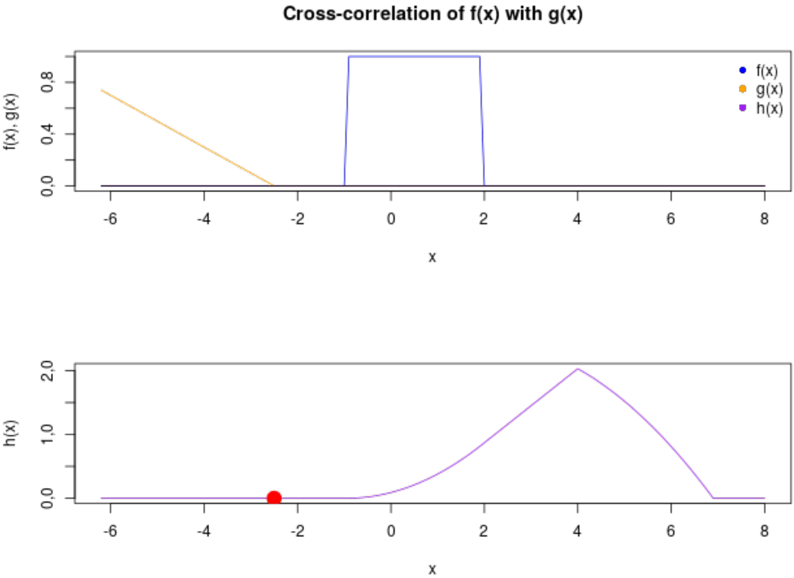
# 1D CROSS CORRELATION ANIMATION



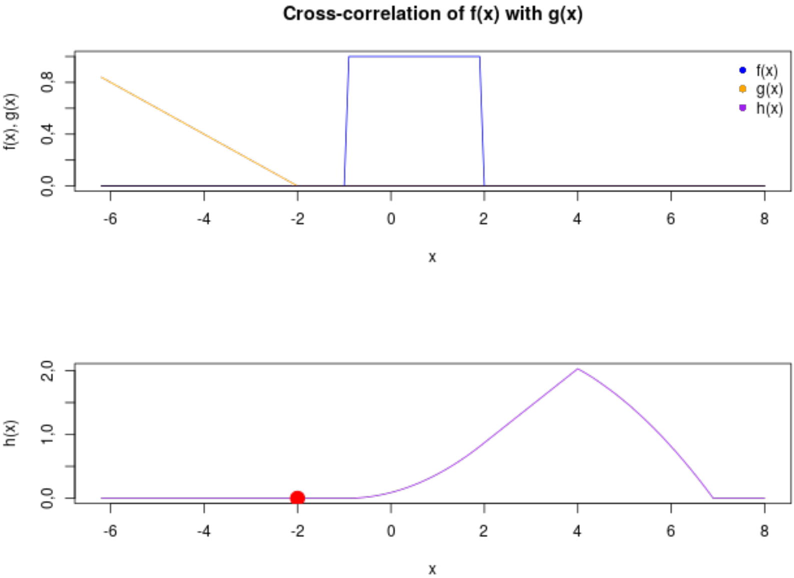
# 1D CROSS CORRELATION ANIMATION



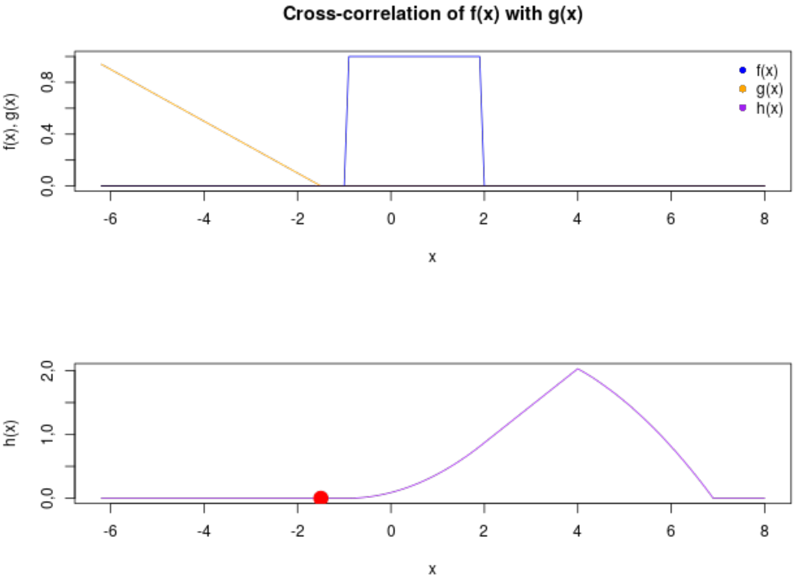
# 1D CROSS CORRELATION ANIMATION



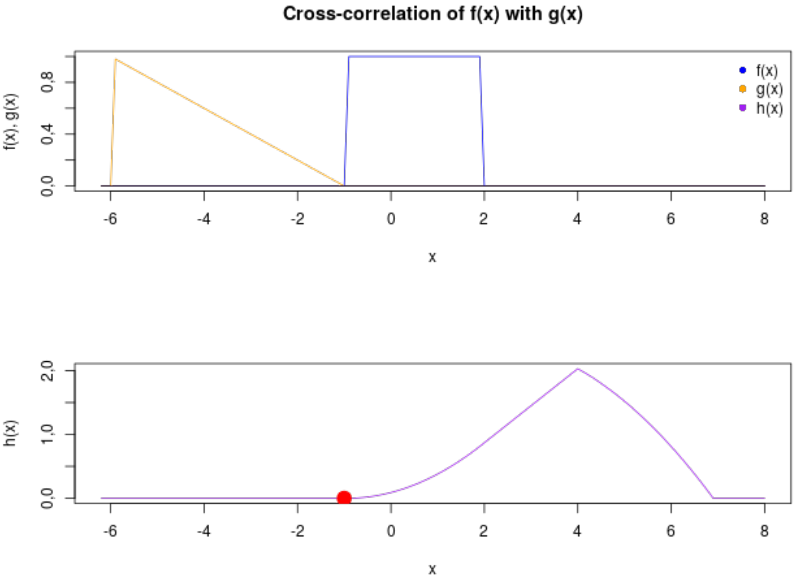
# 1D CROSS CORRELATION ANIMATION



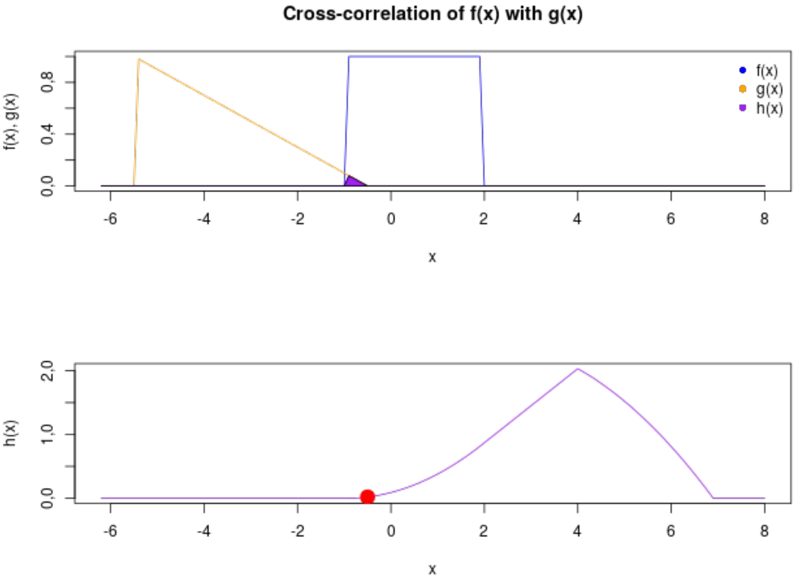
# 1D CROSS CORRELATION ANIMATION



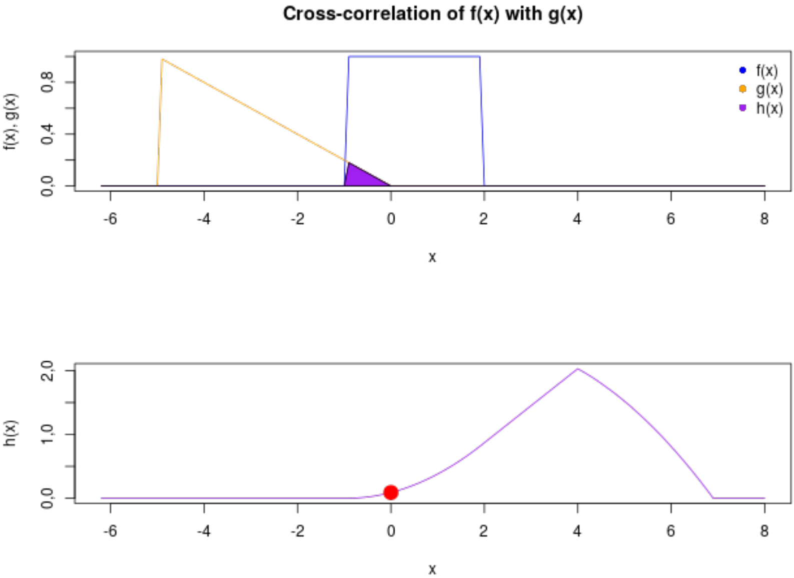
# 1D CROSS CORRELATION ANIMATION



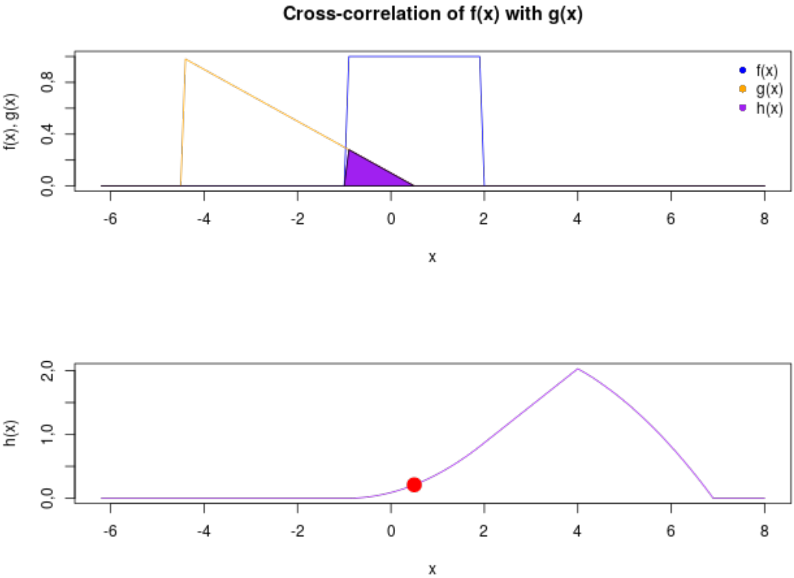
# 1D CROSS CORRELATION ANIMATION



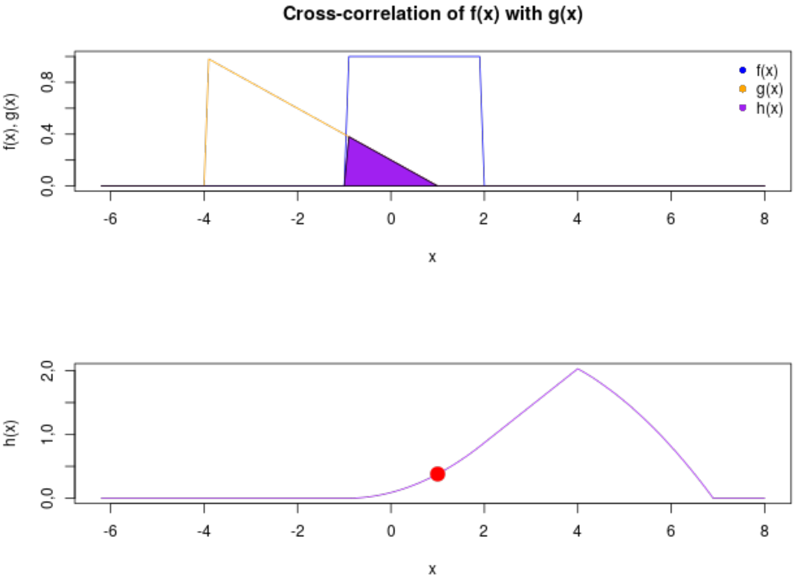
# 1D CROSS CORRELATION ANIMATION



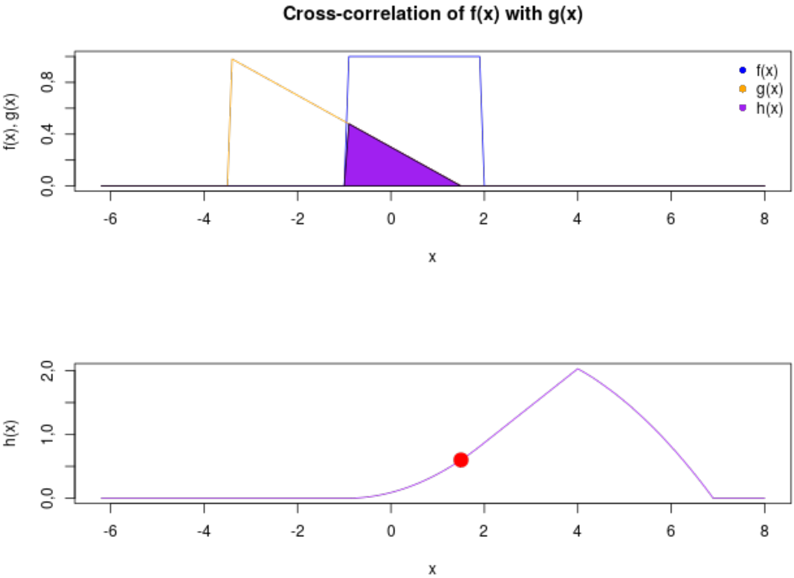
# 1D CROSS CORRELATION ANIMATION



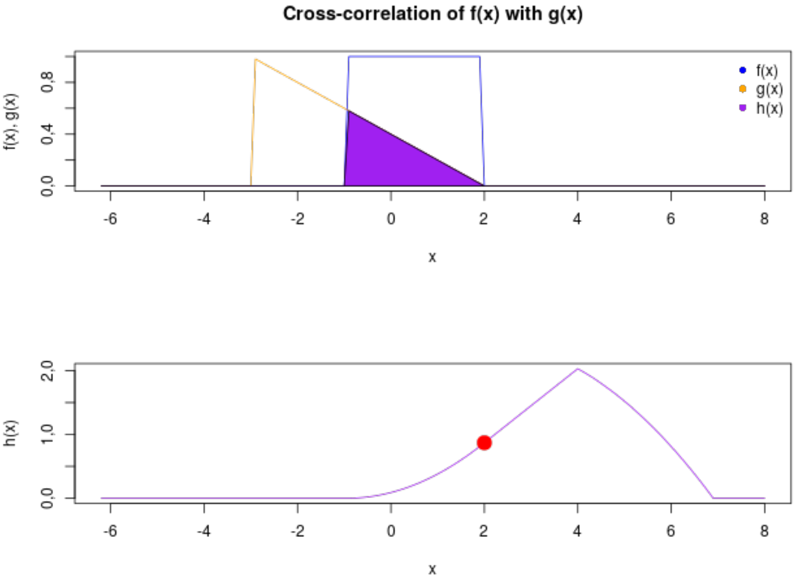
# 1D CROSS CORRELATION ANIMATION



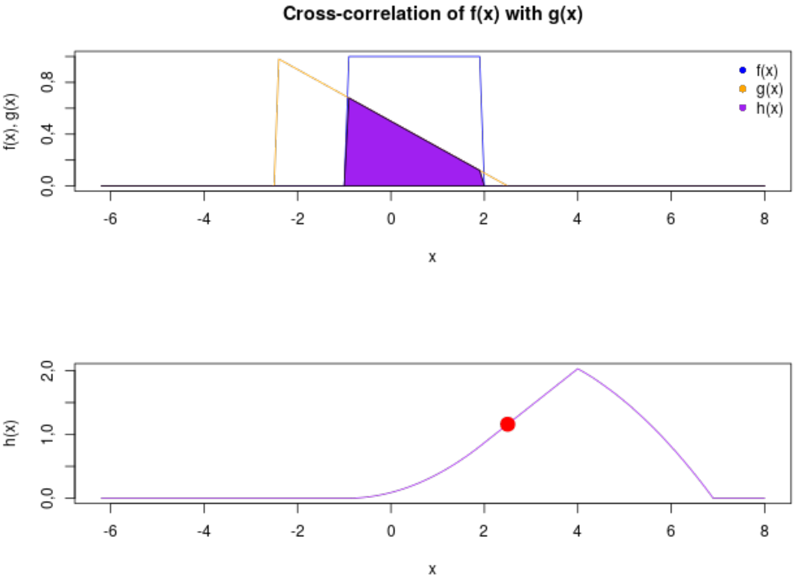
# 1D CROSS CORRELATION ANIMATION



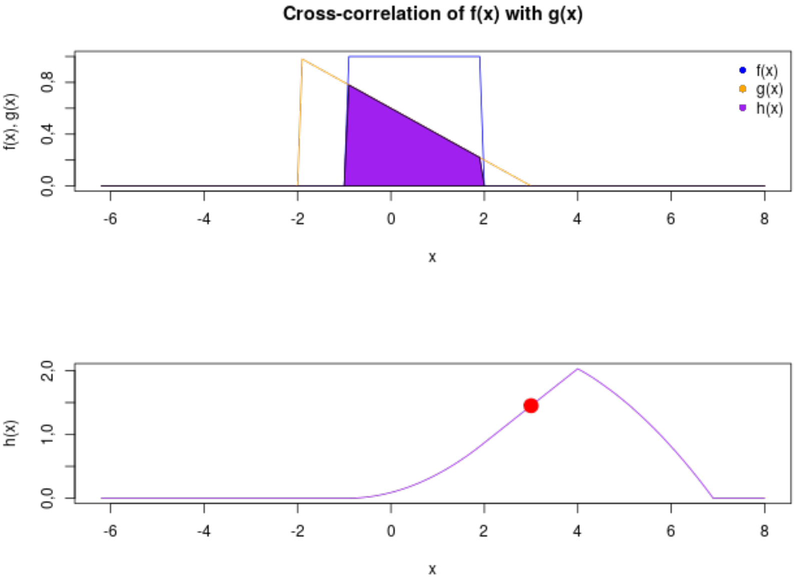
# 1D CROSS CORRELATION ANIMATION



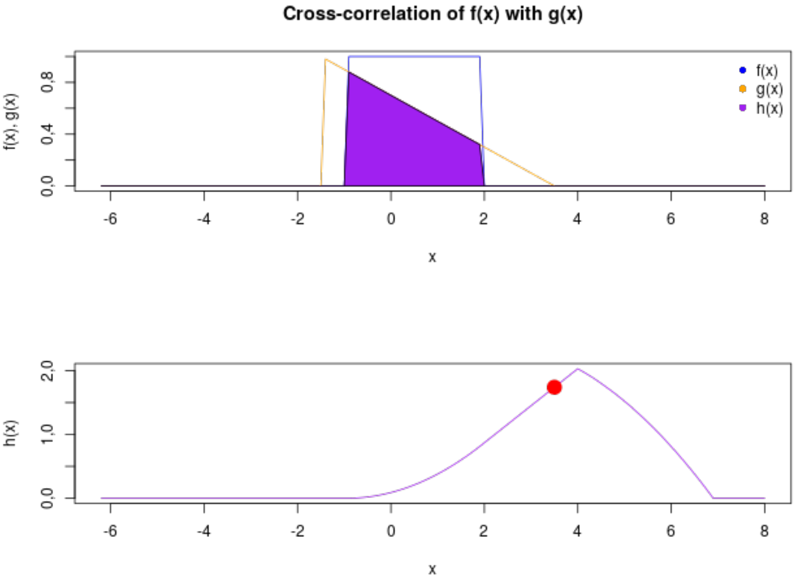
# 1D CROSS CORRELATION ANIMATION



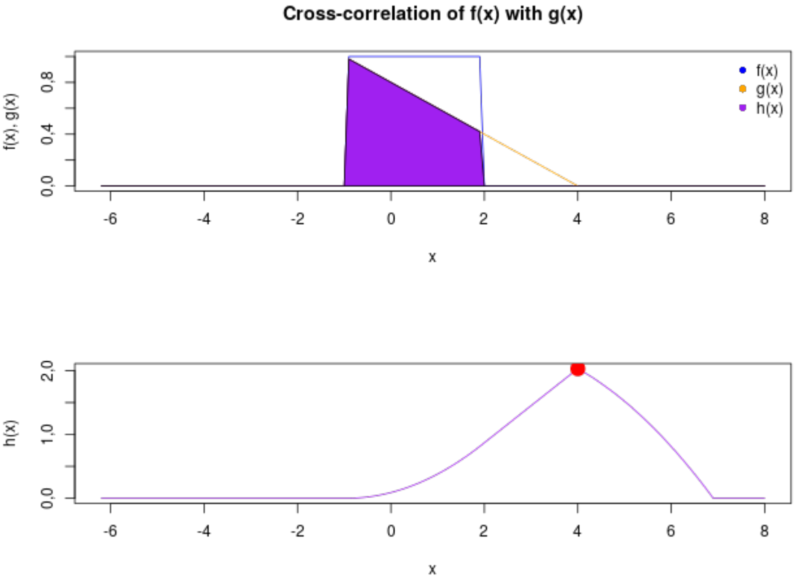
# 1D CROSS CORRELATION ANIMATION



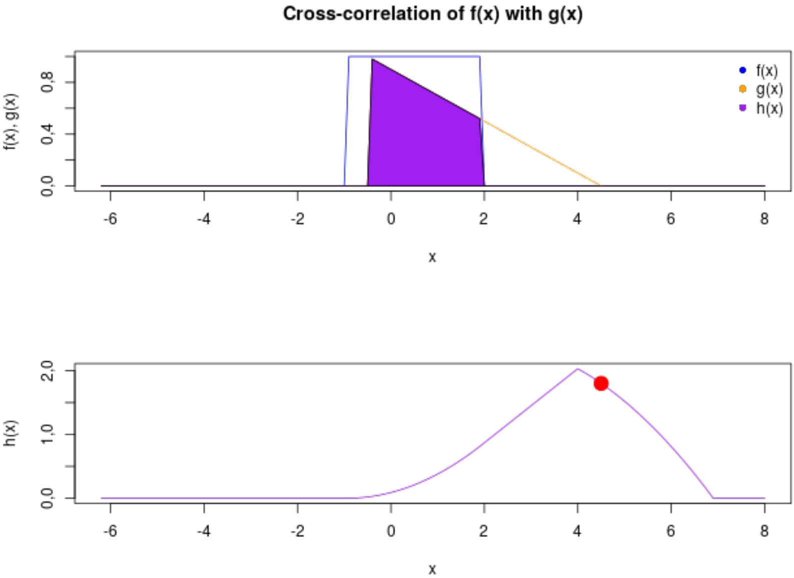
# 1D CROSS CORRELATION ANIMATION



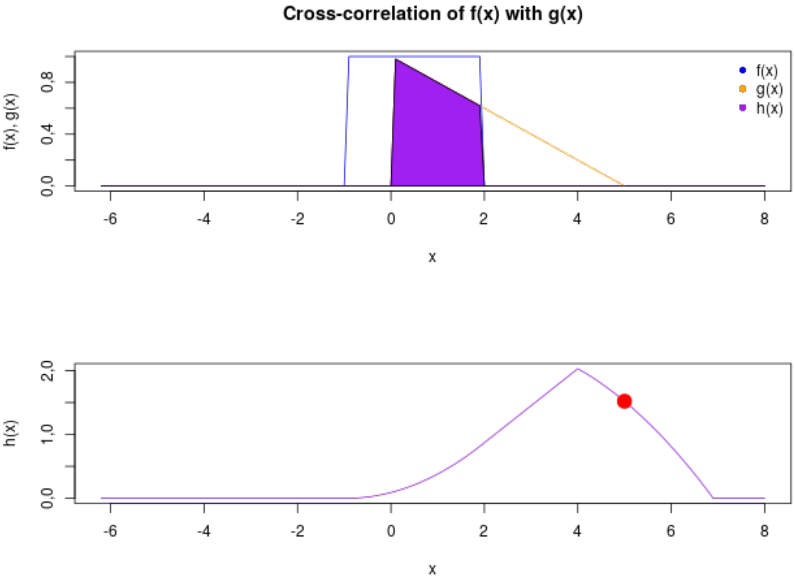
# 1D CROSS CORRELATION ANIMATION



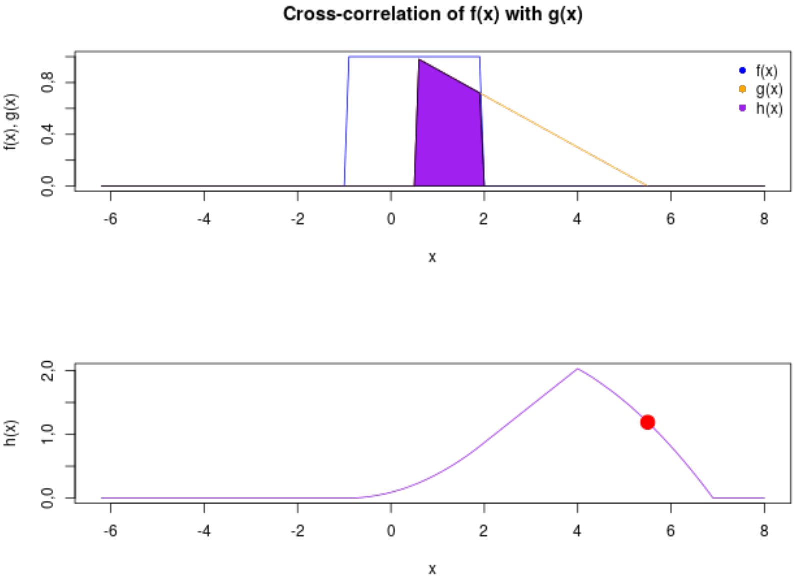
# 1D CROSS CORRELATION ANIMATION



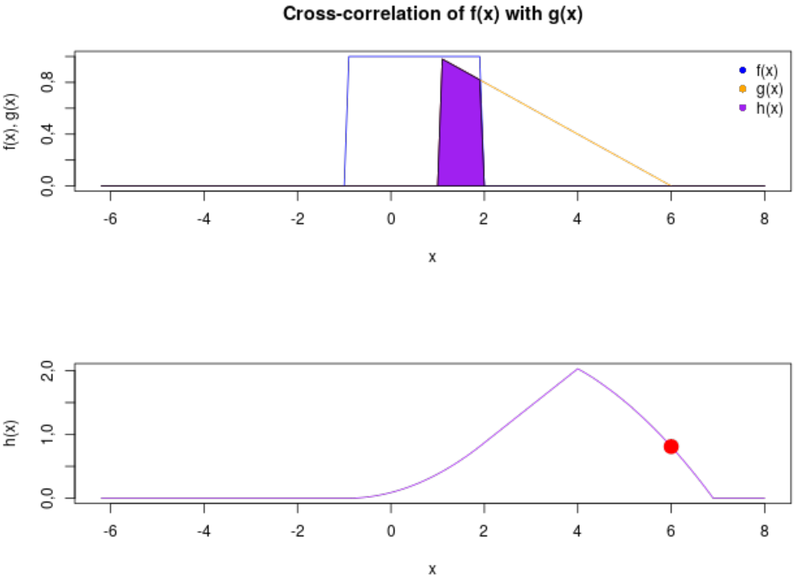
# 1D CROSS CORRELATION ANIMATION



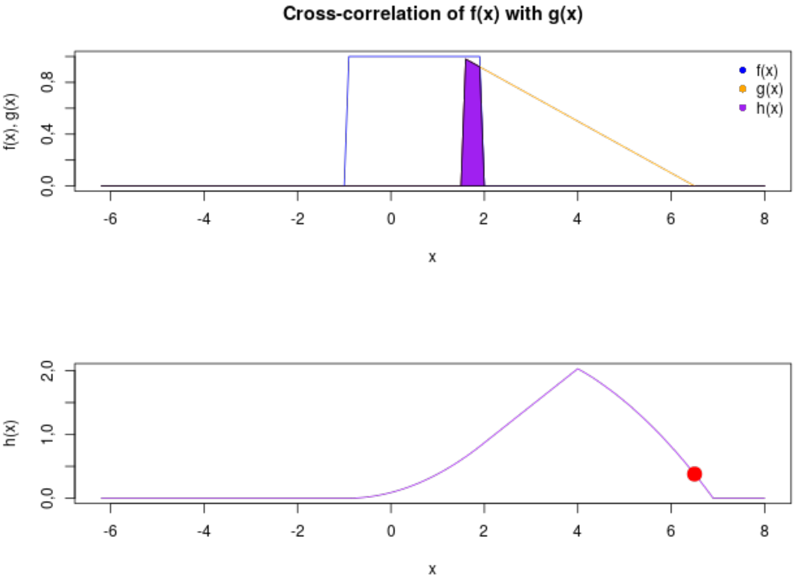
# 1D CROSS CORRELATION ANIMATION



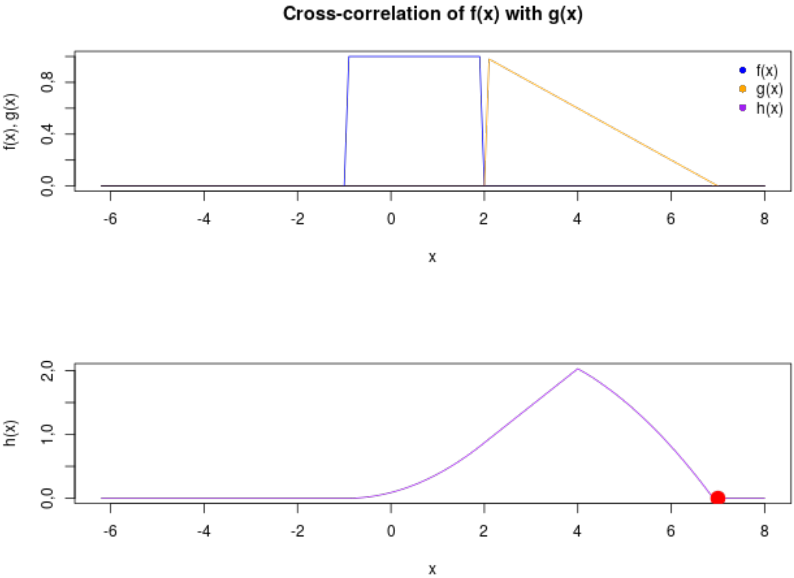
# 1D CROSS CORRELATION ANIMATION



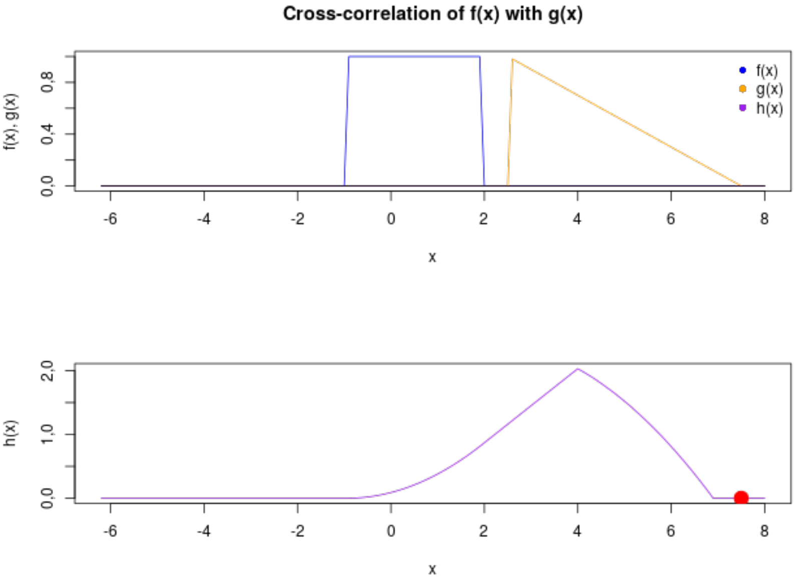
# 1D CROSS CORRELATION ANIMATION



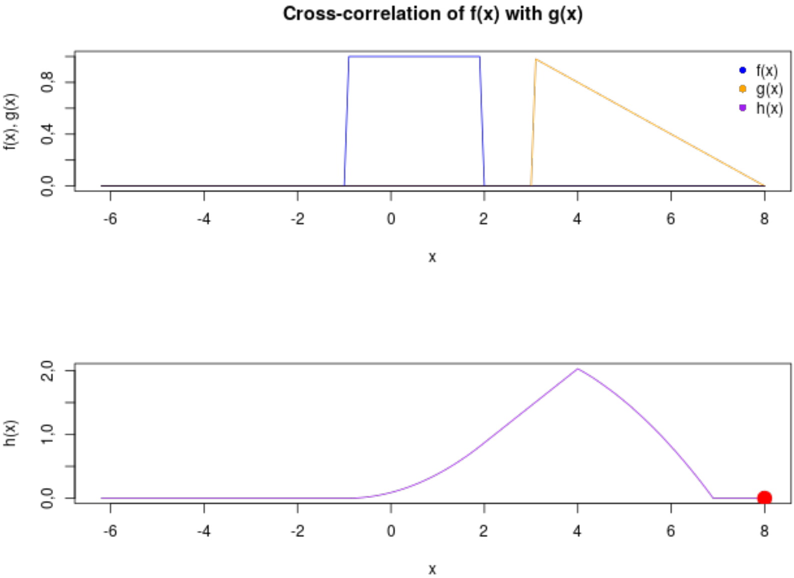
# 1D CROSS CORRELATION ANIMATION



# 1D CROSS CORRELATION ANIMATION



# 1D CROSS CORRELATION ANIMATION



# CROSS CORRELATION VS. CONVOLUTION

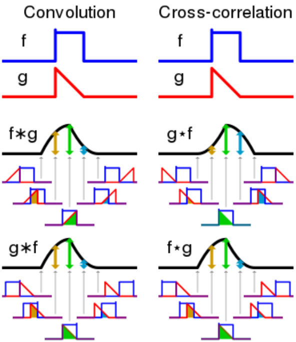



**Figure:** Comparison of convolution and cross-correlation

- Cross correlation is not commutative
- but often implemented instead of convolution in the practice.