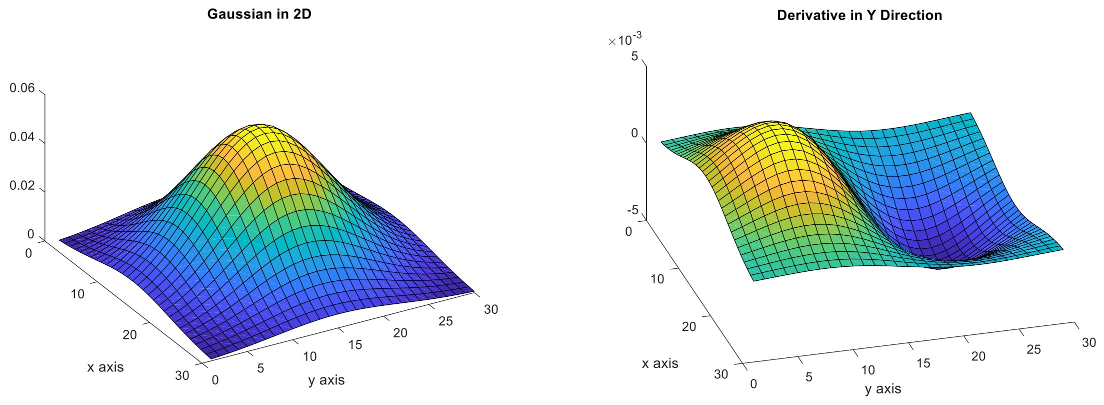
The Canny Edge Detector

We have seen how to detect edges using the simple kernel based method as part of image filtering. We will now look at a popular edge detection method developed by John Canny in 1986. This algorithm has proven to be robust and provides more reliable edge point estimates on a wide range of conditions and image resolutions.

The algorithm:

1. Precalculate derivative of Gaussian filters in the X and Y directions. Filter image with these filters separately to find the matrices G_x and G_y. A 2D Gaussian is shown below on the left and the derivative of the Gaussian in the Y direction is shown on the right.



   Alternatively you can filter the image with a 2D Gaussian and then use the Sobel operator. These are the filters we used previously for edge detection.

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \qquad \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

2. Find magnitude and direction of the gradient

   gradient magnitude = $\sqrt{{G_x}^2 + {G_y}^2}$

   direction = $\arctan\left(\frac{G_y}{G_x}\right)$

3. Apply non-maximum suppression to remove pixels that are not considered to be part of an edge to leave only thin lines.

   This step converts the blurred edges obtained as the gradient magnitudes to sharp edges.

   a) Round the direction to the nearest 45 degrees (in 0°, 45°, 90° and 135°).

b) Compare the edge strength to two of its neighbors along the gradient direction.
c) Keep if the pixel is maximum, otherwise suppress it.

4. Hysteresis thresholding: Canny uses two thresholds, upper and lower:
   a. Accept pixel as edge if its gradient is higher than the upper threshold.
   b. Reject pixel if its gradient value is below the lower threshold.
   c. Accept pixel only if its gradient is between the two thresholds and it is connected to a pixel that is above the upper threshold. Typical values: upper to lower ratio between 2 and 3.

Matlab has an implementation of this edge detector. It can be used in the following way:

```
C = edge(A, 'canny', [0.3, 0.9] );
```

Where the vector argument with two elements are for lower and upper edge thresholds respectively.

Matlab implements a number of edge detectors. For example, the one we used in our filtering discussion (the Sobel operator) is the default and can be used simply by specifying its name:

```
B = edge(A, 'sobel');
```

Original paper:

J. Canny, A Computational Approach to Edge Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 8, No. 6, Nov 1986.