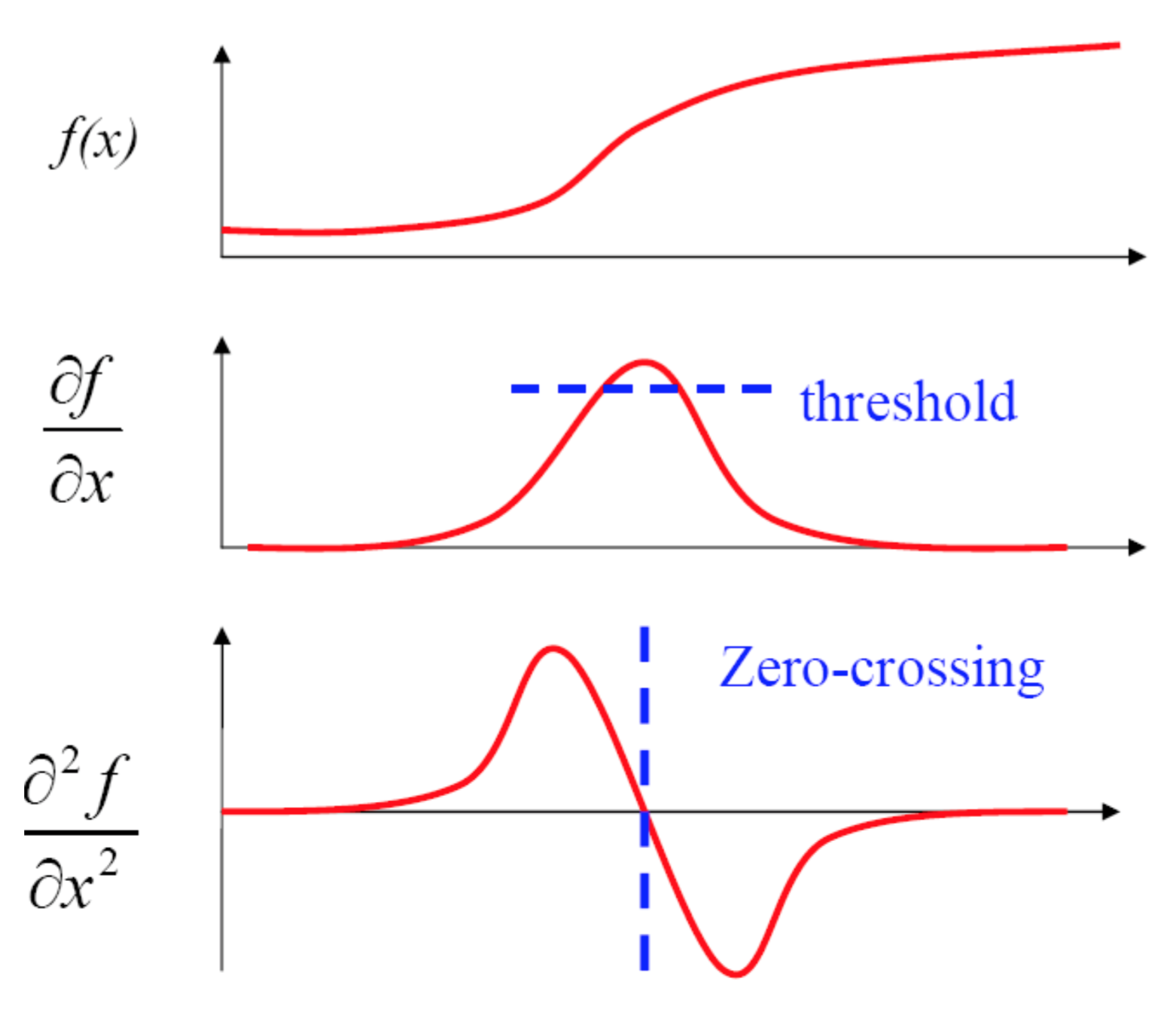
**CV hw10 / 電機所R06921082 陳與賢**

**Description:**

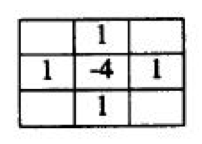
利用python來處理bmp檔，使用各種mask以及threshold來做edge detection，跟hw9的差別在於這次是zero crossing的版本，hw9是用一階微分後找pixel值劇烈變化的地方，hw10是用二階微分找到pixel值變化的轉折點，如下圖所示：



使用二階微分的優點在於，若邊界較寬，則pixel值變化和緩，那用一階微分就會難以區別何處才是真正的edge，而二階微分可以較為準確的標出變化最劇烈的地方。

**Algorithm:**

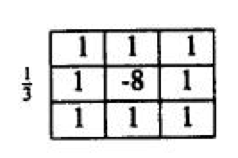
**@ Laplace Mask1**

**kernel:** 

**threshold:** **20**

將input img與此kernel做cross-correlation，得到的值若大於threshold則output給值0，反之給值255

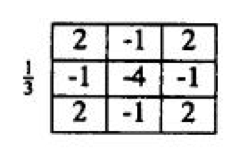
**@ Laplace Mask2**

**kernel:** 

**threshold:** **20**

將input img與此kernel做cross-correlation，得到的值若大於threshold則output給值0，反之給值255

**@ Minimum variance Laplacian**

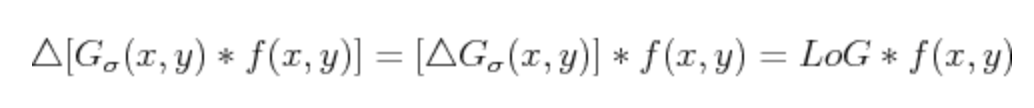
**kernel:** 

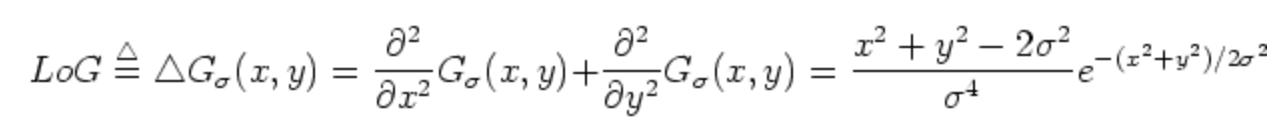
**threshold:** **20**

將input img與此kernel做cross-correlation，得到的值若大於threshold則output給值0，反之給值255

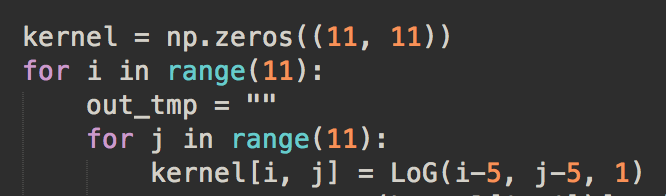
**@ Laplace of Gaussian**

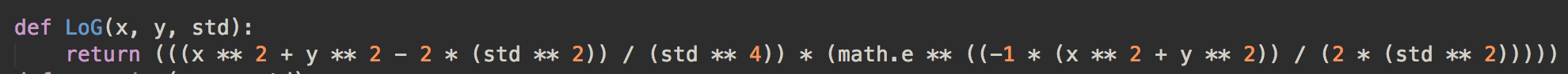
首先用下面公式算出kernel

****

****

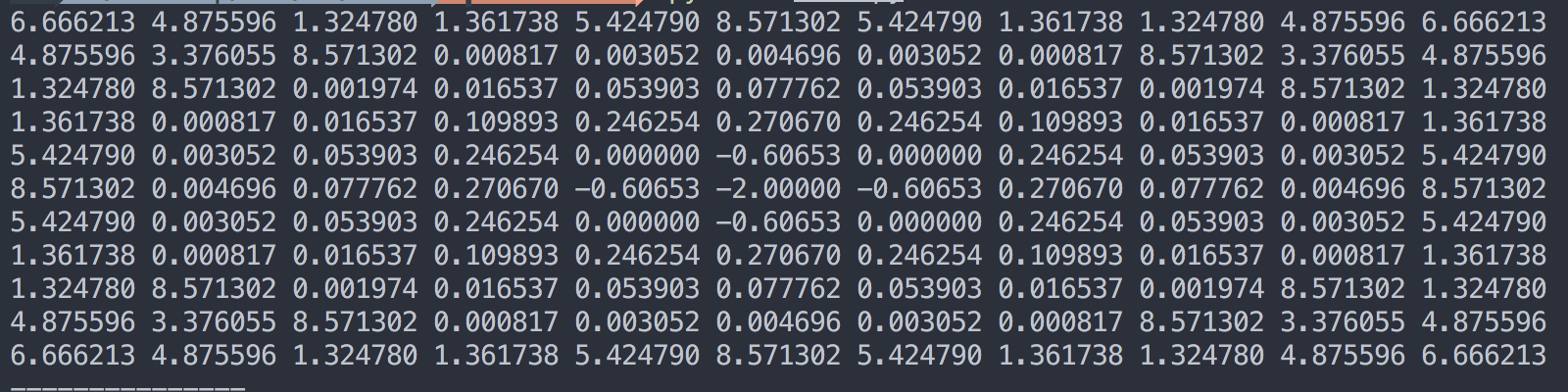
code如下：





**sigma: 1**

**kernel:**（因為排版 我沒有把全部小數點都印出來）

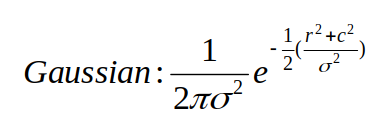


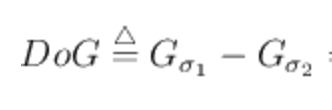
**threshold: 30**

將input img與此kernel做cross-correlation，得到的值若大於threshold則output給值0，反之給值255

**@ Difference of Gaussian**

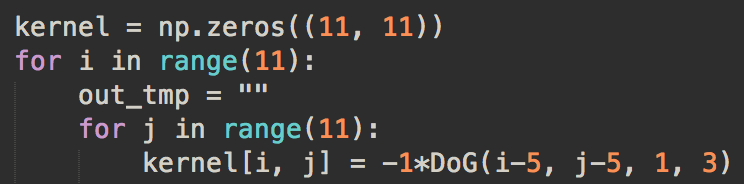
首先用下面公式算出kernel

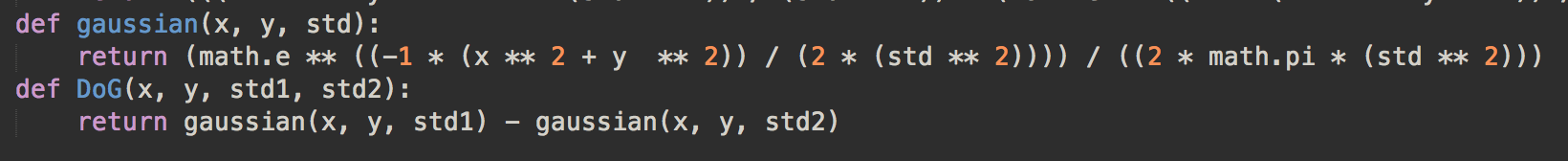


****

code如下：

（因為用G(1)-G(3)出來的值都會是負的，所以我都先乘-1變正數）

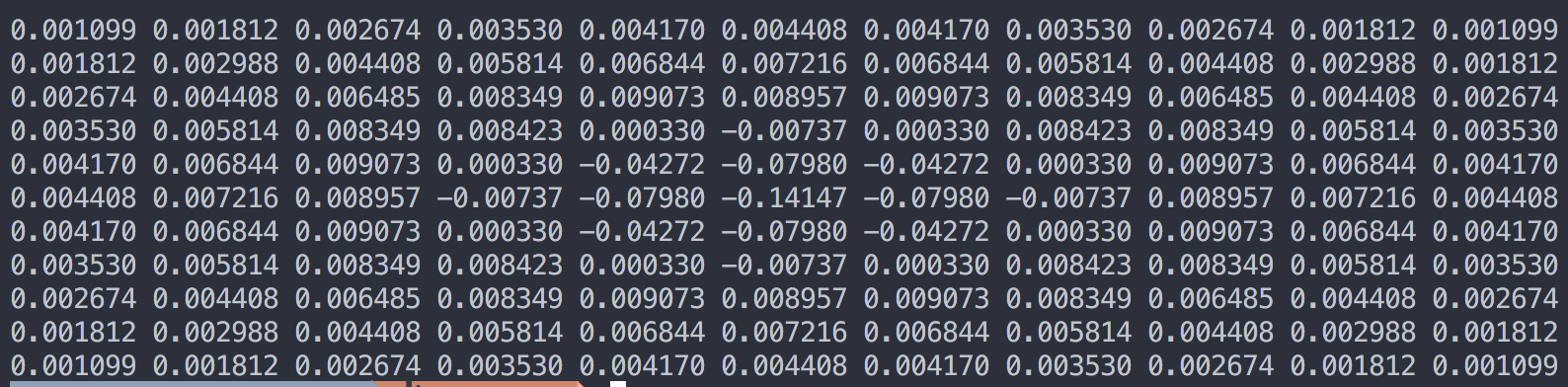




**sigma1: 1**

**sigma2: 3**

**kernel:**（因為排版 我沒有把全部小數點都印出來）

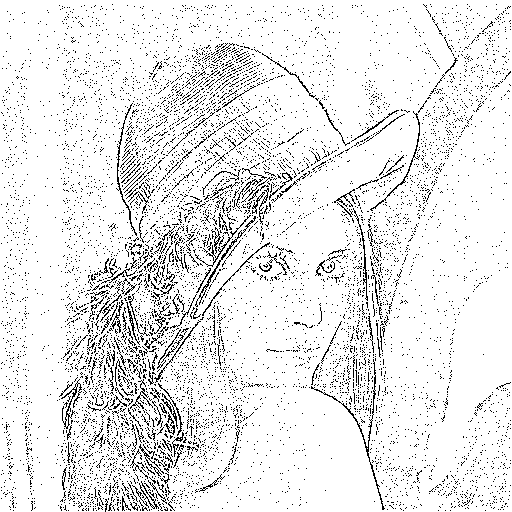
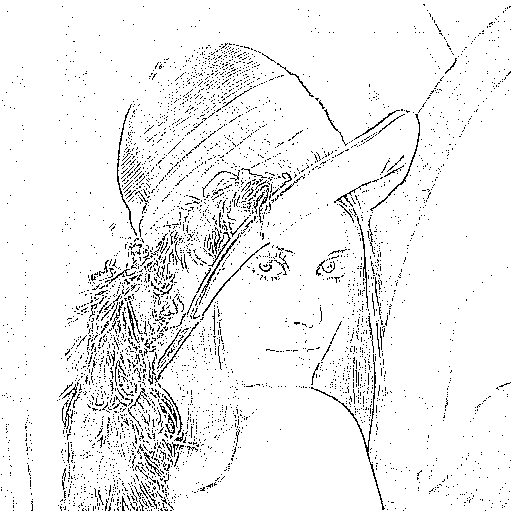


**threshold: 0.1**

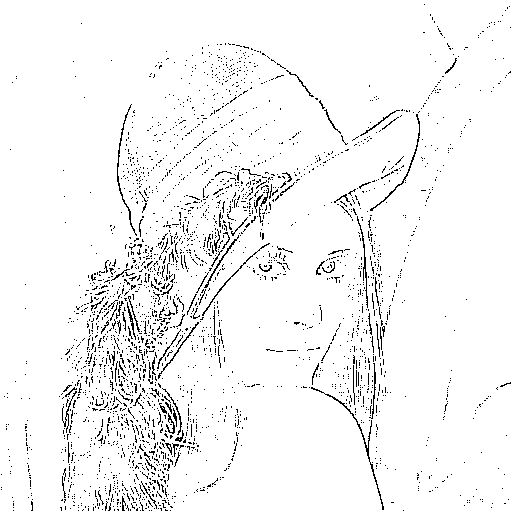
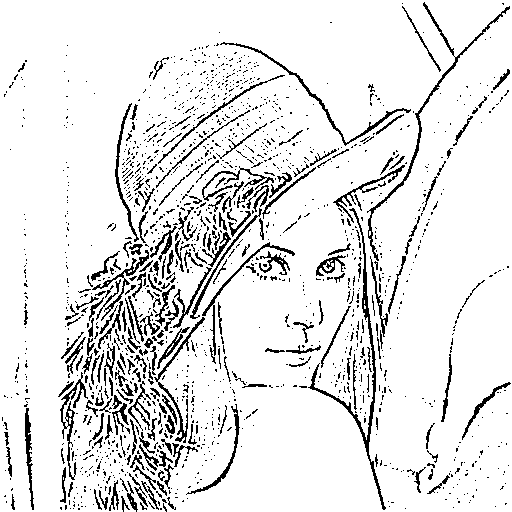
將input img與此kernel做cross-correlation，得到的值若大於threshold則output給值0，反之給值255

**Result:**

Laplace mask1: 20 Laplace mask2: 20

min-variance: 20 LoG: 30

DoG:0.1

