# Imagens Biomédicas – Lab04-Python

Tutores: André Arruda / Maíra Suzuka Kudo / Eric Rocha Santos

Professor: Matheus Cardoso Moraes

## MÁSCARAS PARA FILTRAGEM – (RESTAURAÇÃO E AGUÇAMENTO) EXERCÍCIOS:

1. **→ No Editor, faça um vetor amostra = [15 29 5 8 255 40 1 0 10];**
   a. **Crie uma função que ordene este vetor em ordem crescente de valores, e chame este novo vetor de amostraOrdenada.**

      **Obs.: pode usar → np.sort**

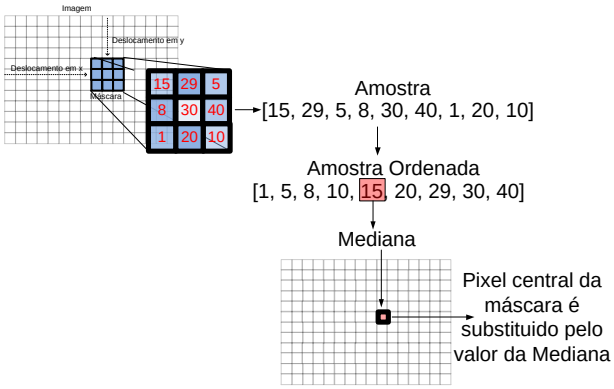
   b. **Crie uma variável mediana, para receber a mediana da amostra ordenada.**

2. **Leia, normalize e exiba a imagem IMRI = imread('...../TransversalMRI_salt-and-pepper.pgm');**

   **Adicionalmente crie uma matriz de zeros, chamada IMRIfiltrada, com o tamanho de IMRI.**

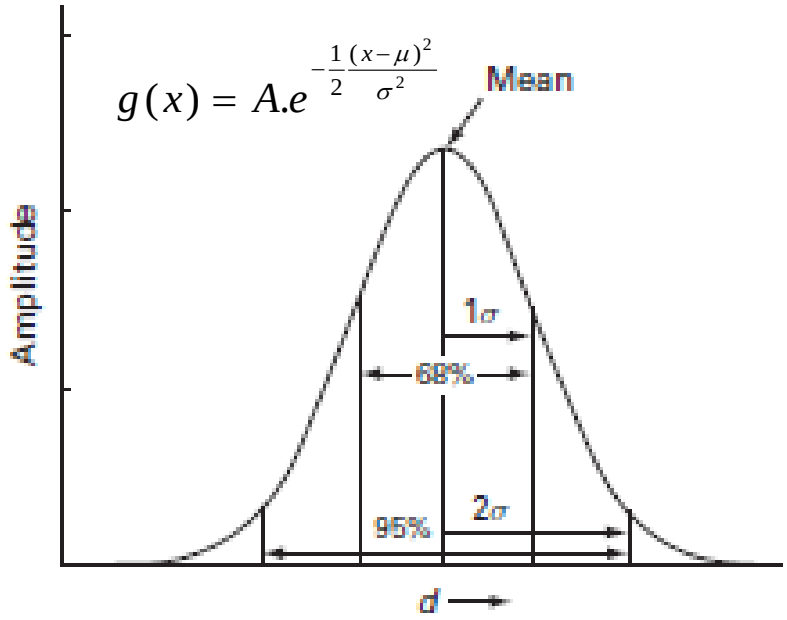
   a. **Crie uma função que use uma máscara 3x3 para varrer a imagem e retornar a mediana local para a posição correspondente em IMRIfiltrada.**

**obs.: usar as funções *"np.concatenate"* e "np.*sort*" podem ajudar)**



b. **Teste sua rotina acima usando a função "scipy.signal.medfilt2d".**

3. **→ No Editor, use a função *scipy.signal.gaussian* (*ou implemente da fórmula original*). Crie uma função gaussiana unidimensional (g(x)), que contenha 9 amostras ( x =0:8), e que possua média 4 e desvio padrão de 1**
   a. **Plote esta função, mostre e explique para o professor.**




   b. **Construa uma máscara 2D com valores seguindo distribuição Gaussiana. Uma função bidimensional Gaussiana pode ser construída a partir da convolução da máscara unidimensional com sua transposta, ou seja:**

**w_Gauss2D = g1* gtranspose1;**

**obs: no Python deve-se fazer o seguinte procedimento**

```
g1 = np.zeros((9,9), float)
g1[4,:] = g
gtranspose1 = np.transpose(g1)
w_Gauss2D = scipy.signal.convolve2d(g1,gtranspose1,'same')
```

c. **Exiba a imagem e verifique se está coerente com o esperado?**

```
plt.figure()
plt.title('w_Gauss2D ')
plt.imshow(w_Gauss2D , cmap='gray') # cmap='jet'
```

4. **Faça a convolução entre a imagem mamography.pgm e a máscara Gaussiana, e exiba a imagem filtrada. A mesma funcionou como filtro de suavização?**

   **obs1: pode usar a a função scipy.signal.convolve2d**

   **obs2: normalize a máscara w_Gauss2D chame de w_Gauss2DNormalizado , no qual o somatório desta, seja igual a 1. Assim, a convolução não acrescentara nível DC na imagem filtrada.**

   ```
   MamoFilt =
   scipy.signal.convolve2d(Mamo,w_Gauss2DNormalizado,'same')
   ```

5. **Usando o que aprendeu acima, faça uma função na qual o usuário indique apenas o desvio_padrão e média, e a mesma retorne uma máscara bidimensional com distribuição gaussiana. Seguindo o princípio de computar primeiro a função 1D para depois a 2D, o comprimento deve ser computado automaticamente em função da media, como abaixo.**

   **Criar Biblioteca → Abrir um arquivo dentro da pasta Aula04 e chamar de bibMascara.py**

   **Dentro desta biblioteca, criar a função: *fazerMascaraGauss2D* usando *def* da seguinte forma:**

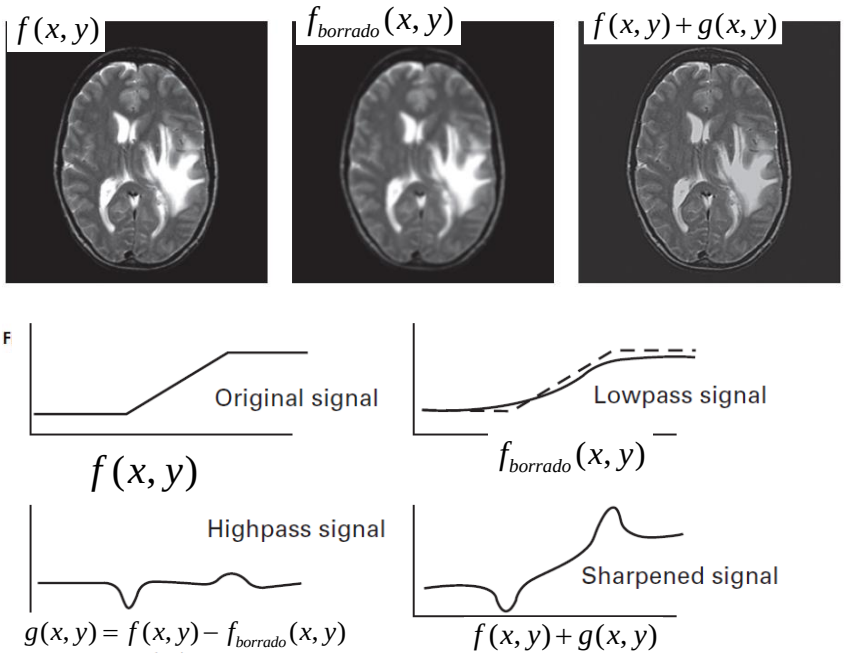
```
def fazerMascaraGauss2D(media, desvio):

    fazer lógica.....

    return wGauss2D
```

6. **Programa Principal: chame a função fazerMascaraGauss2D**

```
# import lib
import bibMascara
# chame a função com as entradas e saídas
w_Gauss2DNormalizado =
bibMascara.fazerMascaraGauss2D(media=4, desvio=1)
```
**obs1: teste outros valores de media e desvio**

a. **Exiba wGass2D, para verificar se a mesma esta retornando as dimensões corretas.**
b. **Faça convoluções entre a imagem e diferentes máscaras. O que acontece quando o tamanho da máscara é aumentado?**

7. **Faça uma operação de afiamento de bordas na imagem TransversalMRI2.pgm.**
   a. **Para criar a imagem borrada ($f_{borrado}$) figura abaixo: crie uma máscara gaussiana com média 7 e desvio 3.**




8. **Para a imagem Stent.pgm e Usando correlação, tire o gradiente em x, em y e compute seu módulo usando as máscaras Priwitt e Sobel abaixo. Exiba o**

**gradiente em x em y e seu módulo. Qual máscara extraiu melhor o Stent da imagem?**

$$w_x priwitt = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad w_y priwitt = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$w_x sobel = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad w_y sobel = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right] \qquad |\nabla f| = \sqrt{\left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2}$$

9. **Ainda para a imagem Stent.pgm e usando correlação, tire o Laplaciano usando as duas máscaras abaixo. Exiba os dois Laplacianos. Qual máscara extraiu melhor o Stent da imagem?**

$$\nabla^2 f = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\nabla^2 f = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

10. **(Desafio) Para a imagem Stent.pgm use o filtro mediana remover as marcas de alinhamento; posteriormente, além do Gradiente e Laplaciano, use as máscaras abaixo para extração do Stent. Qual máscara extraiu melhor o Stent da imagem?**

Operador de Kirsh

$$w_{01} kirsch = \begin{bmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{bmatrix} \quad w_{02} kirsch = \begin{bmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{bmatrix} \quad w_{03} kirsch = \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ 5 & 5 & 5 \end{bmatrix} \quad w_{04} kirsch = \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{bmatrix}$$

$$w_{05} kirsch = \begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix} \quad w_{06} kirsch = \begin{bmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{bmatrix} \quad w_{07} kirsch = \begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \quad w_{08} kirsch = \begin{bmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}$$

$$K = \max\left\{ (i * w_{01} kirsch), (i * w_{02} kirsch), ..., (i * w_{08} kirsch) \right\}$$

## Operador de Robinson

$$w_{01}robinson=\begin{bmatrix}1&0&-1\\2&0&-2\\1&0&-1\end{bmatrix}\quad w_{02}robinson=\begin{bmatrix}0&-1&-2\\1&0&-1\\2&1&0\end{bmatrix}\quad w_{03}robinson=\begin{bmatrix}-1&-2&-1\\0&0&0\\1&2&1\end{bmatrix}\quad w_{04}robinson=\begin{bmatrix}-2&-1&0\\-1&0&1\\0&1&2\end{bmatrix}$$

$$w_{05}robinson=\begin{bmatrix}-1&0&1\\-2&0&2\\-1&0&1\end{bmatrix}\quad w_{06}robinson=\begin{bmatrix}0&1&2\\-1&0&1\\-2&-1&0\end{bmatrix}\quad w_{07}robinson=\begin{bmatrix}1&2&1\\0&0&0\\-1&-2&-1\end{bmatrix}\quad w_{08}robinson=\begin{bmatrix}2&1&0\\1&0&-1\\0&-1&-2\end{bmatrix}$$

$$R=\max\left\{(i*w_{01}robinson),(i*w_{02}robinson),...,(i*w_{08}robinson)\right\}$$

## Operador de Frei-Chen

$$w_{01}\,freiChen=\begin{bmatrix}1&\sqrt{2}&1\\0&0&0\\-1&-\sqrt{2}&-1\end{bmatrix}\quad w_{02}\,freiChen=\begin{bmatrix}1&0&-1\\\sqrt{2}&0&-\sqrt{2}\\1&0&-1\end{bmatrix}\quad w_{03}\,freiChen=\begin{bmatrix}0&-1&\sqrt{2}\\1&0&-1\\-\sqrt{2}&1&0\end{bmatrix}$$

$$w_{04}\,freiChen=\begin{bmatrix}\sqrt{2}&-1&0\\-1&0&1\\0&1&-\sqrt{2}\end{bmatrix}\quad w_{05}\,freiChen=\begin{bmatrix}0&1&0\\-1&0&-1\\0&1&0\end{bmatrix}\quad w_{06}\,freiChen=\begin{bmatrix}-1&0&1\\0&0&0\\1&0&-1\end{bmatrix}$$

$$w_{07}\,freiChen=\begin{bmatrix}1&-2&1\\-2&4&-2\\1&-2&1\end{bmatrix}\quad w_{08}\,freiChen=\begin{bmatrix}-2&1&-2\\1&4&1\\-2&1&-2\end{bmatrix}\quad w_{09}\,freiChen=\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$$FC=\left\{\left((p_{01}.w_{01}\,freiChen)*i\right)+...+\left((p_{09}.w_{09}\,freiChen)*i\right)\right\}$$