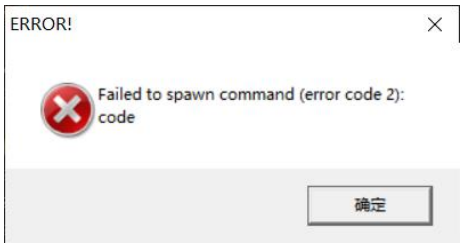
# 图像处理作业一

软工一班
陈仟雅
3017218053
2019.11.12

注：由于 texlive 多次打开 vscode 失败，卸载重装也无效，所以本次报告使用 word 编写。




## 题目一

使用 matlab 写一个函数，img = generateFigure(imgW,imgH)，其作用为产生一幅的彩色图像，图像中用红色显示[0,2*pi]的正弦波，用绿色显示[0,2*pi]的余弦波，蓝色显示[0,2*pi]的 y=x^2 图像

## 分析

题目要求使用 matlab 绘制出红色的正弦波，绿色的余弦波，蓝色的 y=x^2 图像，通过查阅资料得知 plot 函数可以帮助绘图，通过‘r’可以将曲线变红，‘g’可以将曲线变绿，‘b’可以将曲线变蓝，所以综合得到了这样一条语句——plot(x,y1,'r',x,y2,'g',x,y3,'b')，其中 x 设置范围为 0 到 2pi，y1，y2，y3 分别是三条曲线的函数。

又通过查阅资料得知，print 函数可用于输出图片，与此同时还可设置图片格式，于是将图片设置为 png 格式输出。

## 代码段

```
function [I] = generateFigure(imgW,imgH)
%UNTITLED 此处显示有关此函数的摘要
%    此处显示详细说明
%    ‘r’可以将曲线变红，‘g’可以将曲线变绿，‘b’可以将曲线变蓝
%    print 打印出图片
```

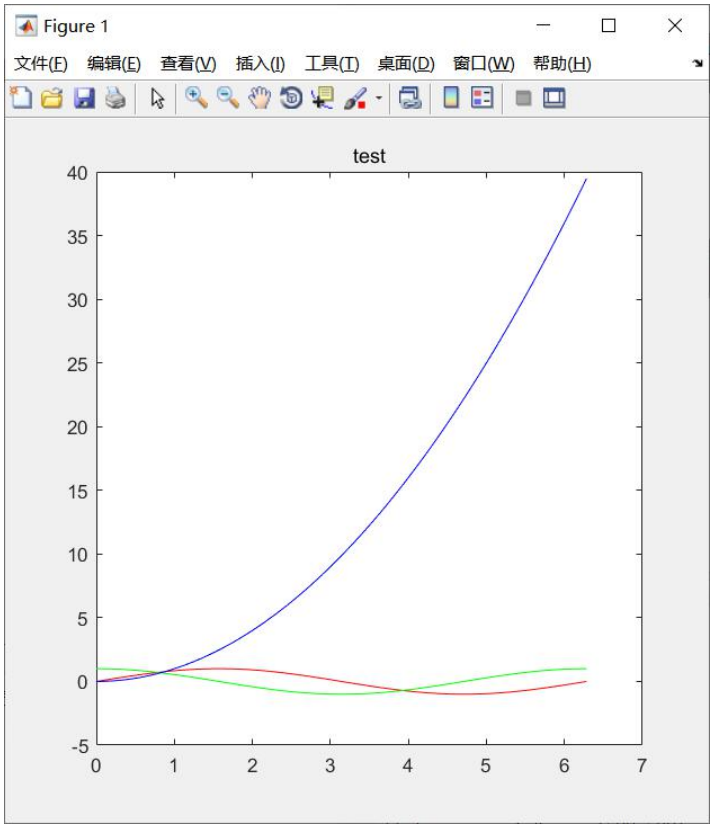
```
x = 0:pi/100:2*pi;
y1 = sin(x);
y2 = cos(x);
y3 = x.^2;
figure(1),plot(x,y1, 'r' ,x,y2, 'g' ,x,y3, 'b' ),title('test');
set (1,'position',[100,100,imgW,imgH] );   % 设置图像边距和大小
print(1, '-dpng', 'test');
I = imread('test.png');
end
```

## 测试用例

```
I = generateFigure(500,500);
```

```
>> I = generateFigure(500,500);
```

## 测试结果

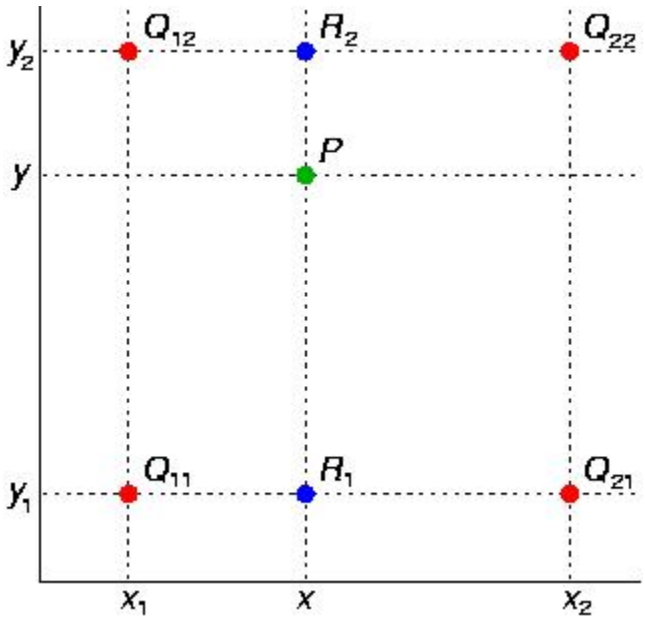


## 题目二

不使用 for 循环，实现 bilinear interpolation（但由于学艺不精，最终没能实现不使用循环的双线性插值）

## 分析

已知 Q12，Q22，Q11，Q21，但是要插值的点为 P 点，这就要用双线性插值了，首先在 x 轴方向上，对 R1 和 R2 两个点进行插值，这个很简单，然后根据 R1 和 R2 对 P 点进行插值，这就是所谓的双线性插值。




假如我们想得到未知函数 f 在点 P=（x，y）的值，假设我们已知函数 f 在 $Q_{11}$ = ( x1, y1), $Q_{12}$ =( x1, y2 ), $Q_{21}$ = ( x2, y1), 及 $Q_{22}$ = ( x2, y2) 四个点的值。首先在 x 方向进行线性插值，得到

$$f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}) \quad \text{Where} \quad R_1 = (x, y_1),$$

$$f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}) \quad \text{Where} \quad R_2 = (x, y_2).$$

然后在 y 方向进行线性插值，得到

$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2).$$

这样就得到所要的结果 f(x, y)

$$f(x,y) \approx \frac{f(Q_{11})}{(x_2-x_1)(y_2-y_1)}(x_2-x)(y_2-y)+\frac{f(Q_{21})}{(x_2-x_1)(y_2-y_1)}(x-x_1)(y_2-y)$$
$$+\frac{f(Q_{12})}{(x_2-x_1)(y_2-y_1)}(x_2-x)(y-y_1)+\frac{f(Q_{22})}{(x_2-x_1)(y_2-y_1)}(x-x_1)(y-y_1).$$

## 代码段

```
function [] = pictures()
%UNTITLED 此处显示有关此函数的摘要
%   此处显示详细说明
%   双线性插值实现
%   f(x,y)=w1*p1+w2*p2+w3*p3+w4*p4 p1,p2,p3,p4 是与(x,y)最近的四个像素点
%   w1,w2,w3,w4 分别是对应的权重
%   src 是原图像（M*N）,dst 是目标图像（P*Q）
%   缩放因子是 M/P,N/Q
%   双线性内插本质上就是从 x 方向和 y 方向分别作了一次线性插值
src = imread('test.png');
% 得到原图像的大小
[M,N,Z] = size(src);
% 设置目标图像的大小是 300*300*3
dst = zeros(300,300,3);
[P,Q,H] = size(dst);
scaler_x = M/P;
scaler_y = N/Q;
% 插值实现
for i=1:P
    for j=1:Q
        % 后向映射得到在原图上的大概坐标
        % 注意一般是浮点类型
        x = i*scaler_x;
        y = j*scaler_y;
        % 取得与(x,y)距离最近的 4 个像素点
        X1 = floor(x);
        X2 = ceil(x);
        Y1 = floor(y);
        Y2 = ceil(y);
        P1_1 = src(X1,Y1,:);
        P1_2 = src(X1,Y2,:);
        P2_1 = src(X2,Y1,:);
        P2_2 = src(X2,Y2,:);
```

```
            % 先在 x 轴方向进行二次线性内插
            fx1 = (X2-x)/(X2-X1).*P1_1+(x-X1)/(X2-X1).*P2_1;
            fx2 = (X2-x)/(X2-X1).*P1_2+(x-X1)/(X2-X1).*P2_2;
            % 在 y 方向进行一次线性内插
            f = (Y2-y)/(Y2-Y1).*fx1+(y-Y1)/(Y2-Y1).*fx2;
            dst(i,j,:) = f;
        end
    end
    dst = im2uint8(mat2gray(dst));
    figure();
    imshow(src)
    title('原图像')
    figure();
    imshow(dst)
    title('目标图像 300*300*3')
```

## 测试结果