# Q1：时域与频域是什么？

**时域**故名思议就是随着时间的推移，我们所能直观感受的东西或事物，比如说音乐，我们听到动听的音乐，这是在时域上发生的事情。

而对于演奏者来说音乐是一些固定的音符，我们听到的音乐在**频域**内是一个永恒的音符，音符的个数是有限且固定的，但可以组合出无限多的乐曲。

傅立叶也告诉我们，任何周期函数都可以看作不同振幅，不同相位的正弦波的叠加。就像用音符组合出音乐一样。

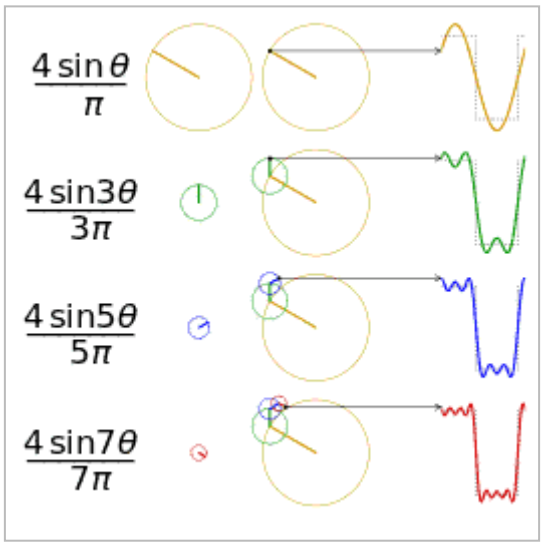
贯穿时域和频域的方法之一，就是傅立叶分析，傅立叶分析又分为两个部分：傅立叶级数和傅立叶变换。

# Q2：傅立叶级数是啥？

**傅立叶级数**指出任何**周期函数**都可以看作不同振幅，不同相位的正弦波的叠加。

> 对比**傅立叶变换**：傅立叶变换指出**非周期的函数**（函数曲线下的面积是有限的）也可以用正弦或余弦乘以加权函数的积分来表示。

说的过程大概是这样子的：

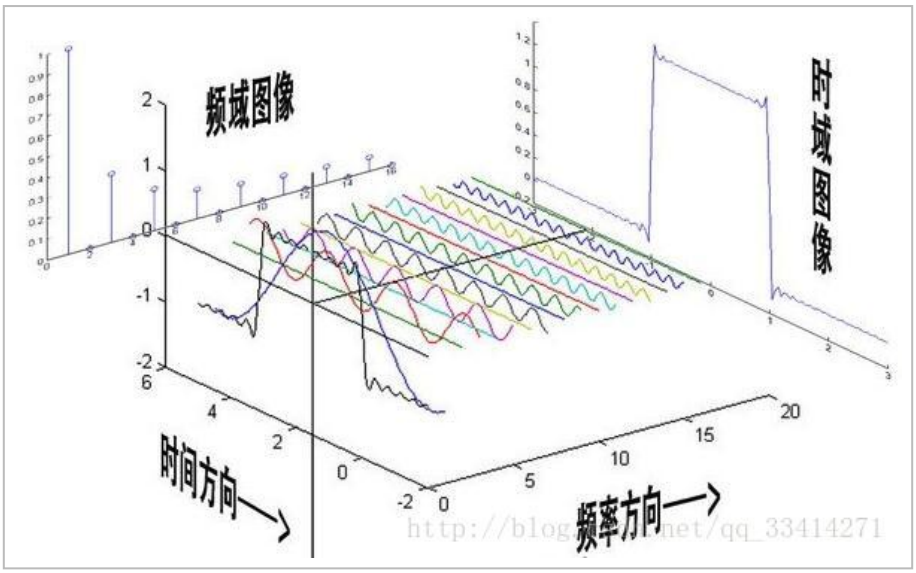



在傅立叶级数中要介绍两个概念：频谱（幅度谱），相位谱。

有了这两个东西之后我们就可以更容易理解把周期函数拆分为各个正弦函数叠加的过程了。

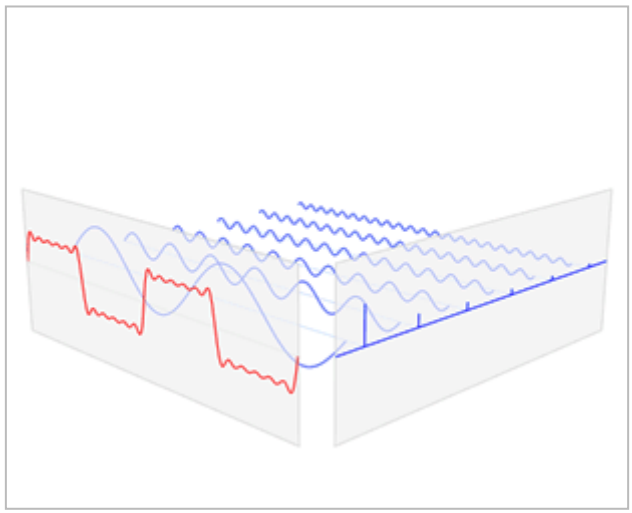
## 频谱（幅度谱）

之前我们提到了时域和频域，从不同的 “域” 来看可能会产生很不一样的效果，到底有多不一样呢？先上个图来看一下。




可以看出，从时域来看，我们会看到一个近似为矩形的波，而我们知道这个矩形的波可以被差分为一些正弦波的叠加。

而从频域方向来看，我们就看到了**每一个正弦波的幅值**，可以发现，在频谱中，偶数项的振幅都是 0，也就对应了图中的彩色直线。振幅为 0 的正弦波。



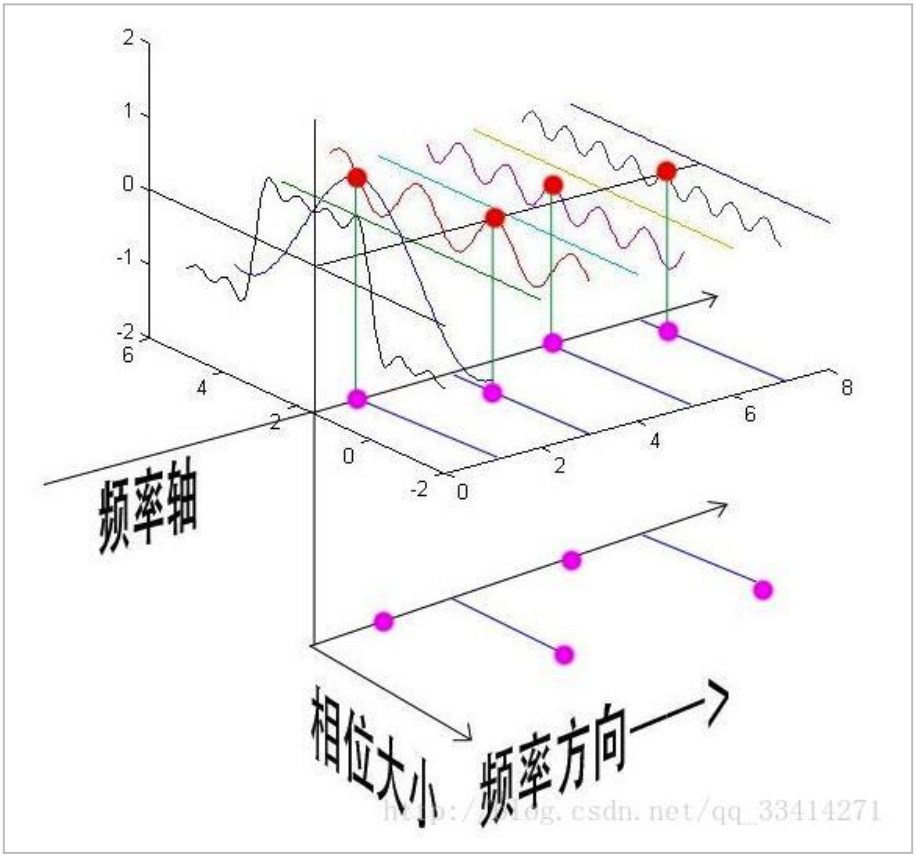
上图也动态展示了频域图像，应该可以加深理解。

## 相位谱

频谱只代表了一个正弦函数的幅值，而要准确描述一个正弦函数，我们不仅需要幅值，还需要相位，不同相位决定了波的位置，所以对于频域分析，仅仅有频谱（振幅谱）是不够的，我们还需要一个相位谱。

那相位谱在哪呢？

先上个图



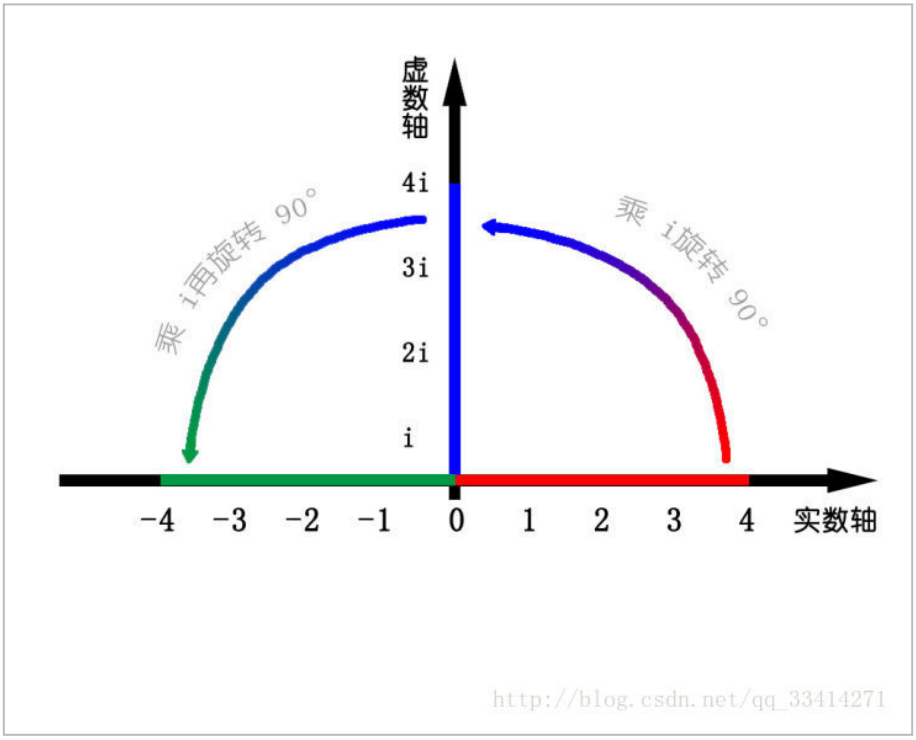

投影点我们用粉色点来表示，红色的点表示离正弦函数频率轴最近的一个峰值，而相位差就是粉色点和红色点水平距离除以周期

将相位差画到一个坐标轴上就形成了相位谱，如上图所示。

# Q3：什么是傅立叶变换？

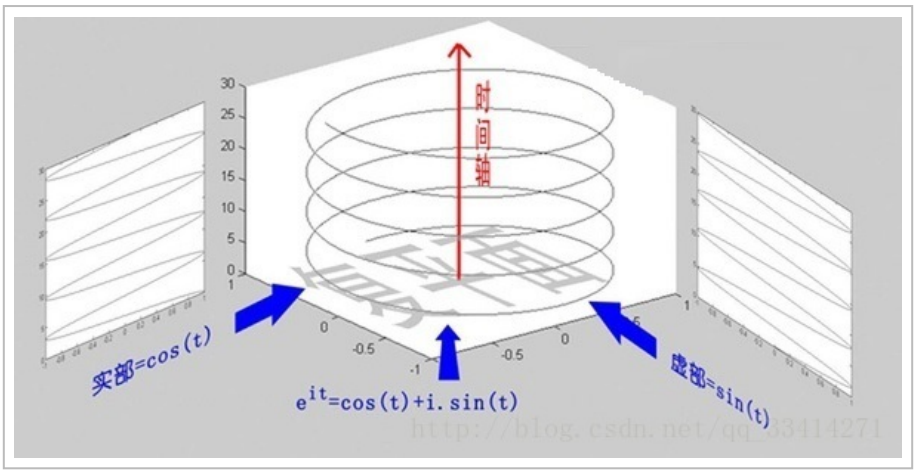
傅里叶级数，在时域是一个周期且连续的函数，而在频域是一个非周期离散的函数。
傅里叶变换，则是将一个时域非周期的连续信号，转换为一个在频域非周期的**连续信号**。

让我们先从复数说起，下面是一个复数的一个表达形式，可以看出乘以 i 的效果是将数值逆时针旋转了 90 度




数轴与虚数轴共同构成了一个复数的平面，也称复平面。这样我们就了解到，乘虚数 i 的一个功能——旋转。

**欧拉公式**将正弦波统一成了简单的指数形式，我们来看看图像上的涵义：

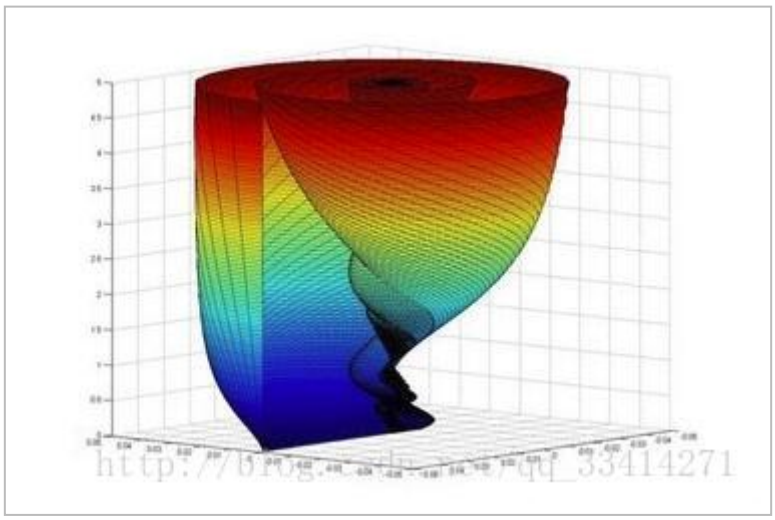



欧拉公式所描绘的，是一个随着时间变化，在复平面上做圆周运动的点，随着时间的改变，在时间轴上就成了一条螺旋线。如果

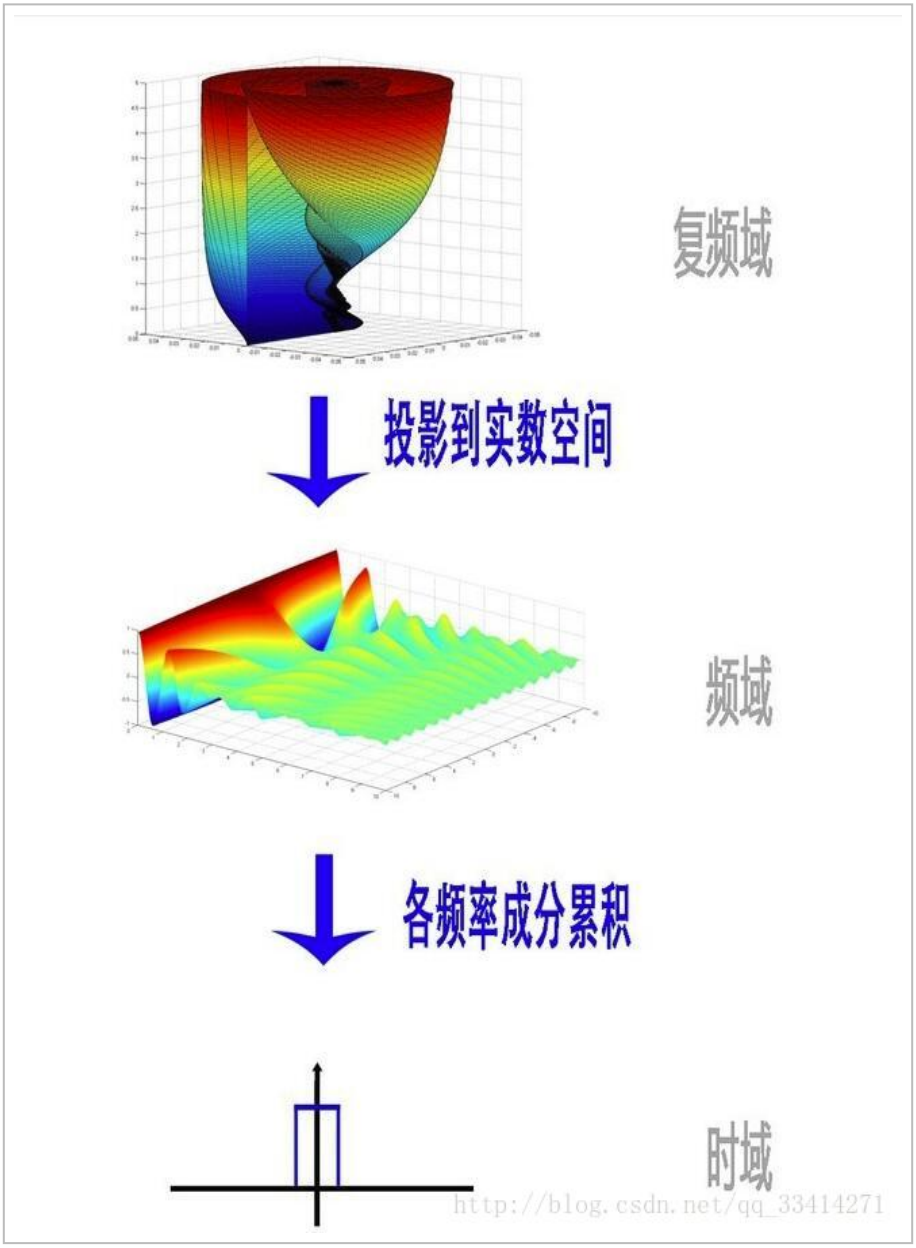
只看它的实数部分，也就是螺旋线在左侧的投影，就是一个最基础的余弦函数。而右侧的投影则是一个正弦函数。

**欧拉公式告诉我们：正弦波的叠加，也可以理解为螺旋线的叠加在实数空间的投影。**

根据欧拉公式里面的复平面，我们可以得到单个矩形波形成的螺旋图如下图所示：




如果你认真去看，海螺图上的每一条螺旋线都是可以清楚的看到的，每一条螺旋线都有着不同的振幅（旋转半径），频率（旋转周期）以及相位。而将所有螺旋线连成平面，就是这幅海螺图了。




上图展示了将海螺图投影到实数空间就形成了傅立叶变换的连续非周期的连续的曲线，此曲线在时域上就表现为一个矩形波的形式。

# Q4：傅立叶变换有什么用？

如果地震波可被分解，找出不同的振幅和速度，那么我们可以针对地震的特定振幅和速度设计对应的抗震建筑物。

如果声波可被分解成低音和高音，我们就可以放大我们关心的部分，缩小我们不关心的部分。

> “比如你喜欢小提琴，那便可以提高高音部分，隐去低音部分。
> 如果你喜欢低音贝斯，那么你就可以提高低音部分，隐去高音部分。”

如果计算机数据可以用震荡波形表示，且其中包含可忽略的数据，那么就可以用傅里叶变换滤去不重要的数据。这在数据科学中被叫为 “数据滤波器” 。

如果是收音机的无线电波，那么我们就可以收听到特定频率的广播。

# Matlab 上的傅立叶变化

## 1. 傅立叶的频谱图和相位谱

首先是一维的傅立叶变换，我们生成一个矩形波为例
用下面的代码可以生成一个矩形波，其中 t 为时间轴，width 为矩形波的宽度

```
ft=rectpuls(t,width);
```

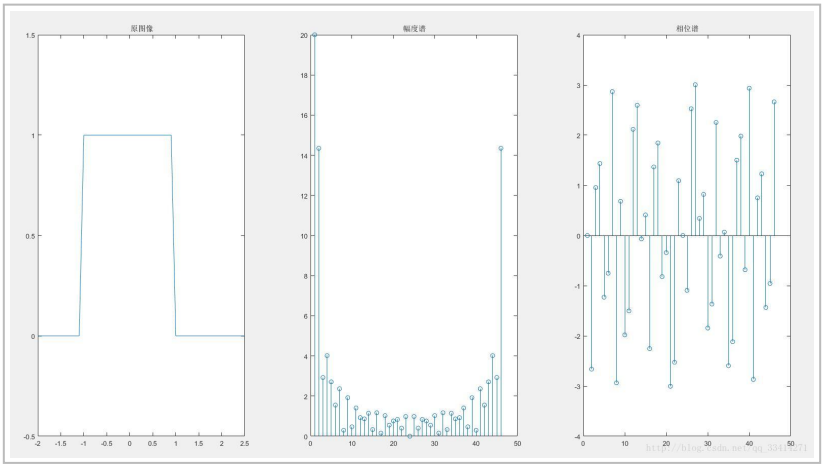
利用 **fft()** 函数可以直接进行变化，**real()** 函数可以求出实部，**imag()** 得到虚部

```
f1=fft(ft);
r1=real(f1);            %实部
i1=imag(f1);            %虚部
```

振幅为对实部和虚部分别平方后加和的平方根（有点绕，可以不用管它，我只是不想写公式），相位是虚部除以实部的反正切，即 arctan(虚部 / 实部)，这个可以用 **angle()** 函数实现

```
margin1=sqrt(r1.^2+i1.^2);          %幅度谱
phase1=angle(f1);       %相位谱
```

好了，看看效果吧




效果和最开始讲的直观理解一致

完整代码

```
%% 一维傅立叶变换
%生成矩形波
width=2;
t=-2:0.1:2.5;
ft=rectpuls(t,width);
figure
subplot(1,3,1),plot(t,ft),title('原图像');
grid on;
ylim([-0.5 1.5])
grid off;
%傅立叶变换
f1=fft(ft);
r1=real(f1);            %实部
i1=imag(f1);            %虚部
margin1=sqrt(r1.^2+i1.^2);       %幅度谱
phase1=angle(f1);      %相位谱
subplot(1,3,2),stem(margin1),title('幅度谱');
subplot(1,3,3),stem(phase1),title('相位谱');
```

一维傅立叶是对波形等一维信号的变换（如语音等），二维傅立叶变化是对二维信号的变换（如图像），那我们就看一下二维傅立叶对图像的变换后的幅度谱和相位谱吧。

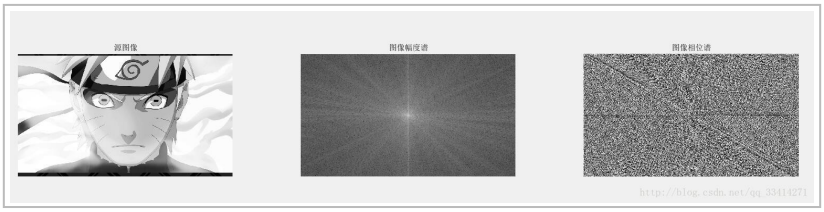
基本操作和调用的函数和一维的情况差不多

代码里有注释，就直接上代码和效果吧

```
%% 二维傅立叶变换
%读取图片
I=imread('Naruto.jpg');
img=rgb2gray(I);

%幅值图和相位谱
%进行傅立叶变换
f=fft2(img);            %傅里叶变换
f=fftshift(f);          %使图像对称,中心化
r=real(f);              %图像频域实部
i=imag(f);              %图像频域虚部
margin=log(sqrt(r.^2+i.^2));        %图像幅度谱，加log便于显示
phase=real(angle(f)*180/pi);        %图像相位谱
figure
subplot(1,3,1),imshow(img),title('源图像');
subplot(1,3,2),imshow(margin,[]),title('图像幅度谱');
subplot(1,3,3),imshow(phase,[]),title('图像相位谱');
```

效果图




可以看出二维傅立叶变换后效果没有一维那么直观了，但道理都是一样的。

## 2. 傅立叶变换的应用——图像的低通滤波

过程大概是这个样子的

对图像进行傅里叶变换（FFT），得到频谱；
用理想低通滤波器对频谱滤波；
对滤波后的频谱进行反傅里叶变换（IFFT），得到滤波后图像。

> 理想低通滤波就是利用一个函数将频率大于某个阈值的频率设为 0，它的函数表达式是这样子的

$$H(u,v)=\begin{cases}1, D(u,v)\le D_0\\ 0, D(u,v)>D_0\end{cases}$$

D(u,v) 的计算方式也就是两点间的距离，很简单就能得到：

$$D(u,v)=\sqrt{\left(u-\frac{P}{2}\right)^2+\left(v-\frac{Q}{2}\right)^2}$$

所以我们就很容易得到此滤波方法的程序：

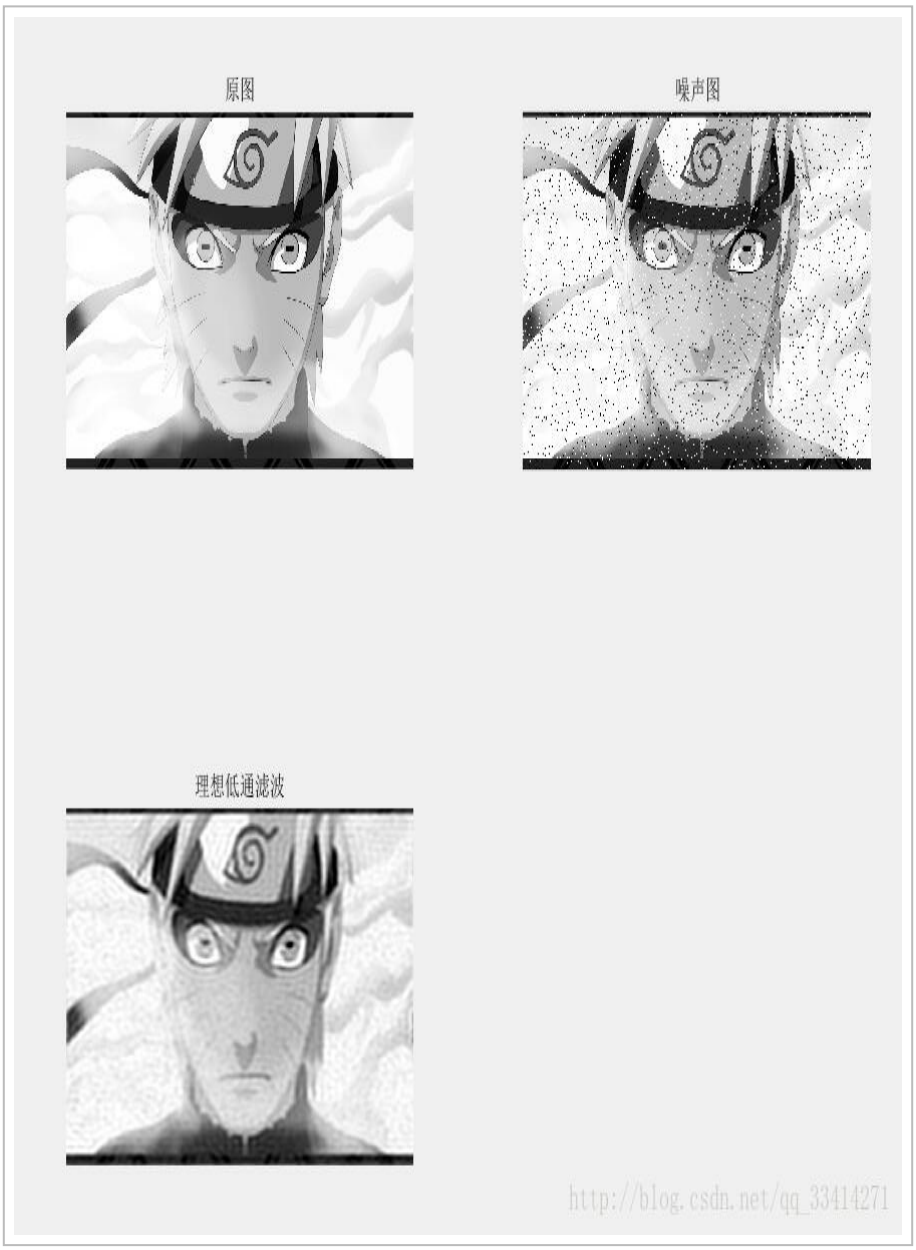
```
%% 傅里叶变换的低通滤波
%低通滤波选用理想低通滤波方式
% d0 是阈值，可以修改，初步设定为50

img_origin=imread('Naruto.jpg');
img_origin=rgb2gray(img_origin);
d0=50;  %阈值
img_noise=imnoise(img_origin,'salt'); % 加椒盐噪声
%img_noise=imnoise(img_origin,'gaussian'); % 加高斯噪声
img_f=fftshift(fft2(double(img_noise)));  %傅里叶变换得到频谱
[m, n]=size(img_f);
m_mid=fix(m/2);  %是不是可以有其他取整方式?
n_mid=fix(n/2);
img_lpf=zeros(m,n);
for i=m:-1:1
    for j=n:-1:1
        d=sqrt((i-m_mid)^2+(j-n_mid)^2);    %理想低通滤波，求距离
        if d<=d0
            h(i,j)=1;
        else
            h(i,j)=0;
        end
        img_lpf(i,j)=h(i,j)*img_f(i,j);
    end
end

img_lpf=ifftshift(img_lpf);     %反傅里叶变换
img_lpf=uint8(real(ifft2(img_lpf)));  %取实数部分

subplot(2,2,1);imshow(img_origin);title('原图');
subplot(2,2,2);imshow(img_noise);title('噪声图');
subplot(2,2,3);imshow(img_lpf);title('理想低通滤波');
```

效果为：




可以看出经过滤波后噪声显然没有那么明显了

好了写到这里就差不多了，总结一下，我们首先直观的理解了傅立叶级数和傅立叶变换，傅立叶级数是说周期性变换的函数可以用有限个正弦波叠加而来，傅立叶变换说非周期变换的函数也可以用连续的正弦波来模拟，也看了在复平面、频域和时域上傅立叶变换的效果（顺便了解了复数的意义和欧拉公式），最后我们用 matlab 实现了实现和展示了频谱图和相位谱，还实现了傅立叶变换的一个实例——理想低通滤波，其中傅立叶变换是基础。

最后的最后，我们在以上的文章中并没有展示傅立叶级数和傅立叶变化本身公式是怎么样的，要是还想理解公式和原理，在傅立叶上做更多应用的筒子们可以**留言点赞关注**哦，给我更新下一波的动力！！！


参考文献

韩 昊 : http://blog.jobbole.com/70549/ (http://blog.jobbole.com/70549/)

https://www.zhihu.com/question/23234701/answer/26017000 (https://www.zhihu.com/question/23234701/answer/26017000)

https://en.wikipedia.org/wiki/File:Fourier_series_square_wave_circles_animation.gif (https://en.wikipedia.org/wiki/File:Fourier_series_square_wave_circles_animation.gif)

非常感谢原作者提供的图以及维基百科上的图片

http://blog.csdn.net/tianrolin/article/details/44084317 (http://blog.csdn.net/tianrolin/article/details/44084317)

http://www.tk4479.net/ytang_/article/details/75451934 (http://www.tk4479.net/ytang_/article/details/75451934)


---

全文完


本文由 简悦 SimpRead (http://ksria.com/simpread) 优化，用以提升阅读体验。