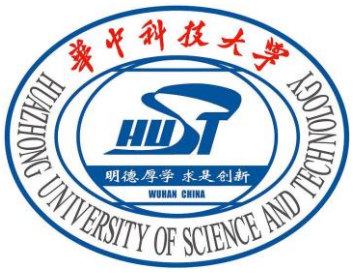


华中科技大学
生物医学数字信号处理实验报告
实验 2 傅里叶级数与傅里叶变换

学院	工程科学学院
班级	工程科学学院（生医）1701 班
姓名	汪能志
学号	U201713082
指导老师	全廷伟

2020 年 9 月 3 日

目录

1. 实验目的

用傅里叶级数逼近方波信号；
利用傅里叶变换从钙信号中重建出动作电位序列；
理解并实现 fft，ifft，fft2 和 ifft2 这四个函数的功能。

2. 方波及其傅里叶级数逼近

2.1. 实验原理

对于离散信号，可以使用累加的方法来近似计算积分：

$$\int_a^b f(x)\mathrm{d}x = \lim_{n\to\infty}\sum_{i=0}^{n}\frac{b-a}{n} f(a+\frac{b-a}{n}i) \tag{2-1}$$

则，对于傅里叶级数：

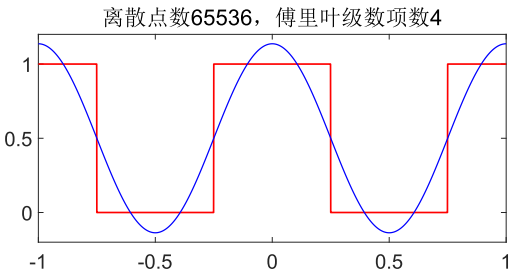
$$\hat{f}(k) = \frac{1}{T}\int_T f(t)e^{-jk\omega}t\mathrm{d}t = \lim_{n\to\infty}\sum_{i=0}^{n}\frac{1}{n} f(\frac{T}{N}i)e^{-jk\omega\frac{T}{n}i} \tag{2-2}$$

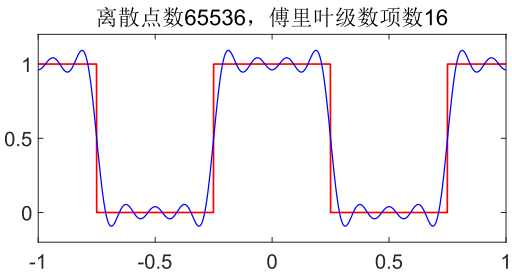
2.2. MATLAB 实现

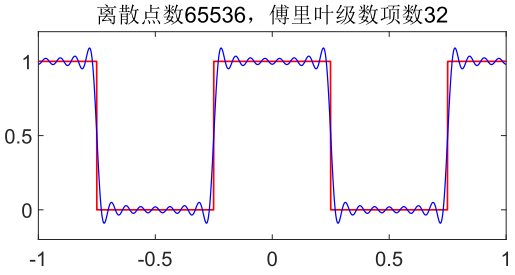
```
% 生成离散方波信号
T = 1;                                  % 周期
w = (2 * pi) / T;
num_sample = 65536;                     % 采样点数
t = linspace(-1, 1, num_sample);
x = double(cos(w * t) >= 0);

num_k = 128;
% 计算傅里叶级数的正交基
base = Get_Base(num_k, t, T);
% 计算傅里叶级数的系数
spec = FS(x, base, T);
% 计算重建信号
recover_x = IFS(spec, base, T);
figure(1)
plot(t, x, '-r', 'LineWidth', 2)
hold on
plot(t, recover_x, '-b', 'LineWidth', 1.5)
hold off
ylim([-0.2 1.2])
title_str = sprintf('离散点数%d，傅里叶级数项数%d', num_sample, num_k);
title(title_str)
```

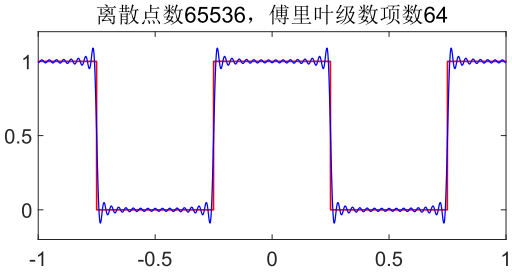
```
function fourier_series = FS(f, base, T)
% 计算傅里叶级数
fourier_series = Approx_Inner(f, base, T) ./ sqrt(T);
end

function recover_signal = IFS(fourier_series, base, T)
% 根据傅里叶级数逼近原信号
recover_signal = real(fourier_series * base .* sqrt(T));
end

function base = Get_Base(num_k, t, T)
% 计算正交基
K = -ceil(num_k / 2):1:ceil(num_k / 2);
base = exp(1j .* 2 .* pi .* (1 / T) .* Outer(K, t)) ./ sqrt(T);
end

function z = Approx_Inner(f, g, T)
% 对于给定的函数（值）f 和 g，以及采样点序列 T，近似计算 f 和 g 的内积
num_sample = numel(f);
z = (f * conj(g')) .* (T ./ num_sample);
end

function z = Outer(a, b)
% a = [a1, …, am] and b = [b1, …, bn]
% result = [
% a1 * b1, a1 * b2, …, a1 * bn;
% a2 * b1, a2 * b2 ,…, a2 * bn;
% …
% am * b1, am * b2, …, am * bn;
% ]
% 效果和 numpy.outer 相同
M = numel(a);
N = numel(b);
z = zeros(M, N);
for i = 1:M
    for j = 1:N
        z(i, j) = a(i) .* b(j);
    end
end
end
```

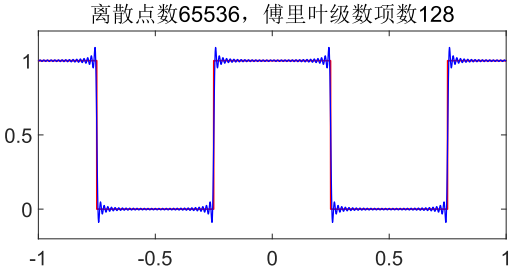
2.3. 实验结果

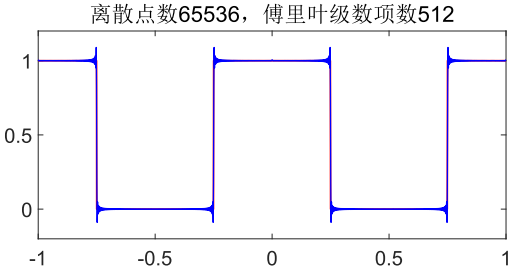

离散点数65536，傅里叶级数项数4

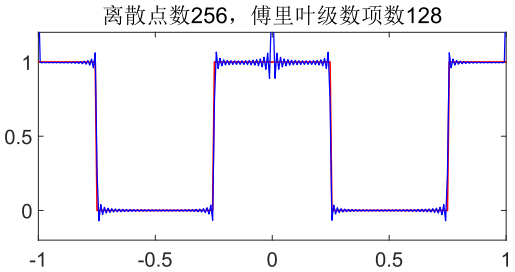

离散点数65536，傅里叶级数项数16

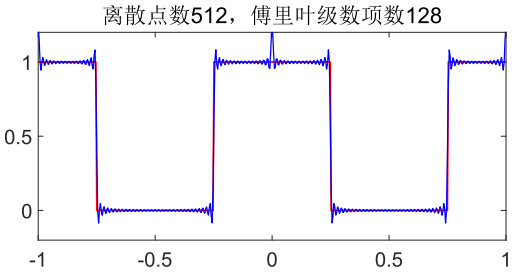

离散点数65536，傅里叶级数项数32

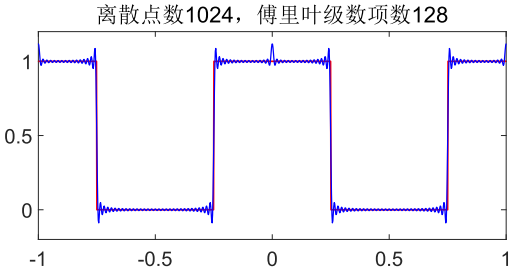

离散点数65536，傅里叶级数项数64

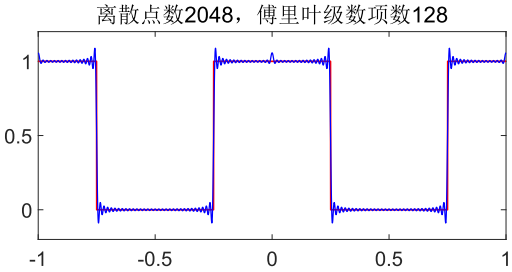

离散点数65536，傅里叶级数项数128

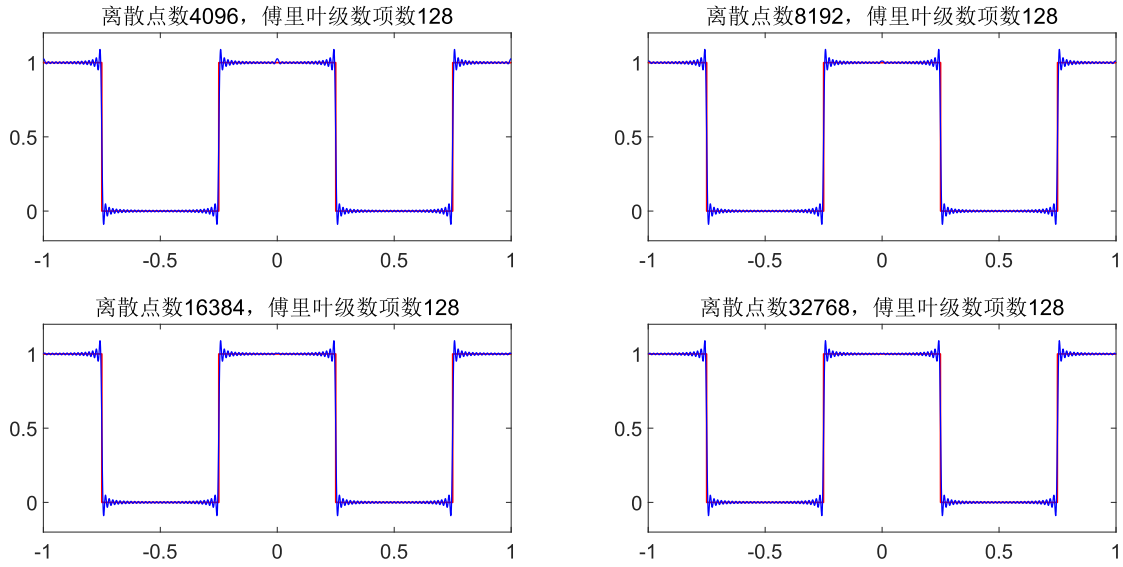

离散点数65536，傅里叶级数项数512


离散点数256，傅里叶级数项数128


离散点数512，傅里叶级数项数128


离散点数1024，傅里叶级数项数128


离散点数2048，傅里叶级数项数128



很容易发现，在保持离散点数不变的情况下，随着傅里叶级数的项数的增加，级数逼近的结果越来越接近方波。但是，由于需要拟合的信号中存在阶跃，因此吉布斯现象永远存在。

由于实验中是使用累加求和的方式近似计算积分，因此离散点越多，则级数逼近的信号越接近方波。在离散点数较少时，在非信号的非阶跃处也出现了类似吉布斯现象的情况。

3. 傅里叶变换与卷积

3.1. 实验原理

定义$\otimes$为循环卷积，$\hat{x}$、$\hat{h}$和$\hat{y}$为补零到相同长度后的信号。则循环卷积有以下性质：

$$\hat{y} = \hat{x} \otimes \hat{h} \Leftrightarrow \mathrm{DFT}(\hat{y}) = \mathrm{DFT}(\hat{x}) \times \mathrm{DFT}(\hat{h}) \tag{2-3}$$

该性质也可用于计算反卷积。

3.2. MATLAB 实现

3.2.1. 傅里叶变换卷积

```
function y = conv_ft(x, h)
% 傅里叶变换卷积
len_x = size(x, 1);
len_h = size(h, 1);
len_y = len_x + len_h - 1;

% 对卷积模板和信号进行补零
x_circle = zeros(len_y, 1);
x_circle(1:len_x) = x;
h_circle = zeros(len_y, 1);
h_circle(1:len_h) = h;

Fy = fft(x_circle) .* fft(h_circle);
y = real(ifft(Fy));
```

```
end
```

3.2.2. 傅里叶变换反卷积

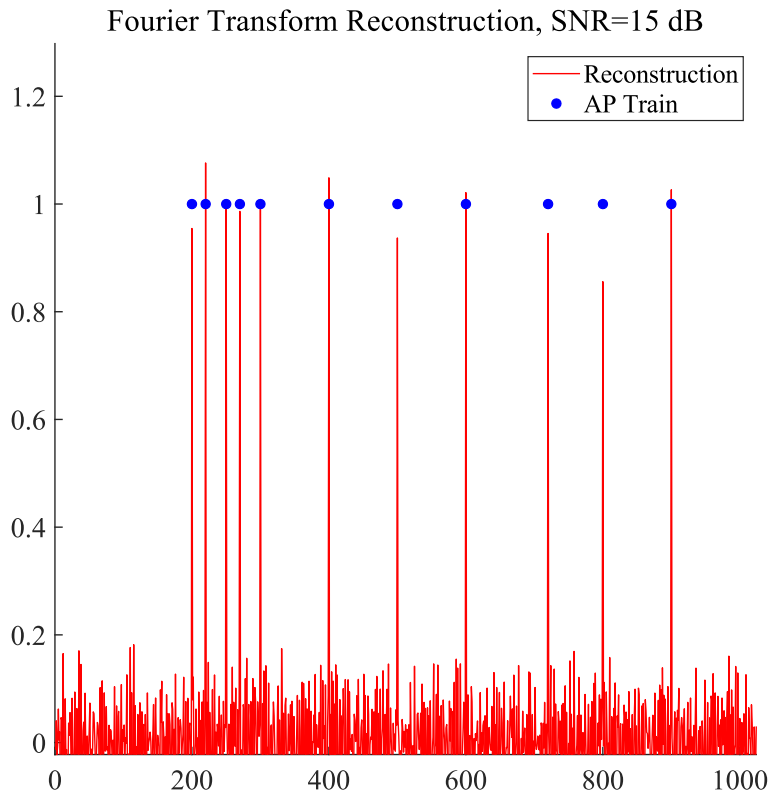
```
function x = deconv_dft(y, h)
% 傅里叶变换反卷积
len_y = size(y, 1);
len_h = size(h, 1);
len_x = len_y - len_h + 1;

% 对卷积模板进行补零
h_circle = zeros(len_y, 1);
h_circle(1:len_h) = h;

Fx = fft(y) ./ fft(h_circle);
x = real(ifft(Fx));
x = x(1:len_x);
end
```

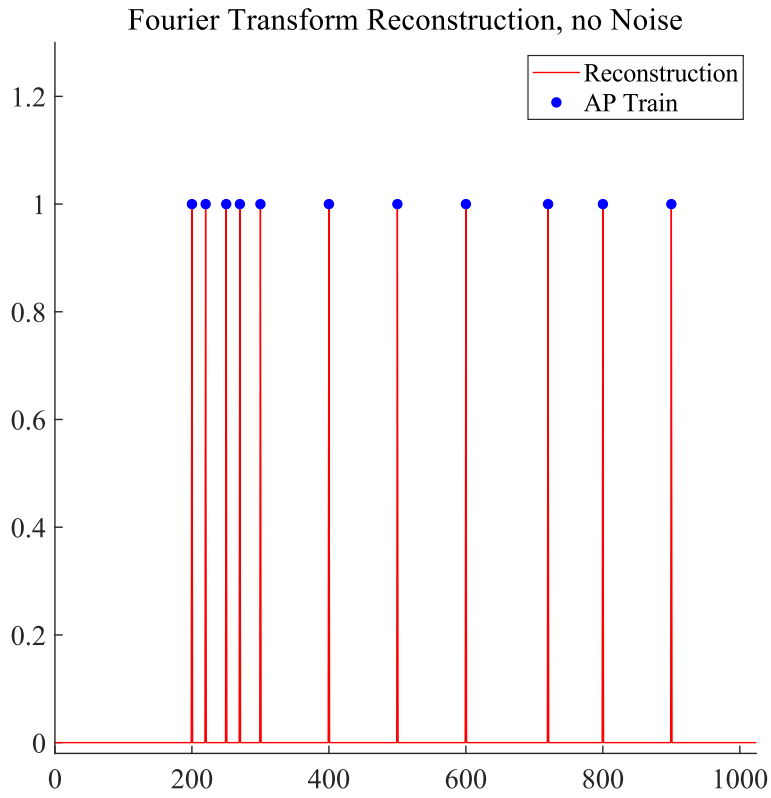
3.2.3. 实验内容

```
% 信号的产生
% 动作电位脉冲
x = zeros(1024, 1);
ap_train = [200, 220, 250, 270, 300, 400, 500, 720, 600, 800, 900];
x(ap_train, 1) = 1;
% 指数下降模板
t = 15;
h = exp(-(0:1:ceil(10 * t)) / t)';
% 无噪声的钙信号
y = conv_ft(x, h);
% 信噪比 dB
SNR = 15;
% 有噪声的钙信号
y_noise = awgn(y, SNR, 'measured');
%%
x_deconv = deconv_dft(y_noise, h);

figure()
plot(x_deconv, '-r');
hold on
plot(ap_train, 1, '.b', 'MarkerSize', 20);
legend('Reconstruction', 'AP Train')
hold off
title('Fourier Transform Reconstruction, no Noise')
xlim([0, 1024])
```

```
ylim([-0.02, 1.3])
box off
```

3.3. 实验结果





对于无噪声的信号，通过傅里叶变换和循环卷积可以重建出误差很小的信号。

但是，对于带有噪声的钙信号，由于叠加的噪声并不是周期性信号，傅里叶变换无法对其进行有效的降噪。由于叠加的是白噪声，因此直接对频域信号进行计算时，甚至有可能增大高频部分的噪声。

4. 一维离散傅里叶变换及其逆变换（DFT & IDFT）

4.1. 实验原理

长度为N的数字信号序列的离散傅里叶变换可以定义为：

$$\begin{cases} X(k)=\mathrm{DFT}[x(n)]=\sum_{n=0}^{N-1}x(n)e^{-j\frac{2\pi}{N}}=\sum_{n=0}^{N-1}x(n)W_N^{nk} \\ x(n)=\mathrm{IDFT}[X(k)]=\frac{1}{N}\sum_{n=0}^{N-1}X(k)e^{j\frac{2\pi}{N}nk}=\frac{1}{N}\sum_{n=0}^{N-1}X(k)W_N^{-nk} \end{cases} \quad k=0,1,2,\cdots,N-1 \tag{2-4}$$

$\vec{X}$和$\vec{x}$可以用列向量的形式表示：

$$\begin{cases} \vec{X}=[X(0),X(1),\cdots,X(N-1)]^T \\ \vec{x}=[x(0),x(1),\cdots,x(N-1)]^T \end{cases} \tag{2-5}$$

由式 2-4 定义的离散傅里叶变换及其逆变换也可以用矩阵表示为：

$$\vec{X} = W \cdot \vec{x} = \begin{bmatrix} e^{-j2\pi\frac{0\times0}{N}} & e^{-j2\pi\frac{0\times1}{N}} & e^{-j2\pi\frac{0\times2}{N}} & \cdots & e^{-j2\pi\frac{0\times(N-1)}{N}} \\ e^{-j2\pi\frac{1\times0}{N}} & e^{-j2\pi\frac{1\times1}{N}} & e^{-j2\pi\frac{1\times2}{N}} & \cdots & e^{-j2\pi\frac{1\times(N-1)}{N}} \\ e^{-j2\pi\frac{2\times0}{N}} & e^{-j2\pi\frac{2\times1}{N}} & e^{-j2\pi\frac{2\times2}{N}} & \cdots & e^{-j2\pi\frac{2\times(N-1)}{N}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ e^{-j2\pi\frac{(N-1)\times0}{N}} & e^{-j2\pi\frac{(N-1)\times1}{N}} & e^{-j2\pi\frac{(N-1)\times2}{N}} & \cdots & e^{-j2\pi\frac{(N-1)\times(N-1)}{N}} \end{bmatrix} \cdot \vec{x} \tag{2-6}$$

$$\vec{x} = W^{-1} \cdot \vec{X} = \frac{1}{N} \begin{bmatrix} e^{j2\pi\frac{0\times0}{N}} & e^{j2\pi\frac{0\times1}{N}} & e^{j2\pi\frac{0\times2}{N}} & \cdots & e^{j2\pi\frac{0\times(N-1)}{N}} \\ e^{j2\pi\frac{1\times0}{N}} & e^{j2\pi\frac{1\times1}{N}} & e^{j2\pi\frac{1\times2}{N}} & \cdots & e^{j2\pi\frac{1\times(N-1)}{N}} \\ e^{j2\pi\frac{2\times0}{N}} & e^{j2\pi\frac{2\times1}{N}} & e^{j2\pi\frac{2\times2}{N}} & \cdots & e^{j2\pi\frac{2\times(N-1)}{N}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ e^{j2\pi\frac{(N-1)\times0}{N}} & e^{j2\pi\frac{(N-1)\times1}{N}} & e^{j2\pi\frac{(N-1)\times2}{N}} & \cdots & e^{j2\pi\frac{(N-1)\times(N-1)}{N}} \end{bmatrix} \cdot \vec{X} \tag{2-7}$$

4.2. MATLAB 实现

4.2.1. DFT

```
function F = dft(x)
len_x = size(x, 1);
w = (0:1:len_x - 1);
W = exp(-2 .* 1i .* pi .* (w' * w) / len_x);
F = W * x;
end
```

4.2.2. IDFT

```
function x = idft(F)
len_x = size(F, 1);
w = (0:1:len_x - 1);
G = exp(2 .* 1i .* pi .* (w' * w) / len_x) ./ len_x;
x = G * F;
end
```

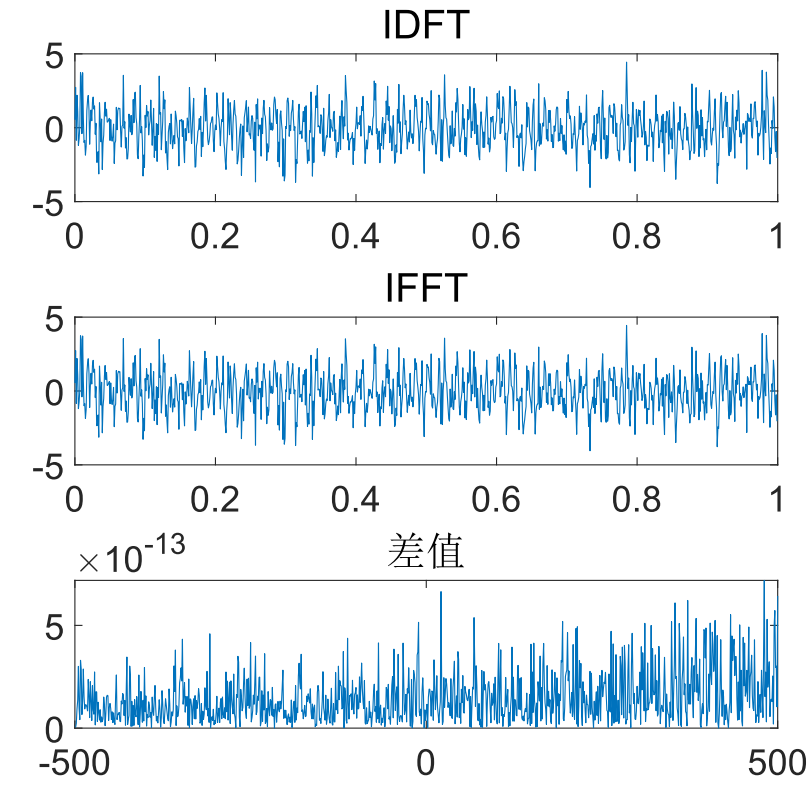
4.2.3. 实验内容

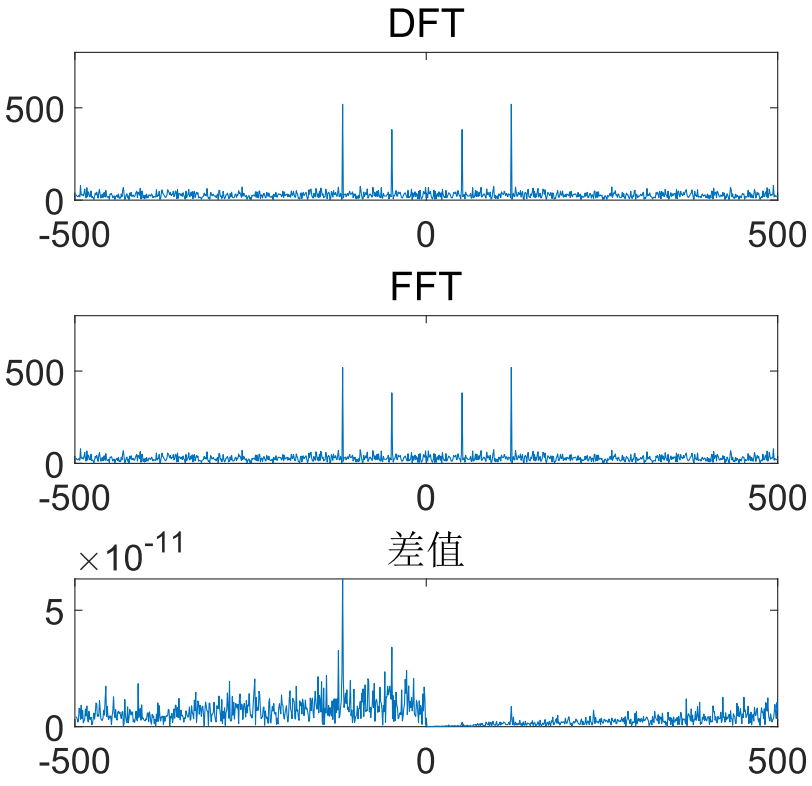
```
Fs = 1000;
T = 1 / Fs;
L = 1000;
t = (0:L - 1)' * T;

S = 0.7 * sin(2 * pi * 50 * t) + sin(2 * pi * 120 * t);
```

```
X = S + 1 * randn(size(t));

Fx_dft = dft(X);
Fx_fft = fft(X);

Fx_diff = Fx_dft - Fx_fft;

X_ifft = real(ifft(Fx_fft));
X_idft = real(idft(Fx_dft));

X_diff = (X_idft - X);

f = Fs * (1:1:L) / L - 500;

figure()
subplot(3, 1, 1)
plot(f, abs(fftshift(Fx_dft)))
ylim([0, 800])
title('DFT')
subplot(3, 1, 2)
plot(f, abs(fftshift(Fx_fft)))
ylim([0, 800])
title('FFT')
subplot(3, 1, 3)
plot(f, abs(fftshift(Fx_diff)))
title('差值')

figure()
subplot(3, 1, 1)
plot(t, X_ifft)
title('IDFT')
subplot(3, 1, 2)
plot(t, X_idft)
title('IFFT')
subplot(3, 1, 3)
plot(f, abs(X_ifft - X_idft))
title('差值')
```

4.3. 实验结果

测试数据为采样率 1000Hz，时间 1s（即 1000 个离散点）的数据。其中包括频率为 50Hz 和 120Hz 的正弦波和高斯噪声。对数据分别使用库函数（fft 和 ifft）以及自定义的离散傅里叶变换函数进行傅里叶变换和逆变换。


IDFT
5
0
-5
0
0.2
0.4
0.6
0.8
1
IFFT
5
0
-5
0
0.2
0.4
0.6
0.8
1
×10-13
差值
5
0
-500
0
500


DFT
500
0
-500
0
500
FFT
500
0
-500
0
500
×10-11
差值
5
0
-500
0
500

在进行离散傅里叶变换后，可以在±50Hz 和±120Hz 处看到明显的峰。库函数和自定义函数的结果的差值小于10^{-11}，属于浮点数运算误差。

5. 二维离散傅里叶变换及其逆变换（DFT2 & IDFT2）

5.1. 实验原理

类似一维离散傅里叶变换，对于$M \times N$的二维离散信号，傅里叶变换和逆变换的定义为：

$$\begin{cases} F[u,v] = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) e^{-j\cdot 2\pi(\frac{ux}{M}+\frac{vy}{N})} \\ f(x,y) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} F(u,v) e^{j\cdot 2\pi(\frac{ux}{M}+\frac{vy}{N})} \end{cases} \tag{2-8}$$

和一维傅里叶变换类似，傅里叶变换及其逆变换也可以用矩阵乘法表示：

$$G_M = \begin{bmatrix} e^{-j2\pi\frac{0\times 0}{M}} & e^{-j2\pi\frac{0\times 1}{M}} & e^{-j2\pi\frac{0\times 2}{M}} & \cdots & e^{-j2\pi\frac{0\times (M-1)}{M}} \\ e^{-j2\pi\frac{1\times 0}{M}} & e^{-j2\pi\frac{1\times 1}{M}} & e^{-j2\pi\frac{1\times 2}{M}} & \cdots & e^{-j2\pi\frac{1\times (M-1)}{M}} \\ e^{-j2\pi\frac{2\times 0}{M}} & e^{-j2\pi\frac{2\times 1}{M}} & e^{-j2\pi\frac{2\times 2}{M}} & \cdots & e^{-j2\pi\frac{2\times (M-1)}{M}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ e^{-j2\pi\frac{(M-1)\times 0}{M}} & e^{-j2\pi\frac{(M-1)\times 1}{M}} & e^{-j2\pi\frac{(M-1)\times 2}{M}} & \cdots & e^{-j2\pi\frac{(M-1)\times (M-1)}{M}} \end{bmatrix} \tag{2-9}$$

$$G_N = \begin{bmatrix} e^{-j2\pi\frac{0\times 0}{N}} & e^{-j2\pi\frac{0\times 1}{N}} & e^{-j2\pi\frac{0\times 2}{N}} & \cdots & e^{-j2\pi\frac{0\times (N-1)}{N}} \\ e^{-j2\pi\frac{1\times 0}{N}} & e^{-j2\pi\frac{1\times 1}{N}} & e^{-j2\pi\frac{1\times 2}{N}} & \cdots & e^{-j2\pi\frac{1\times (N-1)}{N}} \\ e^{-j2\pi\frac{2\times 0}{N}} & e^{-j2\pi\frac{2\times 1}{N}} & e^{-j2\pi\frac{2\times 2}{N}} & \cdots & e^{-j2\pi\frac{2\times (N-1)}{N}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ e^{-j2\pi\frac{(N-1)\times 0}{N}} & e^{-j2\pi\frac{(N-1)\times 1}{N}} & e^{-j2\pi\frac{(N-1)\times 2}{N}} & \cdots & e^{-j2\pi\frac{(N-1)\times (N-1)}{N}} \end{bmatrix} \tag{2-10}$$

$$F = G_M \cdot f \cdot G_N \tag{2-11}$$

$$G_M^{-1} = \frac{1}{M}\begin{bmatrix} e^{j2\pi\frac{0\times 0}{M}} & e^{j2\pi\frac{0\times 1}{M}} & e^{j2\pi\frac{0\times 2}{M}} & \cdots & e^{j2\pi\frac{0\times (M-1)}{M}} \\ e^{j2\pi\frac{1\times 0}{M}} & e^{j2\pi\frac{1\times 1}{M}} & e^{j2\pi\frac{1\times 2}{M}} & \cdots & e^{j2\pi\frac{1\times (M-1)}{M}} \\ e^{j2\pi\frac{2\times 0}{M}} & e^{j2\pi\frac{2\times 1}{M}} & e^{j2\pi\frac{2\times 2}{M}} & \cdots & e^{j2\pi\frac{2\times (M-1)}{M}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ e^{j2\pi\frac{(M-1)\times 0}{M}} & e^{j2\pi\frac{(M-1)\times 1}{M}} & e^{j2\pi\frac{(M-1)\times 2}{M}} & \cdots & e^{j2\pi\frac{(M-1)\times (M-1)}{M}} \end{bmatrix} \tag{2-12}$$

$$G_N^{-1} = \frac{1}{N}\begin{bmatrix} e^{j2\pi\frac{0\times0}{N}} & e^{j2\pi\frac{0\times1}{N}} & e^{j2\pi\frac{0\times2}{N}} & \cdots & e^{j2\pi\frac{0\times(N-1)}{N}} \\ e^{j2\pi\frac{1\times0}{N}} & e^{j2\pi\frac{1\times1}{N}} & e^{j2\pi\frac{1\times2}{N}} & \cdots & e^{j2\pi\frac{1\times(N-1)}{N}} \\ e^{j2\pi\frac{2\times0}{N}} & e^{j2\pi\frac{2\times1}{N}} & e^{j2\pi\frac{2\times2}{N}} & \cdots & e^{j2\pi\frac{2\times(N-1)}{N}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ e^{j2\pi\frac{(N-1)\times0}{N}} & e^{j2\pi\frac{(N-1)\times1}{N}} & e^{j2\pi\frac{(N-1)\times2}{N}} & \cdots & e^{j2\pi\frac{(N-1)\times(N-1)}{N}} \end{bmatrix} \tag{2-13}$$

$$f = G_M^{-1} \cdot F \cdot G_N^{-1} \tag{2-14}$$

5.2. MATLAB 实现

5.2.1. DFT2

```
function F = dft2(x)
[row, col] = size(x);
w_row = (0:1:row - 1);
w_col = (0:1:col - 1);
Gm = exp(-2 .* 1i .* pi .* (w_row' * w_row) / row);
Gn = exp(-2 .* 1i .* pi .* (w_col' * w_col) / col);
F = Gm * x * Gn;
end
```

5.2.2. IDFT2

```
function x = idft2(F)
[row, col] = size(F);
w_row = (0:1:row - 1);
w_col = (0:1:col - 1);
Gm = exp(2 .* 1i .* pi .* (w_row' * w_row) / row) ./ row;
Gn = exp(2 .* 1i .* pi .* (w_col' * w_col) / col) ./ col;
x = Gm * F * Gn;
end
```

5.2.3. 实验内容

```
img = imread('cameraman.tif');
img = im2double(img);
img = imresize(img, [512 512]);

img_dft = dft2(img);
img_idft = real(idft2(img_dft));
img_freq_log = log10(abs(fftshift(img_dft)));

img_dft_2 = fft2(img);
img_idft_2 = real(ifft2(img_dft_2));
```

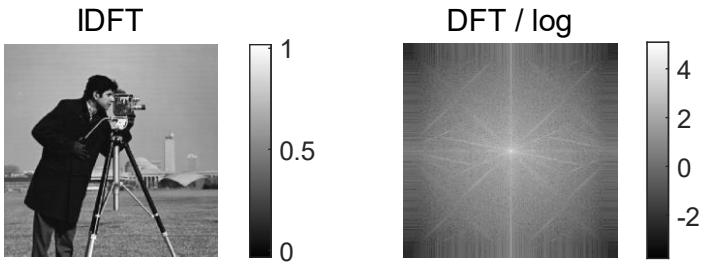
```
img_freq_log_2 = log10(abs(fftshift(img_dft_2)));
img_dft_diff = (img_idft - img_idft_2);

figure(1)
subplot(1, 2, 1)
imshow(img_idft, [])
colorbar
title('IDFT')
subplot(1, 2, 2)
imshow(img_freq_log, [])
colorbar
title('DFT / log')

figure(2)
subplot(1, 2, 1)
imshow(img_idft_2, []);
colorbar
title('IFFT')
subplot(1, 2, 2)
imshow(img_freq_log_2, [])
colorbar
title('FFT / log')

figure(3)
subplot(1, 2, 1)
imshow(img_dft_diff, [])
colorbar
title('重建图像差值')
subplot(1, 2, 2)
imshow(abs(fftshift(img_dft - img_dft_2)), [])
colorbar
title('频谱差值')
```

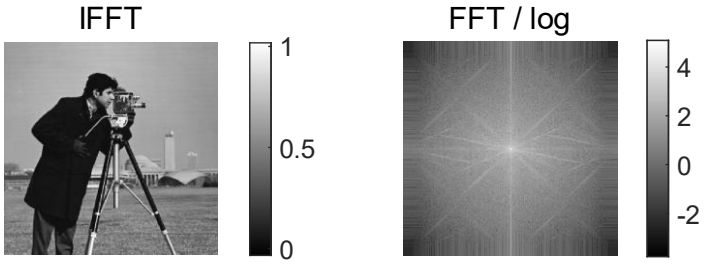
5.3. 实验结果

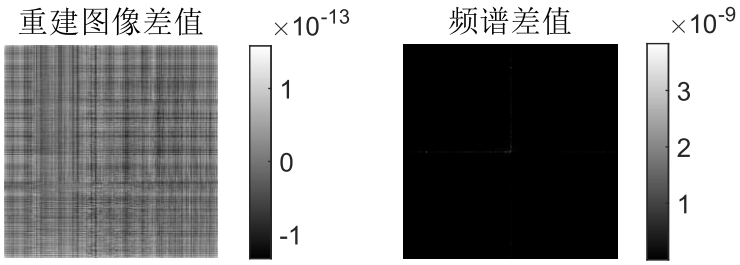
使用 MATLAB 中的图片 cameraman.tif，并将其尺寸放大到512×512，动态范围线性缩放到 0-1 进行实验。

使用 MATLAB 中的函数 immse 衡量图像之间的误差（均方误差），对于尺寸为$M \times N$的两张图片，其均方误差为：

$$\mathrm{IMMSE}(X,Y) = \frac{\| X - Y \|_2^2}{MN} \tag{2-15}$$







通过库函数（fft2 和 ifft2）计算得到的结果与通过自定义函数得到的结果之间的均方误差为 1.01×10^{-27}，可以认为是浮点数运算过程中引入的误差。

6. 实验总结

本次实验的主要内容是傅里叶级数和傅里叶变换。在之前的《信号与系统》课程中，我也接触过和离散傅里叶变换相关的内容，但是对于离散傅里叶变换的实现之前并没有过多的了解。在理论课上，我已经学习了一些相关的内容。对于傅里叶级数和傅里叶变换；离散信号和数字信号之间的联系已经有了一定的了解。通过课程实验，我也对相关的内容有了更深刻的理解。

离散傅里叶变换的计算公式我参考了数字信号处理教材中所提及的矩阵相乘的方法来进行计算。最终实现了和 MATLAB 中的快速傅里叶变换相同的结果。从中，我对离散傅里叶变换的计算方法和结果也有了更深刻的理解。

最后的去卷积实验，由于我对于盲去卷积等相关内容并不是很了解，实验的结果明显没有实验一中的稀疏重建后。由于噪声信号在频域没有明显特征，因此在频域处理的效果不好，甚至有导致高频噪声增加的倾向。