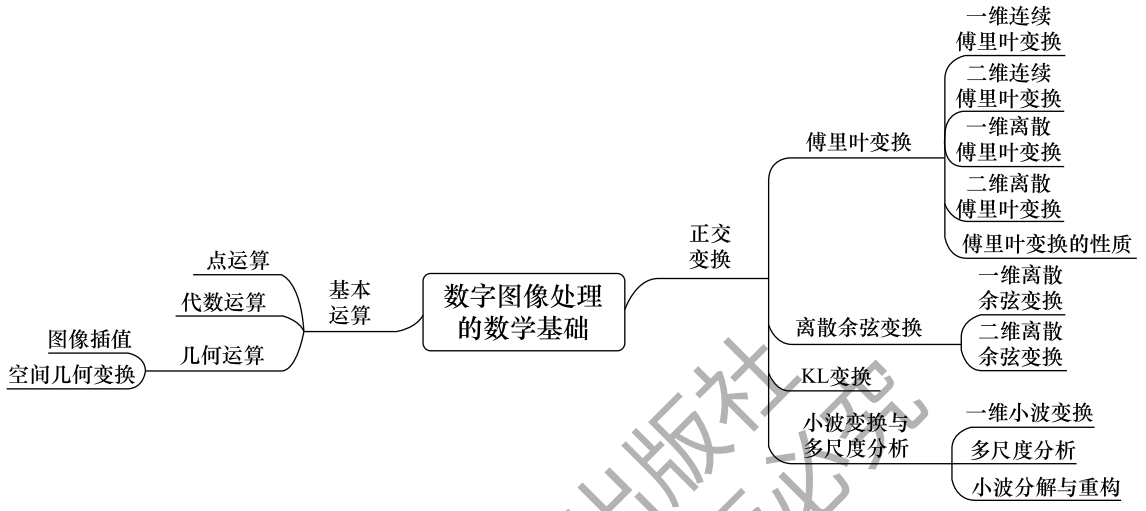
第 3 章　数字图像处理的数学基础

※本章思维导图



※学习目标

1. 能编程实现数字图像处理中的基本运算。
2. 熟练使用傅里叶变换等图像变换处理图像矩阵。
3. 能阐述图像变换的原理和过程，并调用 MATLAB/Python 函数实现。

为了快速有效地对图像进行处理与分析，常常需要对像素矩阵进行运算、变换。本章将介绍图像的基本运算、像素矩阵的变换及其相应的 MATLAB/Python 函数。这些数学工具在后面图像降噪、增强、去模糊、修复与压缩中都有重要应用。

电子工业出版社 版权所有 盗版必究

3.1　数字图像的基本运算

图像间的基本运算主要包括图像的点运算、代数运算及几何运算。MATLAB/Python 提供了图像运算的多种函数，读者通过本节的学习可以编程实现这些基本运算。

3.1.1　点运算

图像的点运算用于逐点改变一幅图像的灰度分布。假设图像像素矩阵的(i,j)位置的灰度值为 $f(x_i,y_i)$，则图像的点运算操作涉及图像的单个像素的灰度值，其处理过程为

$$g(x_i,y_i)=T(f(x_i,y_i)) \tag{3.1}$$

其中，$T(\cdot)$表示某类变换，例如线性变换 $T(x)=ax+b$，a,b 为给定参数；$g(x_i,y_i)$表示运算后的结果。图像的点运算可以改变像素的灰度值，应用于光学仪器校准；提高目标与背景的对比度，达到突出感兴趣区域的效果；利用灰度值划分图像区域，给图像加上轮廓线。MATLAB 中图像线性变换函数为

$$Z=\mathrm{imlincomb}(K_1,A_1,K_2,A_2,\cdots,K_n,A_n,K)$$

其中，$K_1,K_2,\cdots,K_n$ 表示权重系数；$A_1,A_2,\cdots,A_n$ 表示输入图像；K 表示一个常数。

3.1.2 代数运算

图像间的代数运算主要包括像素间的四则运算。假设两幅灰度图像分别为 $f(x,y)$和 $g(x,y)$，是大小为 $M\times N$ 像素的灰度图像。图像间的代数运算指的是对应像素间执行基本四则运算

$$s(x,y)=f(x,y)+g(x,y) \tag{3.2}$$

$$d(x,y)=f(x,y)-g(x,y) \tag{3.3}$$

$$p(x,y)=f(x,y)\times g(x,y) \tag{3.4}$$

$$v(x,y)=f(x,y)\div g(x,y) \tag{3.5}$$

其中，$x=0,1,\cdots,M$；$y=0,1,\cdots,N$。显然，经过运算所得到的图像 $s(x,y)$、$d(x,y)$、$p(x,y)$、$v(x,y)$也是大小为 $M\times N$ 像素的灰度图像。

图像间的代数运算有重要的应用，例如带有不同噪声的同一图像相加可以降噪（见例 3.1）。其他的应用包括图像相减可以增强图像的差别，图像的相乘（相除）可以用来校正阴影。在 MATLAB/Python 中，图像间的代数运算可以通过调用函数来实现。设 X 和 Y 表示需要运算的两幅图像，返回值 Z 表示得到的操作结果，相应的函数如表 3.1 所示。

表 3.1　MATLAB/Python 中图像间的代数运算函数

	加法	减法	乘法	除法
MATLAB	Z=imadd(X,Y)	Z=imsubtract(X,Y)	Z=immultiply(X,Y)	Z=imdivide(X,Y)
Python	Z=cv2.add(X,Y)	Z=cv2.subtract(X,Y)	Z=cv2.multiply(X,Y)	Z=cv2.divide(X,Y)

【例 3.1】 设 $g(x,y)$是无噪声图像，$\eta(x,y)$是噪声，观测 $g_i(x,y)=g(x,y)+\eta_i(x,y)(i=1,2,\cdots,k)$，若噪声 $\eta(x,y)$在点(x,y)处是不相关的且均值为零。$\bar{g}(x,y)$代表由 k 幅不同噪声图像平均而成，即

$$\bar{g}(x,y)=\frac{1}{k}\sum_{i=1}^{k}g_i(x,y) \tag{3.6}$$

可得 $E\{\bar{g}(x,y)\}=g(x,y)$，$\sigma_{\bar{g}(x,y)}=\frac{1}{k}\sigma^2_{\eta(x,y)}$，其中 $\sigma_{\bar{g}(x,y)}$ 和 $\sigma^2_{\eta(x,y)}$ 分别表示 $\bar{g}$ 和 η 在点(x,y)处的方差。当 k 增大时，$\sigma_{\bar{g}(x,y)}$ 减小，从而减少噪声。图像平均的一项重要应用是在天文学领域，该领域中由于单幅图像常常伴有传感器噪声，从而导致无法分析。其解决办法是使用传感器阵列获取图像（CCD 成像），长时间地观察同一场景以达到降噪的目的[1]。

如图 3.1 所示为图像间代数相减实例。

MATLAB 主要代码如下：

```
I=imread('cameraman.tif');
J=uint8(filter2(fspecial('gaussian'),I));
K=imlincomb(1,I,-1,J,128);
```

Python 主要代码如下：

```
# 读入原始图像并转为灰度图像
I=cv2.imread('cameraman.jpg')
I=cv2.cvtColor(I,cv2.COLOR_BGR2GRAY)
```

电子工业出版社 版权所有 盗版必究

```
# 对图像进行高斯平滑操作
J=cv2.filter2D(I,-1,cv2.getGaussianKernel(5,1))
# 对图像进行代数相减
Z=cv2.subtract(I,J)
# 或者通过 addWeighted()对图像进行线性组合
K=cv2.addWeighted(I,1,J,- 1,128,dtype=cv2.CV_8U)
```

(a) 原始图像

(b) 图(a)高斯滤波结果

(c) 图(a)和图(b)的代数相减

图 3.1　图像间代数相减实例

3.1.3　几何运算

图像的几何运算包括图像插值与空间几何变换。

1. 图像插值

图像的插值是根据原始图像像素(原始像素)的灰度值估计周围像素的灰度值,它是图像缩放的基础。MATLAB 提供 3 种二维图像插值方法:最近邻插值、双线性插值和双三次插值。在 3 种方法中,最近邻插值用最近邻位置的灰度值作为目标像素的灰度值估计;双线性插值采用最近的 2×2 邻域内像素的灰度值加权平均作为目标像素的灰度值估计;双三次插值采用最近的 4×4 邻域内像素的灰度值加权平均作为目标像素的灰度值估计。MATLAB 中二维图像插值函数为

$$ZI=\text{interp2}(X,Y,Z,XI,YI,\text{Method})$$

其中,X、Y 为原始像素位置,Z 为原始像素的灰度值。相应地,XI、YI 为目标像素位置的返回值,ZI 为目标像素的估计值。Method 是所用的插值方法,例如,'linear'表示双线性插值,'nearest'表示最近邻插值,'cubic'表示双三次插值。相应的插值算法原理可以参考数值分析方面的教材,这里不再赘述。

Python 可以通过第三方库 scipy 中的 interpolate 实现二维图像的插值。Python 中的插值函数为

$$\text{newfunc}=\text{scipy.interpolate.interp2d}(X,Y,Z,\text{kind}='\text{linear}')$$

其中,X、Y 为原始像素位置;Z=f(X,Y),为在原始像素灰度值处插入的数值;kind 为可选择的插值方法,例如,'linear'表示双线性插值,'cubic'表示双三次插值,'quintic'表示五次插值,默认为'linear'。

2. 空间几何变换

对图像进行平移、旋转时,需要对图像进行空间几何变换,其数学表达式为

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \boldsymbol{T} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3.7}$$

电子工业出版社 版权所有 盗版必究

其中，(x,y,z)表示原空间坐标，(x',y',z')为经过变换 $\boldsymbol{T}$ 后在变换空间中的坐标。因为该变换不是齐次变换，所以定义二维规范齐次坐标为$(x,y,1)$，则相应的仿射变换为

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \boldsymbol{T} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & p \\ c & d & q \\ l & m & s \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.8}$$

其中，$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$实现图形的比例变换、对称变换、旋转变换和错切变换；$\begin{bmatrix} p \\ q \end{bmatrix}$实现平移变换，$p$、$q$ 分别表示沿 x、y 方向的平移量；s 表示图形的缩放比例；$[l \quad m]$的作用是实现透视变换，符号物理意义的解释可以参考文献[3]。

平移变换：假设某点 P_0 对应坐标(x_0,y_0)，经过$(\nabla x,\nabla y)$的平移变换后，变换到点 P 对应坐标(x,y)，在直角坐标系下存在变换关系

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \nabla x \\ 0 & 1 & \nabla y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} \tag{3.9}$$

旋转变换：假设某点 P_0 对应坐标(x_0,y_0)，经过角度为 θ 的顺时针旋转后，变换到点 P 对应坐标(x,y)，在极坐标系下存在变换关系

$$x_0 = r\cos\alpha, y_0 = r\sin\alpha \tag{3.10}$$

$$x = r\cos(\alpha+\theta) = r\cos\alpha\cos\theta - r\sin\alpha\sin\theta = x_0\cos\theta - y_0\sin\theta \tag{3.11}$$

$$y = r\sin(\alpha+\theta) = r\sin\alpha\cos\theta + r\cos\alpha\sin\theta = x_0\sin\theta + y_0\cos\theta \tag{3.12}$$

而有

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} \tag{3.13}$$

MATLAB 中图像空间几何变换函数为 imtransform()。图 3.2 为图 3.1(a)经过仿射变换 $\begin{bmatrix} 1 & 0 & 0 \\ 0.5 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$后的图像。

MATLAB 主要代码如下：

```
I=imread('cameraman.tif');
tform=maketform('affine',[1 0 0;.5 1 0;0 0 1]);
J=imtransform(I,tform);
```

Python 主要代码如下：

```
# 仿射变换
M=np.array([1,0,0],[0.5,1,0]],np.float32)
Affine_img=cv2.warpAffine(img,M,(w,h))
```

电子工业出版社 版权所有 盗版必究

图 3.2　经过仿射变换后的图像

3.2　数字图像的正交变换

回顾线性代数中线性变换的概念，对于向量 $\boldsymbol{x},\boldsymbol{y}\in R^d$，$a,b\in R^1$，我们称 $\mathcal{L}$ 为线性变换，当且仅当 $\mathcal{L}(\cdot)$ 满足如下性质：

$$\mathcal{L}(a\boldsymbol{x}+b\boldsymbol{y})=a\mathcal{L}\boldsymbol{x}+b\mathcal{L}\boldsymbol{y} \tag{3.14}$$

$$\mathcal{L}(a\boldsymbol{x})=a\mathcal{L}\boldsymbol{x} \tag{3.15}$$

在图像处理中常见的线性变换为积分变换。使用积分变换时，图像作为线性空间来处理，通过某些变换将常用图像函数从空域(Spatial Domain)变换到频域(Frequency Domain)，图像被表达为某种线性积分变换的一组基函数$\{\varphi_i\}_{i=1,2,\cdots}$的线性组合。例如，傅里叶变换(Fourier Transform)使用正弦函数和余弦函数作为基函数。如果所使用的基函数满足正交条件，则称该变换为正交变换。本节将详细介绍 4 种变换，在此之前首先介绍狄拉克函数 $\delta(x,y)$ 的定义与性质。

从连续信号过渡到离散信号，通常要使用狄拉克函数 $\delta(x,y)$。$\delta(x,y)$ 满足如下条件：

$$\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\delta(x,y)\mathrm{d}x\mathrm{d}y=1 \tag{3.16}$$

另外，$\delta(x,y)$ 函数有如下重要性质：对于所有 $x,y\neq 0$，$\delta(x,y)=0$，则恒等式

$$\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}f(x,y)\delta(x-\lambda,y-u)\mathrm{d}x\mathrm{d}y=f(\lambda,u) \tag{3.17}$$

成立。

3.2.1　傅里叶变换

1. 一维连续傅里叶变换

设 $f(x)$ 为变量 x 的连续可积函数，则定义 $f(x)$ 的傅里叶变换为

$$F(u)=\int_{-\infty}^{+\infty}f(x)\mathrm{e}^{-\mathrm{j}2\pi ux}\mathrm{d}x \tag{3.18}$$

其中，j 为虚数单位，u 为频域变量，x 为空域变量。

将 $F(u)$ 恢复到 $f(x)$ 的变换称为傅里叶逆变换，定义为

$$f(x)=\int_{-\infty}^{+\infty}F(u)\mathrm{e}^{\mathrm{j}2\pi ux}\mathrm{d}u \tag{3.19}$$

需要注意的是，实函数的傅里叶变换，其结果多为复函数。不妨假设 $F(u)$ 的实部和虚部分别为 $R(u)$ 和 $I(u)$，则有

电子工业出版社 版权所有 盗版必究

$$F(u)=R(u)+\mathrm{j}I(u) \tag{3.20}$$

约定傅里叶变换的傅里叶谱、能量谱及相位角分别为$|F(u)|$、$E(u)$、$\varphi(u)$，即

$$|F(u)|=\sqrt{R^2(u)+I^2(u)} \tag{3.21}$$

$$E(u)=|F(u)|^2=R^2(u)+I^2(u) \tag{3.22}$$

$$\varphi(u)=\arctan\frac{I(u)}{R(u)} \tag{3.23}$$

式(3.18)中，指数项$\mathrm{e}^{-\mathrm{j}2\pi ux}$应用欧拉公式可以展开为

$$\mathrm{e}^{-\mathrm{j}2\pi ux}=\cos 2\pi ux-\mathrm{j}\sin 2\pi ux \tag{3.24}$$

能量谱是图像的重要特征，反映图像的灰度分布。例如，精细结构和细微结构的图像高频分量丰富，而低频分量反映图像的概貌。

从欧拉公式可以看出，指数函数可以表达为正弦函数和余弦函数的代数和，利用正弦函数和余弦函数的奇偶性可以简化傅里叶变换式(3.18)的计算。可以证明傅里叶变换是正交变换，也是完备的。

【例 3.2】 求函数$f(x)=\begin{cases}\dfrac{1}{2}, & -1\leqslant x\leqslant 1\\ 0, & \text{其他}\end{cases}$的傅里叶变换及其傅里叶谱。

解： 由傅里叶变换式(3.18)得

$$F(u)=\int_{-1}^{1}\frac{1}{2}\mathrm{e}^{-\mathrm{j}2\pi ux}\,\mathrm{d}x=\frac{\sin(2\pi u)}{2\pi u} \tag{3.25}$$

$$|F(u)|=\left|\frac{\sin(2\pi u)}{2\pi u}\right| \tag{3.26}$$

MATLAB 主要代码如下：

```
x=linspace(-5,5,100);
y1(1:40)=0;
y1(41:60)=1/2;
y1(61:100)=0;
y2=sin(2*pi*x)./(2*pi*x);
y3=abs(sin(2*pi*x)./(2*pi*x));
```

Python 主要代码如下：

```
import matplotlib.pyplot as plt
import numpy as np
import math

x=np.linspace(-5,5,100)# -5到5,共输出100个数,步长为0.1
y1=np.zeros(100)
y1[40:60]=0.5
y2=np.sin(2*math.pi*x)/(2*math.pi*x)
y3=abs(np.sin(2*math.pi-x)/(2*math.pi-x))
```

将$f(x)$、$F(u)$、$|F(u)|$绘图，如图 3.3 所示。对于复杂形式的$f(x)$，傅里叶变换的计算并不简单。

电子工业出版社 版权所有 盗版必究

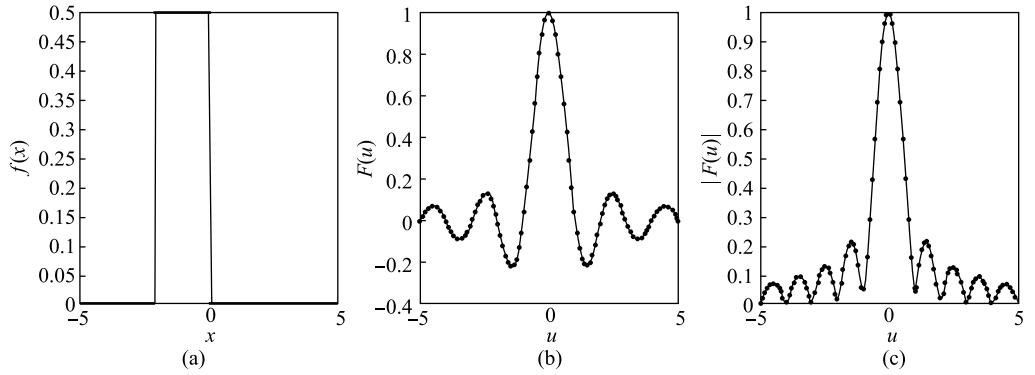


图 3.3　函数 $f(x)$、$f(x)$的傅里叶变换及其傅里叶谱示意图

傅里叶变换的一个重要应用是快速计算卷积。卷积运算在深度学习与图像处理中有着广泛应用。本节约定连续变量 t 的两个连续函数 $g(t)$和 $h(t)$的卷积由算子 $*$ 表示，定义为

$$g(t)*h(t)=\int_{-\infty}^{+\infty}g(\tau)h(t-\tau)\mathrm{d}\tau \tag{3.27}$$

其中，τ 表示时域积分变量，而 t 表示两个连续函数间的时域平移。函数 $g(t)$和 $h(t)$的定义域从$-\infty$到$+\infty$。

令 $g(t)*h(t)$、$g(t)$、$h(t)$的傅里叶变换分别为 $F(g(t)*h(t))$、$G(u)$、$H(u)$，t 为空域变量，u 为频域变量，则

$$\begin{aligned}F(g(t)*h(t))&=\int_{-\infty}^{+\infty}\left[\int_{-\infty}^{+\infty}g(\tau)h(t-\tau)\mathrm{d}\tau\right]\mathrm{e}^{-2\mathrm{j}\pi ut}\mathrm{d}t\\&=\int_{-\infty}^{+\infty}g(\tau)\left[\int_{-\infty}^{+\infty}h(t-\tau)\mathrm{e}^{-2\mathrm{j}\pi ut}\mathrm{d}t\right]\mathrm{d}\tau\\&=H(u)\int_{-\infty}^{+\infty}g(\tau)\mathrm{e}^{-\mathrm{j}2\pi u\tau}\mathrm{d}\tau=H(u)G(u)\end{aligned} \tag{3.28}$$

其中，j 为虚数单位。

2. 二维连续傅里叶变换

一维连续傅里叶变换可以推广到两个变量连续可积的函数 $f(x,y)$。若 $F(u,v)$是可积的，则存在如下傅里叶变换对：

$$F(u,v)=\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}f(x,y)\mathrm{e}^{-\mathrm{j}2\pi(ux+vy)}\mathrm{d}x\mathrm{d}y \tag{3.29}$$

$$f(x,y)=\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}F(u,v)\mathrm{e}^{\mathrm{j}2\pi(ux+vy)}\mathrm{d}u\mathrm{d}v \tag{3.30}$$

相应的傅里叶谱（幅度谱）、能量谱及相位谱分别定义为

$$|F(u,v)|=\sqrt{R^2(u,v)+I^2(u,v)} \tag{3.31}$$

$$E(u,v)=R^2(u,v)+I^2(u,v) \tag{3.32}$$

$$\varphi(u,v)=\arctan\frac{I(u,v)}{R(u,v)} \tag{3.33}$$

注意：本章将在章末练习题中探讨幅度谱与相位谱所包含的图像信息。

【例 3.3】 求函数 $f(x,y)=\begin{cases}\dfrac{1}{2}, & -1\leqslant x,y\leqslant 1\\ 0, & \text{其他}\end{cases}$的傅里叶变换。

电子工业出版社
版权所有 盗版必究

解： 由傅里叶变换式(3.29)得

$$F(u,v)=\int_{-1}^{1}\int_{-1}^{1}\frac{1}{2}\mathrm{e}^{-\mathrm{j}2\pi(ux+vy)}\mathrm{d}x\mathrm{d}y=\frac{\sin(2\pi u)}{2\pi u}\ \frac{\sin(2\pi v)}{2\pi v} \tag{3.34}$$

3. 一维离散傅里叶变换

由于现实世界对函数的观测通常是离散的，假设对连续函数 $f(x)$ 等间隔采样得到一个离散序列，该序列的采样点为 N 个，均匀间隔为 ΔT，则这个离散序列具有如下形式：$\{f(0),f(\Delta T),\cdots,f(N-1)\Delta T\}$。

一维离散傅里叶变换对定义为

$$F(u)=\sum_{x=0}^{N-1}f(x)\mathrm{e}^{-\mathrm{j}\frac{2\pi ux}{N}}\quad(u=0,1,\cdots,N-1) \tag{3.35}$$

$$f(x)=\frac{1}{N}\sum_{u=0}^{N-1}F(u)\mathrm{e}^{\mathrm{j}\frac{2\pi ux}{N}}\quad(x=0,1,\cdots,N-1) \tag{3.36}$$

一般地，取 $\Delta T=1$，则这个离散序列表示为$\{f(0),f(1),\cdots,f(N-1)\}$。一维离散傅里叶变换对的矩阵形式为

$$\begin{bmatrix}F(0)\\F(1)\\\vdots\\F(N-1)\end{bmatrix}=\begin{bmatrix}W^0&W^0&W^0&\cdots&W^0\\W^0&W^{1\times1}&W^{2\times1}&\cdots&W^{(N-1)\times1}\\\vdots&\vdots&\vdots&&\vdots\\W^0&W^{1\times(N-1)}&W^{2\times(N-1)}&\cdots&W^{(N-1)\times(N-1)}\end{bmatrix}\begin{bmatrix}f(0)\\f(1)\\\vdots\\f(N-1)\end{bmatrix} \tag{3.37}$$

$$\begin{bmatrix}f(0)\\f(1)\\\vdots\\f(N-1)\end{bmatrix}=\begin{bmatrix}W^0&W^0&W^0&\cdots&W^0\\W^0&W^{-1\times1}&W^{-2\times1}&\cdots&W^{-(N-1)\times1}\\\vdots&\vdots&\vdots&&\vdots\\W^0&W^{-1\times(N-1)}&W^{-2\times(N-1)}&\cdots&W^{-(N-1)\times(N-1)}\end{bmatrix}\begin{bmatrix}F(0)\\F(1)\\\vdots\\F(N-1)\end{bmatrix} \tag{3.38}$$

式中，$W=\mathrm{e}^{-\mathrm{j}\frac{2\pi}{N}}$ 称为变换核。

下面给出由一维连续傅里叶变换式(3.18)、式(3.19)到一维离散傅里叶变换式(3.35)、式(3.36)的简单推导过程。由恒等式(3.17)可得

$$f(k\Delta T)=\int_{-\infty}^{+\infty}f(x)\delta(x-k\Delta T)\mathrm{d}x\quad(k=0,1,\cdots,N-1) \tag{3.39}$$

其中，$\delta(x)=\begin{cases}1, & x=0\\0, & \text{其他}\end{cases}$。离散后得到

$$f(t)=\sum_{n=-\infty}^{+\infty}f(t)\delta(t-n\Delta T) \tag{3.40}$$

于是得到

$$\begin{aligned}F(u)&=\int_{-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty}f(t)\delta(t-n\Delta T)\mathrm{e}^{-\mathrm{j}2\pi ut}\mathrm{d}t=\sum_{n=-\infty}^{+\infty}\int_{-\infty}^{+\infty}f(t)\delta(t-n\Delta T)\mathrm{e}^{-\mathrm{j}2\pi ut}\mathrm{d}t\\&=\sum_{n=-\infty}^{+\infty}f(n)\mathrm{e}^{-\mathrm{j}2\pi un\Delta T}\end{aligned} \tag{3.41}$$

令 $u=\frac{k}{N\Delta T}$，得到 $F(u)=\sum_{x=0}^{N-1}f(x)\mathrm{e}^{-\mathrm{j}\frac{2\pi ux}{N}}(u=0,1,\cdots,N-1)$，即式(3.35)。类似地，可以得到傅里叶逆变换形式。

【例 3.4】 假设函数 $f(0)=1,f(1)=2,f(2)=4,f(3)=4$，求傅里叶变换 $F(0)$、$F(1)$、$F(2)$、

电子工业出版社 版权所有 盗版必究