# 上机实习\_1

## 任务一

1. 绘制连续的振幅为2、频率为10Hz的 sin正弦信号；绘制振幅为4﹑频率为20Hz ，初相位为50°的cos余弦信号；分别显示出来。

代码：

Fs = 200000.0  
Ts = 2.0/Fs  
t = np.arange(-1,1,Ts)  
fs1 = 10  
y1 = 2\*(np.sin(2\*np.pi\*fs1\*t)) *#sin信号*plt.title(**u'10Hz的正弦信号'**)  
plt.plot(t, y1)  
plt.xlim(-0.5, 0.5)  
plt.axvline(x=0, color=**'k'**)  
plt.axhline(y=0, color=**'k'**)  
plt.show()

fs2 = 20  
phi = (50/180) \* np.pi  
y2 = 4\*(np.cos(2\*np.pi\*fs2\*t + phi))*#cos信号*plt.title(**u'20Hz的余弦信号'**)  
plt.plot(t, y2)  
plt.xlim(-0.5, 0.5)  
plt.axvline(x=0, color=**'k'**)  
plt.axhline(y=0, color=**'k'**)  
plt.show()

运行结果：

|  |  |
| --- | --- |
|  |  |

图 1 10Hz的sin信号和20Hz的cos信号

结果分析：

当采样率过低时，会出现信号变形，无法显示正确的图形。

|  |  |
| --- | --- |
|  |  |

图 2 采样率为50Hz的信号显示

1. 绘制上述两个信号相加的结果，显示出来。

代码：

plt.title(**u'相加信号'**)  
plt.plot(t,y1+y2)  
plt.xlim(-0.5,0.5)  
plt.axvline(x=0, color=**'k'**)  
plt.axhline(y=0, color=**'k'**)  
plt.show()

运行结果：

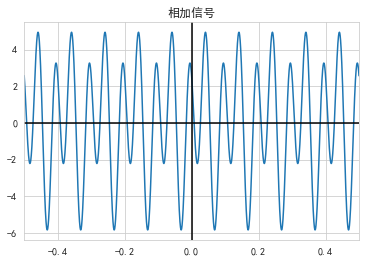


图 3 相加信号

1. 绘制连续信号Si(t)，并显示。

代码：

X = np.arange(-20, 20, 0.01)  
Y = []  
**def** f(x):  
 **return** np.sin(x)/x  
**for** i **in** X:  
 v, err = integrate.quad(f, 0, i, points=[0])  
 Y.append(v)plt.title(**u'Si(t)信号'**)  
plt.plot(X,Y)  
plt.axvline(x=0, color=**'k'**)  
plt.axhline(y=0, color=**'k'**)  
plt.show()

运行结果：

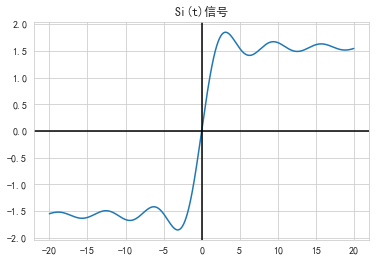


图 4 Si(t)信号

1. 已知f(t)的波形如图所示，试画出f(3-2t)的波形。

代码：

**def** fun(x,x1,x2):  
 **if** x < x1:  
 **return** (-2)\*x+1  
 **elif** x >= x2:  
 **return** (-2)\*x+2  
 **else**:  
 **return** 1  
X = np.linspace(-1/2, 1, 1000)  
Y = np.array([fun(t, 0, 1/2) **for** t **in** X])  
plt.plot(X,Y)  
plt.title(**u'f(3-2t)'**)  
plt.axvline(x=0, color=**'k'**)  
plt.axhline(y=0, color=**'k'**)  
plt.show()

运行结果：

|  |  |
| --- | --- |
|  |  |

图 5 f(t)和f(3-2t)

## 任务二

1. 设计一个周期方波信号并画出信号波形(图中展示3个周期)﹔

代码：

N1 = 20000  
t = np.linspace(-1, 1, N1, endpoint=**False**)  
sq = signal.square(2 \* np.pi \* 2 \* t) *#square函数生成方波  
  
# 方波信号*plt.plot(t, sq)  
plt.xlim(-0.75, 0.75)  
plt.title(**u'方波信号-4Hz 采样10000Hz'**)  
plt.axvline(x=0, color=**'k'**)  
plt.axhline(y=0, color=**'k'**)  
plt.ylabel(**'Amplitude'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**'Time'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.show()

运行结果：

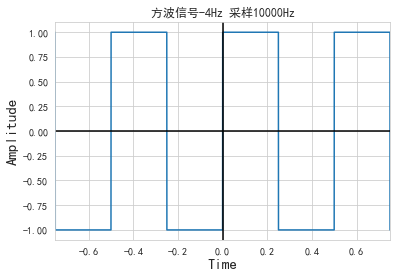


图 6 周期方波信号

1. 分别画出周期方波信号的三角型和复数型傅里叶级数频谱(用stem函数绘制)，要求用已知方波的频谱表达式;

代码：

n = [2\*i+1 **for** i **in** range(10)]  
w = []  
A\_n = []  
phi = []  
**for** i **in** n:  
 an = 4/math.pi \* (1/i)  
 A\_n.append(an)  
 wn = 4\*math.pi \* i  
 w.append(wn)  
 phi.append(0)  
  
*# 三角型频谱图*plt.stem(w,A\_n, label=**"all"**)  
plt.xlim(0, 100)  
plt.axhline(y=0,ls=**"-"**,c=**"red"**)  
plt.title(**u'方波信号-4Hz 三角型频谱'**)  
plt.ylabel(**u'An'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**u'ω'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.show()  
  
*# 复数型频谱图*plt.stem(w,phi, label=**"all"**)  
plt.xlim(0, 100)  
plt.axhline(y=0,ls=**"-"**,c=**"red"**)  
plt.title(**u'方波信号-4Hz 复数型频谱'**)  
plt.ylabel(**u'φ'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**u'ω'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.ylim(-0.1,1.2)  
plt.show()

运行结果：

|  |  |
| --- | --- |
|  |  |

图 7 周期方波信号的三角型频谱与复数型频谱

1. 设计一个周期三角信号并画出（图中展示3个周期);

代码：

N2 = 40000  
x = np.linspace(-2, 2, N2, endpoint=**False**)  
y = -signal.sawtooth(2 \* np.pi \* 5 \* x, 0.5)+1  
  
plt.plot(x, y)  
plt.xlim(-0.3, 0.3)  
plt.axvline(x=0, color=**'k'**)  
plt.axhline(y=0, color=**'k'**)  
plt.title(**'三角信号-5Hz 采样10000Hz'**)  
plt.ylabel(**'Amplitude'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**'Time'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.show()

运行结果：

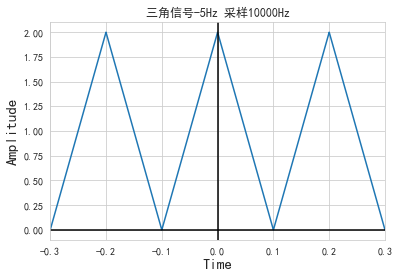


图 8 周期三角信号

1. 画出周期三角信号的三角型和复数型傅里叶级数频谱。

代码：

n = [2\*i+1 **for** i **in** range(10)]  
w = [0]  
A\_n = [1]  
phi = [0]  
**for** i **in** n:  
 an = 4\*2/math.pi\*\*2 \* (1/i\*\*2)  
 A\_n.append(an)  
 wn = 10\*math.pi \* i  
 w.append(wn)  
 phi.append(0)  
  
*# 三角型频谱图*plt.stem(w,A\_n, label=**"all"**)  
plt.xlim(0, 200)  
plt.axhline(y=0,ls=**"-"**,c=**"red"**)  
plt.title(**u'三角信号-5Hz 三角型频谱'**)  
plt.ylabel(**u'An'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**u'ω'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.show()  
  
*# 复数型频谱图*plt.stem(w,phi, label=**"all"**)  
plt.xlim(0, 200)  
plt.axhline(y=0,ls=**"-"**,c=**"red"**)  
plt.title(**u'三角信号-5Hz 复数型频谱'**)  
plt.ylabel(**u'An'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**u'φ'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.ylim(-0.12, 1.2)  
plt.show()

运行结果：

|  |  |
| --- | --- |
|  |  |

图 9 周期方波信号的三角型频谱与复数型频谱

## 任务三

1. 将任务二中的周期方波信号的T0增大到4T0。(保持脉宽不变)，画出信号；画出周期方波信号的频谱;

代码：

N1 = 80000  
t = np.linspace(-4, 4, N1, endpoint=**False**)  
sq = signal.square(2 \* np.pi \* 2/4 \* t) *#square函数生成方波  
  
# 方波信号*plt.plot(t, sq)  
plt.xlim(-2.5, 2.5)  
plt.title(**u'方波信号-1Hz 采样10000Hz'**)  
plt.axvline(x=0, color=**'k'**)  
plt.axhline(y=0, color=**'k'**)  
plt.ylabel(**'Amplitude'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**'Time'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.show()  
  
n = [2\*i+1 **for** i **in** range(100)]  
w = []  
A\_n = []  
phi = []  
**for** i **in** n:  
 an = 4/math.pi \* (1/i)  
 A\_n.append(an)  
 wn = math.pi \* i  
 w.append(wn)  
 phi.append(0)  
  
*# 三角型频谱图*plt.stem(w,A\_n, label=**"all"**)  
plt.xlim(0, 100)  
plt.axhline(y=0,ls=**"-"**,c=**"red"**)  
plt.title(**u'方波信号-1Hz 三角型频谱'**)  
plt.ylabel(**u'An'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**u'ω'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.show()  
  
*# 复数型频谱图*plt.stem(w,phi, label=**"all"**)  
plt.xlim(0, 100)  
plt.axhline(y=0,ls=**"-"**,c=**"red"**)  
plt.title(**u'方波信号-1Hz 复数型频谱'**)  
plt.ylabel(**u'φ'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**u'ω'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.ylim(-0.12, 1.2)  
plt.show()

运行结果：

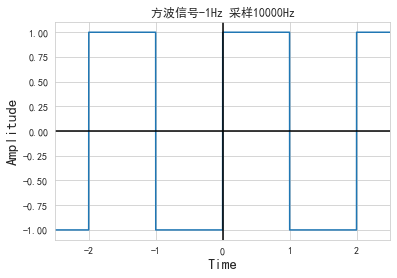


图 10 4T0周期方波信号

|  |  |
| --- | --- |
|  |  |

图 11 4T0周期方波信号的三角型频谱与复数型频谱

1. 分别将周期方波信号的周期扩大到16T0和64T0，并画出周期方波信号和相应的频谱

1）16T0

代码：

N1 = 320000  
t = np.linspace(-16, 16, N1, endpoint=**False**)  
sq = signal.square(2 \* np.pi \* 2/16 \* t) *#square函数生成方波  
  
# 方波信号*plt.plot(t, sq)  
plt.xlim(-10, 10)  
plt.title(**u'方波信号-1/4Hz 采样10000Hz'**)  
plt.axvline(x=0, color=**'k'**)  
plt.axhline(y=0, color=**'k'**)  
plt.ylabel(**'Amplitude'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**'Time'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.show()  
  
n = [2\*i+1 **for** i **in** range(1000)]  
w = []  
A\_n = []  
phi = []  
**for** i **in** n:  
 an = 4/math.pi \* (1/i)  
 A\_n.append(an)  
 wn = 1/4 \* math.pi \* i  
 w.append(wn)  
 phi.append(0)  
  
*# 三角型频谱图*plt.stem(w,A\_n, label=**"all"**)  
plt.xlim(0, 100)  
plt.axhline(y=0,ls=**"-"**,c=**"red"**)  
plt.title(**u'方波信号-1/4Hz 三角型频谱'**)  
plt.ylabel(**u'An'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**u'ω'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.show()  
  
*# 复数型频谱图*plt.stem(w,phi, label=**"all"**)  
plt.xlim(0, 100)  
plt.axhline(y=0,ls=**"-"**,c=**"red"**)  
plt.title(**u'方波信号-1/4Hz 复数型频谱'**)  
plt.ylabel(**u'φ'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**u'ω'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.ylim(-0.12, 1.2)  
plt.show()

运行结果：

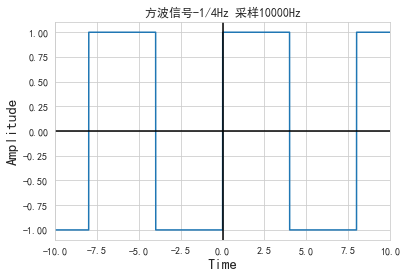


图 12 16T0周期方波信号

|  |  |
| --- | --- |
|  |  |

图 13 16T0周期方波信号的三角型频谱与复数型频谱

2）64T0

代码：

N1 = 500000  
t = np.linspace(-50, 50, N1, endpoint=**False**)  
sq = signal.square(2 \* np.pi \* 2/64 \* t) *#square函数生成方波  
  
# 方波信号*plt.plot(t, sq)  
plt.xlim(-45, 45)  
plt.title(**u'方波信号-1/16Hz 采样10000Hz'**)  
plt.axvline(x=0, color=**'k'**)  
plt.axhline(y=0, color=**'k'**)  
plt.ylabel(**'Amplitude'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**'Time'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.show()  
  
n = [2\*i+1 **for** i **in** range(1000)]  
w = []  
A\_n = []  
phi = []  
**for** i **in** n:  
 an = 4/math.pi \* (1/i)  
 A\_n.append(an)  
 wn = 1/16\*math.pi \* i  
 w.append(wn)  
 phi.append(0)  
  
*# 三角频谱图*plt.stem(w,A\_n, label=**"all"**)  
plt.xlim(0, 100)  
plt.axhline(y=0,ls=**"-"**,c=**"red"**)  
plt.title(**u'方波信号-1/16Hz 三角型频谱'**)  
plt.ylabel(**u'An'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**u'ω'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.show()  
  
*# 复数频谱图*plt.stem(w,phi, label=**"all"**)  
plt.xlim(0, 100)  
plt.axhline(y=0,ls=**"-"**,c=**"red"**)  
plt.title(**u'方波信号-1/16Hz 复数型频谱'**)  
plt.ylabel(**u'φ'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.xlabel(**u'ω'**, fontdict={**'weight'**:**'normal'**,**'size'**: 15})  
plt.ylim(-0.12, 1.2)  
plt.show()

运行结果：

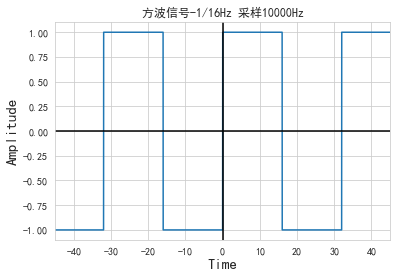


图 14 64T0周期方波信号

|  |  |
| --- | --- |
|  |  |

图 15 16T0周期方波信号的三角型频谱与复数型频谱

1. 比较不同周期的方波信号的频谱，并加以说明

在频谱图中，可以看出取一定时间间隔T，当周期τ变大，两点之间的谱线数目：增多。

离散性:频谱由不连续的线条组成，每一条线代表一个正弦量，故称为离散频谱。时域的周期性对应于频域的离散性。谐波性:频谱的每条谱线只能出现在基波频率的整数倍频率上。收敛性:各次谐波的振幅，总的趋势是随着谐波次数的增高而逐渐减小。

当周期T增大，频谱也相应地渐趋密集，频谱的幅度也相应的渐趋减小。当T→时，频谱线无限密集，频谱幅度无限趋小。这时，离散频谱就变成连续频谱。

附录：

1. #!/usr/bin/env python
2. # coding: utf-8
4. # In[2]:

7. **import** math
8. **import** numpy as np
9. **import** matplotlib.pyplot as plt
10. **from** scipy **import** integrate
11. **from** matplotlib **import** pyplot
13. pyplot.rcParams['font.sans-serif'] = ['SimHei']
14. pyplot.rcParams['axes.unicode\_minus'] = False

17. # # 任务一
19. # In[3]:

22. Fs = 200000.0
23. Ts = 2.0/Fs
24. t = np.arange(-1,1,Ts)
25. fs1 = 10
27. y1 = 2\*(np.sin(2\*np.pi\*fs1\*t)) #sin信号
28. plt.title(u'10Hz的正弦信号')
29. plt.plot(t, y1)
30. plt.xlim(-0.5, 0.5)
31. plt.axvline(x=0, color='k')
32. plt.axhline(y=0, color='k')
33. plt.show()

36. # In[4]:

39. fs2 = 20
40. phi = (50/180) \* np.pi
41. y2 = 4\*(np.cos(2\*np.pi\*fs2\*t + phi))#cos信号
42. # plt.axhline(y=0,ls=":",c="red")
43. plt.title(u'20Hz的余弦信号')
44. plt.plot(t, y2)
45. plt.xlim(-0.5, 0.5)
46. plt.axvline(x=0, color='k')
47. plt.axhline(y=0, color='k')
48. plt.show()

51. # In[5]:

54. # plt.axhline(y=0,ls=":",c="red")
55. plt.title(u'相加信号')
56. plt.plot(t,y1+y2)
57. plt.xlim(-0.5,0.5)
58. plt.axvline(x=0, color='k')
59. plt.axhline(y=0, color='k')
60. plt.show()

63. # In[6]:

66. X = np.arange(-20, 20, 0.01)
67. Y = []
68. **def** f(x):
69. **return** np.sin(x)/x
70. **for** i **in** X:
71. v, err = integrate.quad(f, 0, i, points=[0])
72. Y.append(v)
73. # plt.axhline(y=0,ls=":",c="red")
74. plt.title(u'Si(t)信号')
75. plt.plot(X,Y)
76. plt.axvline(x=0, color='k')
77. plt.axhline(y=0, color='k')
78. plt.show()

81. # In[7]:

84. **def** fun(x,x1,x2):
85. **if** x < x1:
86. **return** x-1
87. **elif** x >= x2:
88. **return** x-2
89. **else**:
90. **return** 1
91. X = np.linspace(1, 4, 3000)
92. Y = np.array([fun(t, 2, 3) **for** t **in** X])
93. plt.title(u'f(t)')
94. plt.plot(X,Y)
95. plt.axvline(x=0, color='k')
96. plt.axhline(y=0, color='k')
97. plt.show()

100. # In[8]:

103. **def** fun(x,x1,x2):
104. **if** x < x1:
105. **return** (-2)\*x+1
106. **elif** x >= x2:
107. **return** (-2)\*x+2
108. **else**:
109. **return** 1
110. X = np.linspace(-1/2, 1, 1000)
111. Y = np.array([fun(t, 0, 1/2) **for** t **in** X])
112. plt.plot(X,Y)
113. plt.title(u'f(3-2t)')
114. plt.axvline(x=0, color='k')
115. plt.axhline(y=0, color='k')
116. plt.show()

119. # # 任务二
120. # T0 = 0.5
122. # In[9]:

125. **import** numpy as np
126. **from** scipy **import** fftpack,signal
127. **import** matplotlib.pyplot as plt
129. N1 = 20000
130. t = np.linspace(-1, 1, N1, endpoint=False)
131. sq = signal.square(2 \* np.pi \* 2 \* t) #square函数生成方波
133. # 方波信号
134. plt.plot(t, sq)
135. plt.xlim(-0.75, 0.75)
136. plt.title(u'方波信号-4Hz 采样10000Hz')
137. plt.axvline(x=0, color='k')
138. plt.axhline(y=0, color='k')
139. plt.ylabel('Amplitude', fontdict={'weight':'normal','size': 15})
140. plt.xlabel('Time', fontdict={'weight':'normal','size': 15})
141. plt.show()
143. n = [2\*i+1 **for** i **in** range(10)]
144. w = []
145. A\_n = []
146. phi = []
147. **for** i **in** n:
148. an = 4/math.pi \* (1/i)
149. A\_n.append(an)
150. wn = 4\*math.pi \* i
151. w.append(wn)
152. phi.append(0)
154. # 三角型频谱图
155. plt.stem(w,A\_n, label="all")
156. plt.xlim(0, 100)
157. plt.axhline(y=0,ls="-",c="red")
158. plt.title(u'方波信号-4Hz 三角型频谱')
159. plt.ylabel(u'An', fontdict={'weight':'normal','size': 15})
160. plt.xlabel(u'ω', fontdict={'weight':'normal','size': 15})
161. plt.show()
163. # 复数型频谱图
164. plt.stem(w,phi, label="all")
165. plt.xlim(0, 100)
166. plt.axhline(y=0,ls="-",c="red")
167. plt.title(u'方波信号-4Hz 复数型频谱')
168. plt.ylabel(u'φ', fontdict={'weight':'normal','size': 15})
169. plt.xlabel(u'ω', fontdict={'weight':'normal','size': 15})
170. plt.ylim(-0.1,1.2)
171. plt.show()

174. # In[10]:

177. **import** numpy as np
178. **from** scipy **import** signal
179. **import** matplotlib.pyplot as plt
181. N2 = 40000
182. x = np.linspace(-2, 2, N2, endpoint=False)
183. y = -signal.sawtooth(2 \* np.pi \* 5 \* x, 0.5)+1
185. plt.plot(x, y)
186. plt.xlim(-0.3, 0.3)
187. plt.axvline(x=0, color='k')
188. plt.axhline(y=0, color='k')
189. plt.title('三角信号-5Hz 采样10000Hz')
190. plt.ylabel('Amplitude', fontdict={'weight':'normal','size': 15})
191. plt.xlabel('Time', fontdict={'weight':'normal','size': 15})
192. plt.show()
194. n = [2\*i+1 **for** i **in** range(10)]
195. w = [0]
196. A\_n = [1]
197. phi = [0]
198. **for** i **in** n:
199. an = 4\*2/math.pi\*\*2 \* (1/i\*\*2)
200. A\_n.append(an)
201. wn = 10\*math.pi \* i
202. w.append(wn)
203. phi.append(0)
205. # 三角型频谱图
206. plt.stem(w,A\_n, label="all")
207. plt.xlim(0, 200)
208. plt.axhline(y=0,ls="-",c="red")
209. plt.title(u'三角信号-5Hz 三角型频谱')
210. plt.ylabel(u'An', fontdict={'weight':'normal','size': 15})
211. plt.xlabel(u'ω', fontdict={'weight':'normal','size': 15})
212. plt.show()
214. # 复数型频谱图
215. plt.stem(w,phi, label="all")
216. plt.xlim(0, 200)
217. plt.axhline(y=0,ls="-",c="red")
218. plt.title(u'三角信号-5Hz 复数型频谱')
219. plt.ylabel(u'φ', fontdict={'weight':'normal','size': 15})
220. plt.xlabel(u'ω', fontdict={'weight':'normal','size': 15})
221. plt.ylim(-0.12, 1.2)
222. plt.show()

225. # # 任务三
227. # ## 任务2中T0
229. # In[11]:

232. **import** numpy as np
233. **from** scipy **import** fftpack,signal
234. **import** matplotlib.pyplot as plt
236. N1 = 20000
237. t = np.linspace(-1, 1, N1, endpoint=False)
238. sq = signal.square(2 \* np.pi \* 2 \* t) #square函数生成方波
240. # 方波信号
241. plt.plot(t, sq)
242. plt.xlim(-0.75, 0.75)
243. plt.title(u'方波信号-4Hz 采样10000Hz')
244. plt.axvline(x=0, color='k')
245. plt.axhline(y=0, color='k')
246. plt.ylabel('Amplitude', fontdict={'weight':'normal','size': 15})
247. plt.xlabel('Time', fontdict={'weight':'normal','size': 15})
248. plt.show()

251. # ## 4T0
253. # In[23]:

256. **import** numpy as np
257. **from** scipy **import** fftpack,signal
258. **import** matplotlib.pyplot as plt
260. N1 = 80000
261. t = np.linspace(-4, 4, N1, endpoint=False)
262. sq = signal.square(2 \* np.pi \* 2/4 \* t) #square函数生成方波
264. # 方波信号
265. plt.plot(t, sq)
266. plt.xlim(-2.5, 2.5)
267. plt.title(u'方波信号-1Hz 采样10000Hz')
268. plt.axvline(x=0, color='k')
269. plt.axhline(y=0, color='k')
270. plt.ylabel('Amplitude', fontdict={'weight':'normal','size': 15})
271. plt.xlabel('Time', fontdict={'weight':'normal','size': 15})
272. plt.show()
274. n = [2\*i+1 **for** i **in** range(100)]
275. w = []
276. A\_n = []
277. phi = []
278. **for** i **in** n:
279. an = 4/math.pi \* (1/i)
280. A\_n.append(an)
281. wn = math.pi \* i
282. w.append(wn)
283. phi.append(0)
285. # 三角型频谱图
286. plt.stem(w,A\_n, label="all")
287. plt.xlim(0, 100)
288. plt.axhline(y=0,ls="-",c="red")
289. plt.title(u'方波信号-1Hz 三角型频谱')
290. plt.ylabel(u'An', fontdict={'weight':'normal','size': 15})
291. plt.xlabel(u'ω', fontdict={'weight':'normal','size': 15})
292. plt.show()
294. # 复数型频谱图
295. plt.stem(w,phi, label="all")
296. plt.xlim(0, 100)
297. plt.axhline(y=0,ls="-",c="red")
298. plt.title(u'方波信号-1Hz 复数型频谱')
299. plt.ylabel(u'φ', fontdict={'weight':'normal','size': 15})
300. plt.xlabel(u'ω', fontdict={'weight':'normal','size': 15})
301. plt.ylim(-0.12, 1.2)
302. plt.show()

305. # ## 16T0
307. # In[18]:

310. **import** numpy as np
311. **from** scipy **import** fftpack,signal
312. **import** matplotlib.pyplot as plt
314. N1 = 320000
315. t = np.linspace(-16, 16, N1, endpoint=False)
316. sq = signal.square(2 \* np.pi \* 2/16 \* t) #square函数生成方波
318. # 方波信号
319. plt.plot(t, sq)
320. plt.xlim(-10, 10)
321. plt.title(u'方波信号-1/4Hz 采样10000Hz')
322. plt.axvline(x=0, color='k')
323. plt.axhline(y=0, color='k')
324. plt.ylabel('Amplitude', fontdict={'weight':'normal','size': 15})
325. plt.xlabel('Time', fontdict={'weight':'normal','size': 15})
326. plt.show()
328. n = [2\*i+1 **for** i **in** range(1000)]
329. w = []
330. A\_n = []
331. phi = []
332. **for** i **in** n:
333. an = 4/math.pi \* (1/i)
334. A\_n.append(an)
335. wn = 1/4 \* math.pi \* i
336. w.append(wn)
337. phi.append(0)
339. # 三角型频谱图
340. plt.stem(w,A\_n, label="all")
341. plt.xlim(0, 100)
342. plt.axhline(y=0,ls="-",c="red")
343. plt.title(u'方波信号-1/4Hz 三角型频谱')
344. plt.ylabel(u'An', fontdict={'weight':'normal','size': 15})
345. plt.xlabel(u'ω', fontdict={'weight':'normal','size': 15})
346. plt.show()
348. # 复数型频谱图
349. plt.stem(w,phi, label="all")
350. plt.xlim(0, 100)
351. plt.axhline(y=0,ls="-",c="red")
352. plt.title(u'方波信号-1/4Hz 复数型频谱')
353. plt.ylabel(u'φ', fontdict={'weight':'normal','size': 15})
354. plt.xlabel(u'ω', fontdict={'weight':'normal','size': 15})
355. plt.ylim(-0.12, 1.2)
356. plt.show()

359. # ## 64T0
361. # In[22]:

364. **import** numpy as np
365. **from** scipy **import** fftpack,signal
366. **import** matplotlib.pyplot as plt
368. N1 = 500000
369. t = np.linspace(-50, 50, N1, endpoint=False)
370. sq = signal.square(2 \* np.pi \* 2/64 \* t) #square函数生成方波
372. # 方波信号
373. plt.plot(t, sq)
374. plt.xlim(-45, 45)
375. plt.title(u'方波信号-1/16Hz 采样10000Hz')
376. plt.axvline(x=0, color='k')
377. plt.axhline(y=0, color='k')
378. plt.ylabel('Amplitude', fontdict={'weight':'normal','size': 15})
379. plt.xlabel('Time', fontdict={'weight':'normal','size': 15})
380. plt.show()
382. n = [2\*i+1 **for** i **in** range(1000)]
383. w = []
384. A\_n = []
385. phi = []
386. **for** i **in** n:
387. an = 4/math.pi \* (1/i)
388. A\_n.append(an)
389. wn = 1/16\*math.pi \* i
390. w.append(wn)
391. phi.append(0)
393. # 三角频谱图
394. plt.stem(w,A\_n, label="all")
395. plt.xlim(0, 100)
396. plt.axhline(y=0,ls="-",c="red")
397. plt.title(u'方波信号-1/16Hz 三角型频谱')
398. plt.ylabel(u'An', fontdict={'weight':'normal','size': 15})
399. plt.xlabel(u'ω', fontdict={'weight':'normal','size': 15})
400. plt.show()
402. # 复数频谱图
403. plt.stem(w,phi, label="all")
404. plt.xlim(0, 100)
405. plt.axhline(y=0,ls="-",c="red")
406. plt.title(u'方波信号-1/16Hz 复数型频谱')
407. plt.ylabel(u'φ', fontdict={'weight':'normal','size': 15})
408. plt.xlabel(u'ω', fontdict={'weight':'normal','size': 15})
409. plt.ylim(-0.12, 1.2)
410. plt.show()