Numpy作业

魏屹海　　202218019427001　　中国科学院空天信息创新研究院

1、 依照课件3中的内容给出测算三角波（triangle_wave（））y1、y2、y3、y4四种方式的计算速度与结果比较的代码，并对其运算显示结果。

```
import numpy as np
import time


def triangle_wave(x, c, c0, hc):
    x = x - x.astype(int)  # 三角波的周期为1，因此只取x坐标的小数部分进行计算
    return np.where(x >= c, 0, np.where(x < c0, x / c0 * hc, (c - x) / (c - c0) * hc))


def triangle_wave2(x, c, c0, hc):
    x = x - x.astype(int)
    return np.select([x >= c, x < c0, True], [0, x / c0 * hc, (c - x) / (c - c0) * hc])


def triangle_wave3(x, c, c0, hc):
    x = x - x.astype(int)
    return np.piecewise(x,
                        [x >= c, x < c0],
                        [0,  # x>=c
                         lambda x: x / c0 * hc,  # x<c0
                         lambda x: (c - x) / (c - c0) * hc])  # else


def triangle_wave4(x, c, c0, hc):
    """显示每个分段函数计算的数据点数"""

    def f1(x):
        return x / c0 * hc

    def f2(x):
        return (c - x) / (c - c0) * hc

    x = x - x.astype(int)
    return np.piecewise(x, [x >= c, x < c0], [0, f1, f2])


x = np.linspace(0, 2, 1000)

start = time.perf_counter()
y = triangle_wave(x, 0.6, 0.4, 1.0)
print("y:time of using where nest ,  :", time.perf_counter() - start)
```

```
start = time.perf_counter()
y2 = triangle_wave2(x, 0.6, 0.4, 1.0)
print("y2:time of using select ,  :", time.perf_counter() - start)

start = time.perf_counter()
y3 = triangle_wave3(x, 0.6, 0.4, 1.0)
print("y3:time of using piecewise :", time.perf_counter() - start)

start = time.perf_counter()
y4 = triangle_wave4(x, 0.6, 0.4, 1.0)
print("y4:time of using traditional selection struction :", time.perf_counter() -
start)

if np.alltrue(y == y2) and np.alltrue(y2 == y3) and np.alltrue(y3 == y4):
    print("The four ways of Generating triangular wave have the same result")
```

结果如下

```
y:time of using where nest ,  : 3.31669999695805e-05
y2:time of using select ,  : 9.775000000900036e-05
y3:time of using piecewise : 0.00010045900000932306
y4:time of using traditional selection struction : 4.6624999981759174e-05
The four ways of Generating triangular wave have the same result
```

2、arr11 = 5-np.arange(1,13).reshape(4,3), 计算所有元素及每一列的和；对每一个元素、每一列求累积和；计算每一行的累计积；计算所有元素的最小值；计算每一列的最大值；计算所有元素、每一行的均值；计算所有元素、每一列的中位数；计算所有元素的方差，每一行的标准差。

```
import numpy as np

arr11 = 5 - np.arange(1, 13).reshape(4, 3)

print("The original array formula is ",arr11)
# 1 Compute the sum of all elements and each column
print("Sum of all the elements", np.sum(arr11))
print("Sum of each column", np.sum(arr11, axis=0))
# 2 Compute the cumulative sum of each element and each column
print("Cumulative sum of all elements:", np.cumsum(arr11))
print("Cumulative sum of each column:", np.cumsum(arr11, axis=0))
# 3 Compute the cumulative product of each row
print("Cumulative product of each row:", np.cumprod(arr11, axis=1))
# 4 Compute the minimum value of all element
print("Minimum value of all elements:", np.min(arr11))
# 5 Compute the maximum value of each column
```

```
print("Maximum value of each column:", np.max(arr11, axis=0))
# 6 Compute the mean of all elements and each row
print("Mean of all elements:", np.mean(arr11))
print("Mean of each row:", np.mean(arr11, axis=1))
# 7 Compute the median of all elements and each column
print("Median of all elements:", np.median(arr11))
print("Median of each column:", np.median(arr11, axis=0))
# 8 Compute the variance of all elements and the standard deviation of each row
print("Variance of all elements:", np.var(arr11))
print("Standard deviation of each row:", np.std(arr11, axis=1))
```

运行结果如下：

```
The original array formula is  [[ 4  3  2]
 [ 1  0 -1]
 [-2 -3 -4]
 [-5 -6 -7]]
Sum of all the elements -18
Sum of each column [ -2  -6 -10]
Cumulative sum of all elements: [  4   7   9  10  10   9   7   4   0  -5 -11 -18]
Cumulative sum of each column: [[  4   3   2]
 [  5   3   1]
 [  3   0  -3]
 [ -2  -6 -10]]
Cumulative product of each row: [[   4   12   24]
 [   1    0    0]
 [  -2    6  -24]
 [  -5   30 -210]]
Minimum value of all elements: -7
Maximum value of each column: [4 3 2]
Mean of all elements: -1.5
Mean of each row: [ 3.  0. -3. -6.]
Median of all elements: -1.5
Median of each column: [-0.5 -1.5 -2.5]
Variance of all elements: 11.916666666666666
Standard deviation of each row: [0.81649658 0.81649658 0.81649658 0.81649658]
```

3、在数组[1, 2, 3, 4, 5]中每相邻两个数字中间插入两个0。

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5])
zeros = np.zeros((len(arr)-1)*2, dtype=int)
result = np.insert(zeros, np.arange(0, len(arr))*2, arr)
print(result)
```

运行结果如下：

```
[1 0 0 2 0 0 3 0 0 4 0 0 5]

进程已结束，退出代码0
```

4、归一化，将矩阵规格化到0～1，即最小的变成0，最大的变成1，最小与最大之间的等比缩放。试对Z = np.random.random((5,5))进行归一化。

```
import numpy as np

Z = np.random.random((5, 5))
Z_min, Z_max = Z.min(), Z.max()
Z_norm = (Z - Z_min) / (Z_max - Z_min)

print(Z_norm)
```

运行结果：

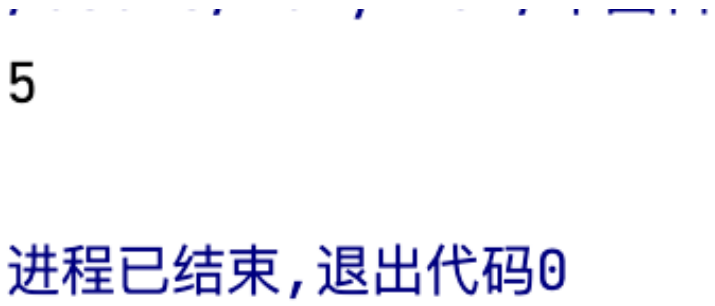
```
[[0.01934209 0.23625882 0.95251747 0.01773569 0.14881274]
 [0.         0.91024705 0.74421193 0.3818578  0.81662593]
 [0.11899335 0.92622149 0.82279753 1.         0.18806805]
 [0.34627047 0.4911435  0.86418172 0.18871102 0.71887652]
 [0.63419784 0.85504681 0.04834994 0.00708782 0.95799037]]
```

5、找出数组中与给定值最接近的数（通用方法）。（例：任一数组Z=array([[0,1,2,3],[4,5,6,7]])，给任一定值z=5.1，如何找出Z中的5）

```
import numpy as np

# 定义数组
Z = np.array([[0, 1, 2, 3], [4, 5, 6, 7]])

# 给定值
z = 5.1
```

```
# 将数组展平成一维数组
Z_flat = Z.flatten()

# 计算每个元素与给定值之间的差值的绝对值
diff = np.abs(Z_flat - z)

# 找到差值的最小值及其索引
min_diff_index = np.argmin(diff)
min_diff_value = diff[min_diff_index]

# 返回原数组中对应索引的元素
result = Z_flat[min_diff_index]

print(result) # 输出：5
```

运行结果：



6、解方程：3x + 6y -5z = 12；x-3y+2z = -2；5x -y +4z = 10。

```
import numpy as np

# 定义系数矩阵A和常数向量b
A = np.array([[3, 6, -5], [1, -3, 2], [5, -1, 4]])
b = np.array([12, -2, 10])

# 解方程
x = np.linalg.solve(A, b)

print(x)
```

运行结果：

```
/Users/weiyihai/中国科
[1.75 1.75 0.75]

进程已结束,退出代码0
```

7、参见课件4第45页，对g(y)在100个切比雪夫节点之上分别使用Polynomial（Polynomial.fit）和Chebyshev（Chebyshev.fit）进行插值，在[-1,1]区间上取1000个等距点对误差进行比较。g(x)= sin(z2) + sin(z)**2, 其中z=(x - 1) * 5。**

```
import numpy as np
from matplotlib import pyplot as plt
from numpy.polynomial import Chebyshev


def g(_):
    z = (_ - 1) * 5
    return np.sin(z ** 2) + np.sin(z) ** 2


n = 101
# Generate node
x = np.linspace(-1, 1, n)
cheb_nodes = Chebyshev.basis(n).roots()
poly_nodes = np.polynomial.Polynomial.basis(n).roots()
xd = np.linspace(-1, 1, 1000)

# Interpolation calculate
cheb_fit = Chebyshev.fit(cheb_nodes, g(cheb_nodes), n - 1, domain=[-1, 1])
# poly_fit = np.polynomial.Polynomial.fit(x,g(x),n-1,domain=[-1,1])
poly_fit1 = np.polynomial.Polynomial.fit(poly_nodes, g(poly_nodes), n - 1, domain=[-1,
1])
# Calculate error
cheb_error = np.abs(g(xd) - cheb_fit(xd)).max()
poly_error = np.abs(g(xd) - poly_fit1(xd)).max()

print(u"插值多项式的最大误差："),
print(u"ploynomial点：", poly_error),
print(u"chebyshev：", cheb_error)
# Draw picture
plt.plot(xd, g(xd), label='g(x)')
plt.plot(xd, cheb_fit(xd), label='Chebyshev fit')
```

```
plt.plot(xd, poly_fit1(xd), label='Polynomial fit')
plt.legend()

plt.figure()
plt.bar(['Chebyshev', 'Polynomial'], [cheb_error, poly_error])
plt.ylabel('Maximum error')
plt.show()
```

运行结果：

```
/Users/weiyihai/中国科学院/中国科学院大学/国科大博一下学期资料数据
/Users/weiyihai/中国科学院/中国科学院大学/国科大博一下学期资料数据
  return pu._fit(polyvander, x, y, deg, rcond, full, w)
插值多项式的最大误差：
ploynomial点:  1.768064992194896
chebyshev:  4.5918729651983625e-09
```

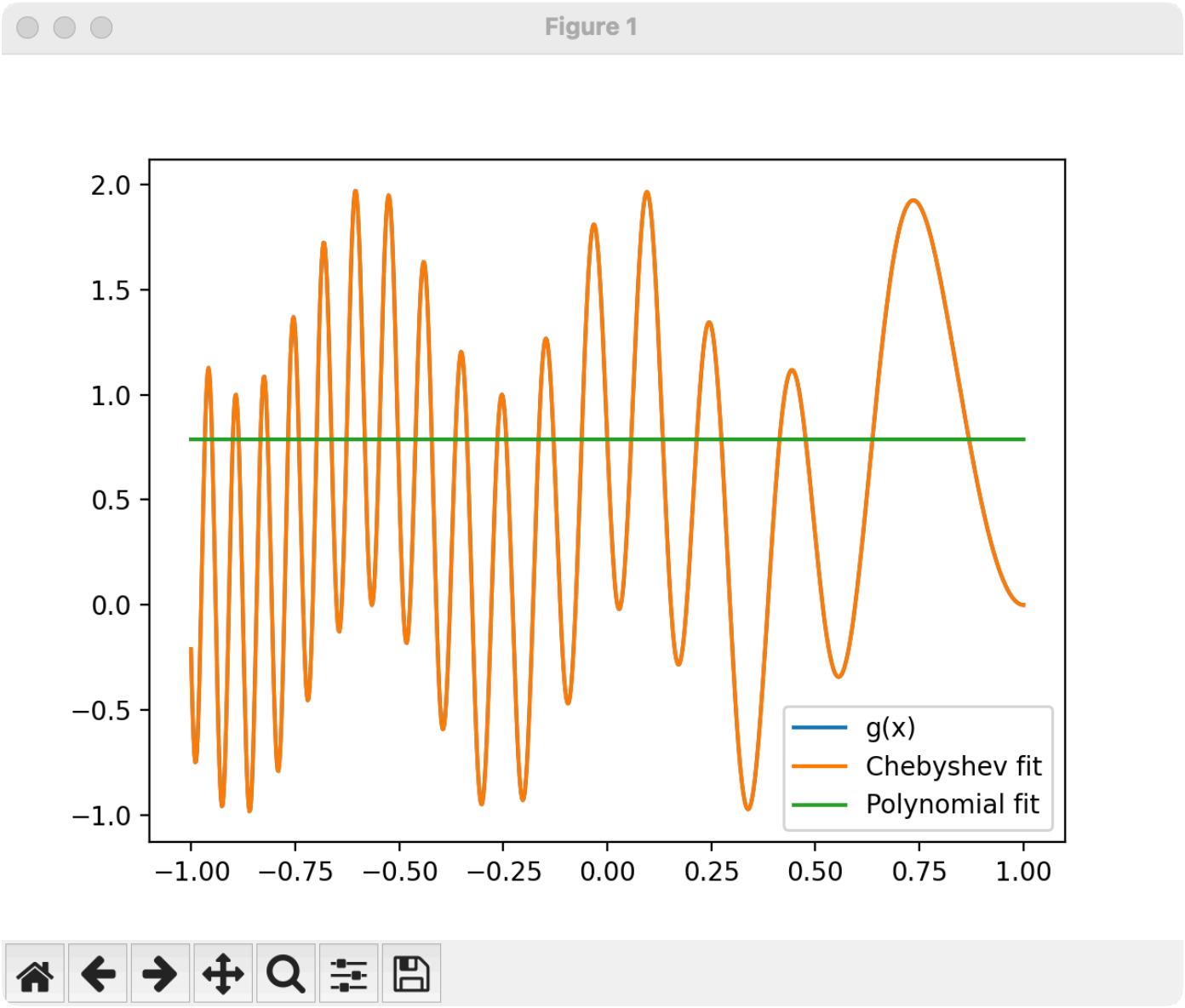
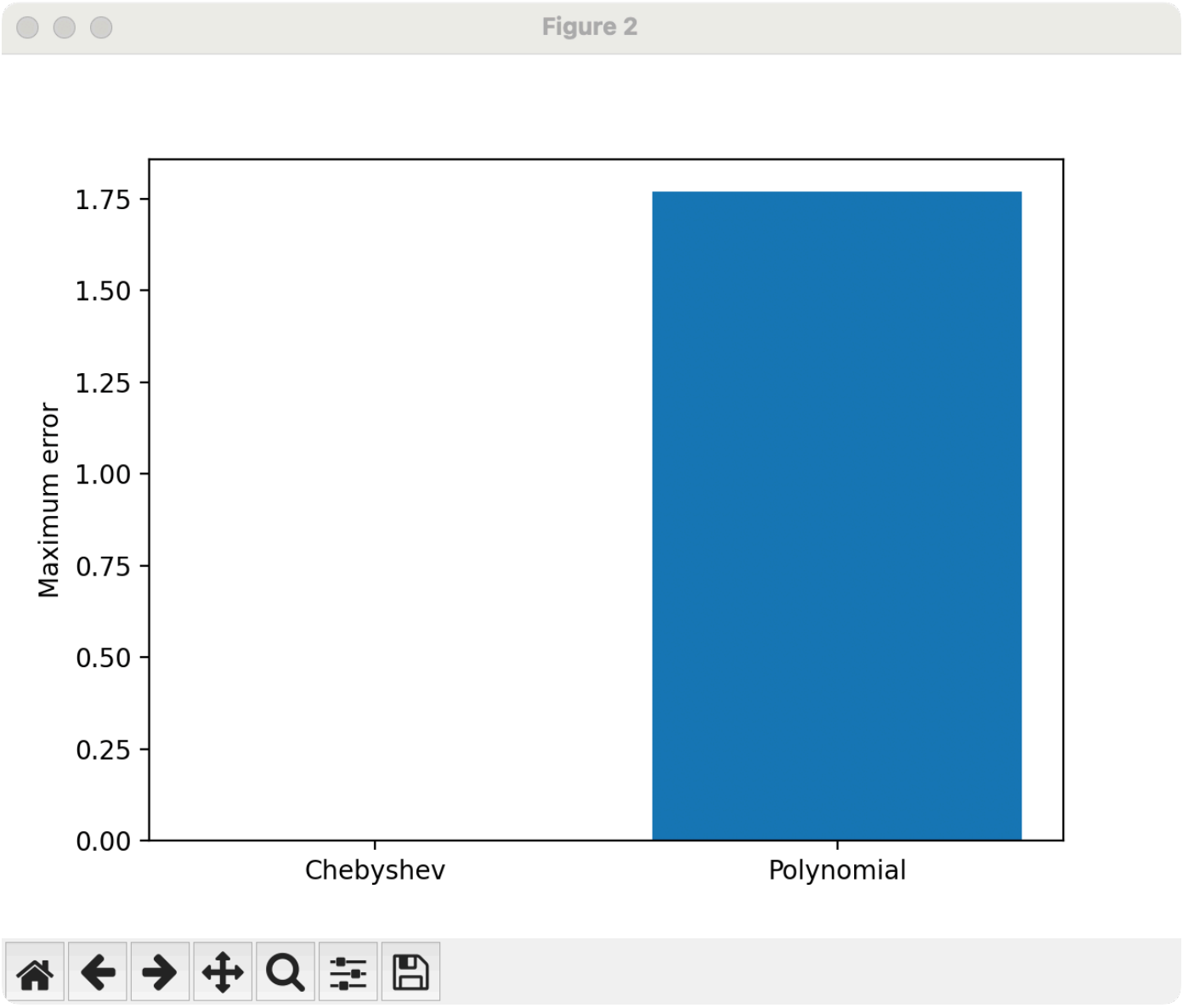


Figure 1
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
g(x)
Chebyshev fit
Polynomial fit



8、试用bincount()函数替代histogram()函数完成统计男青少年年龄和身高的例子的计算（数据见height.csv）

```
import csv

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# 存储年龄和身高数据
age_data = []
height_data = []

# 用csv库打开csv文件并读取数据
# with open('height.csv', 'r') as file:
#     reader = csv.reader(file)
#     for row in reader:
```

```
#         age_data.append(float(row[0]))
#         height_data.append(float(row[1]))

# 用pandas库读取文件
df = pd.read_csv('height.csv', usecols=[0, 1])
age_data = df.iloc[:, 0].values
height_data = df.iloc[:, 1].values

# 用hist统计男青少年年龄
# age_bins = [i * 2 for i in range(11)]
# plt.hist(age_data, bins=age_bins, color='blue')  # 用hist函数实现直方图统计
#
# 用np.bincount计算
age_bins = [i * 2 + 1 for i in range(10)]
age_hist = np.bincount(np.array(age_data).astype(int))
age_hist = np.cumsum(age_hist.reshape(-1, 2), axis=1)[:, -1]
plt.bar(age_bins, age_hist, width=2, color='blue')

plt.title('Age distribution of male teenagers')
plt.xlabel('Age')
plt.ylabel('Frequency')

plt.figure(2)
# 用hist统计男青少年身高
# height_bins = [i * 10 for i in range(10, 19)]
# plt.hist(height_data, bins=height_bins, color='green')


# # 用np.bincount计算
height_bins = [i * 10+5 for i in range(10, 18)]
height_hist = np.bincount(np.array(height_data).astype(int))
# 数据的拆解变形，计算每个统计点对应的数据
height_hist1 = height_hist[:int(height_hist.size / 10) *
10].reshape(int(height_hist.size / 10), 10)
height_hist2 = height_hist[int(height_hist.size / 10) * 10:]
height_hist2 = height_hist[int(height_hist.size / 10) * 10:]
height_hist2 = np.concatenate([height_hist2,np.zeros(10-len(height_hist2))])
height_hist2 = height_hist2.reshape(1, 10)
#
height_hist = np.concatenate([height_hist1,height_hist2],axis=0)
height_hist = np.cumsum(height_hist.reshape(-1, 10), axis=1)[:, -1]
plt.bar(height_bins, height_hist[10:], width=10, color='green')

plt.title('Height distribution of male teenagers')
plt.xlabel('Height')
plt.ylabel('Frequency')
plt.show()
```

运行结果

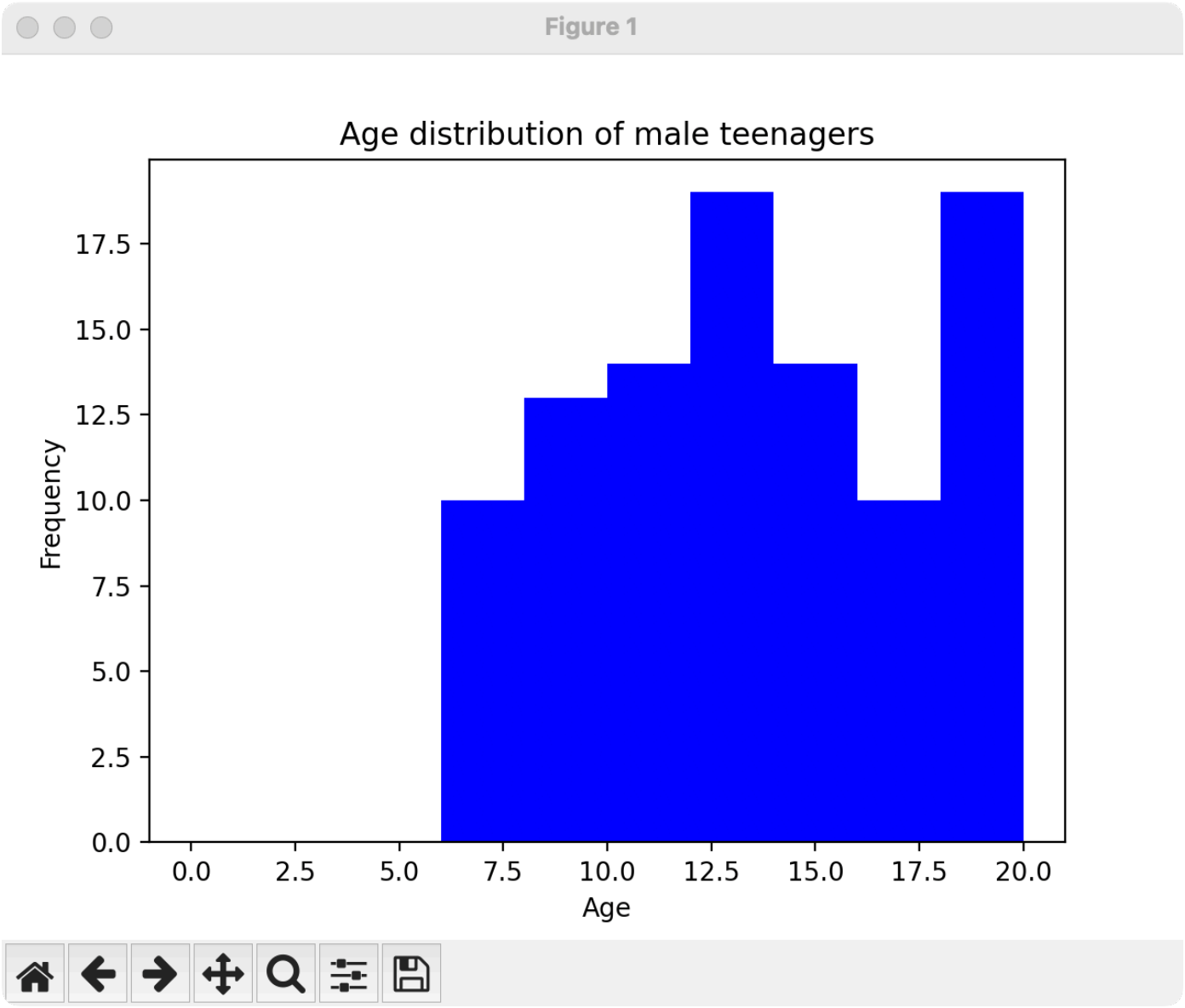
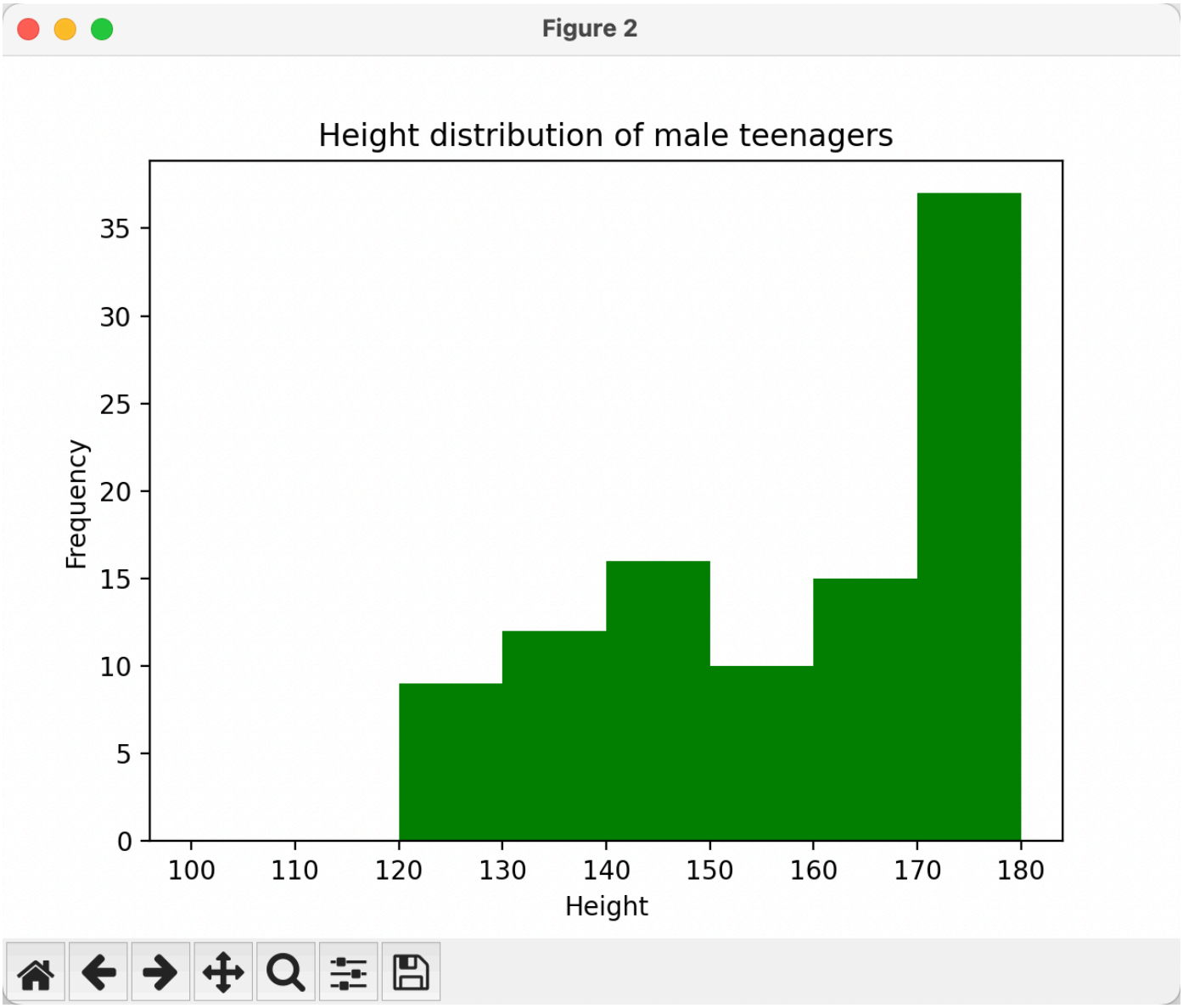


Figure 1
Age distribution of male teenagers
Frequency
Age
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0



9、使用二项分布进行赌博计算. 同时抛5枚硬币，如果正面朝上少于3枚，则输掉8元，否则就赢8元。如果手中有1000元作为赌资，请问赌博10000次后可能会是什么情况呢? (参见课件)

```
import numpy as np
from matplotlib.pyplot import plot, show

cash = np.zeros(1000)
cash[0] = 1000
outcome = np.random.binomial(5, 0.5, size=len(cash))

for i in range(1, len(cash)):
    if outcome[i] < 3 and cash[i - 1] != 0:
        cash[i] = cash[i - 1] - 8
    elif outcome[i] < 6 and cash[i - 1] != 0:
        cash[i] = cash[i - 1] + 8
    elif cash[i-1] == 0:
```

```
            print(u"输光钱钱了")
            break
        else:
            raise AssertionError("Unexpected outcome " + outcome)
    print(outcome.min(), outcome.max())

    # 使用Matplotlib绘制cash数组:
    plot(np.arange(len(cash)), cash)
    show()
```

运行结果: