Numerical Python

optimization

CS101 Lecture #18

Randomness Refresher

Randomness Refresher 1/20

Randomness refresher

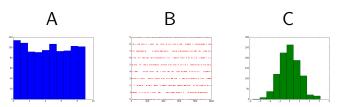
- randint(start,end,size=tuple)
- uniform(start,end,size=tuple)
- randn(d1,d2,...) #specify size of output
- Note that the interfaces for each are slightly different.

Randomness Refresher 2/20

Question #1

```
x = np.random.randint( 0,10, size=(1000,1) )
plt.hist( x )
plt.show()
```

What is a possible output of this code?

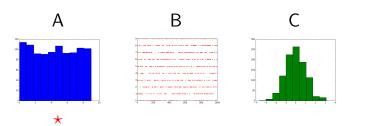


Randomness Refresher 3/20

Question #1

```
x = np.random.randint( 0,10, size=(1000,1) )
plt.hist( x )
plt.show()
```

What is a possible output of this code?

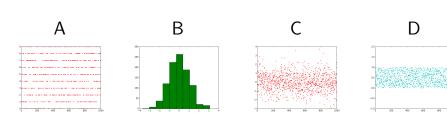


Randomness Refresher 4/20

Question #2

```
x = np.random.uniform( size=(1000,1) )
plt.plot( x, 'c.' )
plt.ylim( (-1,2) )
plt.show()
```

What is a possible output of this code?

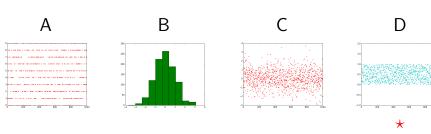


Randomness Refresher 5/20

Question #2

```
x = np.random.uniform( size=(1000,1) )
plt.plot( x, 'c.' )
plt.ylim( (-1,2) )
plt.show()
```

What is a possible output of this code?



http://matplotlib.org/api/colors_api.html

Randomness Refresher 6/20

Optimization 7/20

On vacation, you purchase a range of *n* souvenirs of varying weight and value. When it comes time to pack, you find that your bag has a weight limit of 50 pounds. What is the best set of items to take on the flight?

Optimization 8/20

- Given a function f(x), find x such that f(x) is maximized (or minimized).
- The goal is to search the domain for the optimal x yielding the optimal f(x).
- Many clever techniques exist, but we'll start with a naïve approach.

Optimization 9/20

Setup

```
import numpy as np

n = 10
items = list(range(n))
weights = np.random.uniform( size=(n,1) )*50
values = np.random.uniform( size=(n,1) )*100
```

Optimization 10/20

Setup

```
def f( wts, vals ):
    total_weight = 0
    total_value = 0
    for i in range( len( wts ) ):
        total_weight += wts[ i ]
        total value += vals[ i ]
    if total_weight >= 50:
        return 0
    else:
        return total value
```

Optimization 11/20

TART

n d mith

na wiin above th er of you feet. Shoulders i front of bar

Arms vertical to floor outside of knees LIFT

Drive heels into floor and push up with legs. Head inline with

spine, chin up, looking

straight ahead.

int your chest out on't squeeze your shoulder-blades.

Given a function f(x), find x such that f(x) is maximized (or executive minimized).

Brute-force searches the *entire* domain of *f*.

Hands about tension.

How could we do this in our case?

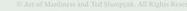
Do not lear backward o bend forwar Keep bar close body—roll it ov your knees and th until hips and kn

LOWER

Push hips back first, and then bend your mees once bar reaches mee level, keeping bar close to body.

> OR: Drop.

eet slightly more than h dth apart, pointed strai, head or slightly outwar



- Two useful functions from itertools to keep in mind:
 - combinations: provide all subsets of size n.
 - product: replace nested for loops.

Optimization 13/20

combinations: provide all subsets of size n.
import itertools

a = [1,2,3,4]
for x in itertools.combinations(a,2):
 print(x)

Optimization 14/20

```
product: replace nested for loops.
Can use repeat=n argument as well.
import itertools
a = [1,2,3,4]

b = ['g','h','i']
for x in itertools.product( a,b ):
    print(x)
for x in itertools.product( a, repeat=3 ):
    print(x)
```

Optimization 15/20

Question #3

```
x = 'ABCD'
z = 'XYZ'
for a in itertools.product( x,y ):
    print( ' '.join( a ) )
Which of the following is not printed?
 A'AX'
 B 'B D'
 C 'C X'
 D'DZ'
```

Optimization 16/20

Question #4

```
x = 'ABCD'
z = 'XYZ'
for a in itertools.product( x,y ):
    print( ' '.join( a ) )
Which of the following is not printed?
 A'AX'
 B 'B D' ★
 C'CX'
 D'DZ'
```

Optimization 17/20

Setup

```
import itertools
max value = 0.0
max_set = None
for i in range(1,n+1):
    for set in itertools.combinations(items,i):
        wts = \Pi
        vals = []
        for item in set:
            wts.append( weights[ item ] )
            vals.append( values[ item ] )
        value = f( wts, vals )
        if value > max_value:
            max_value = value
            max_set = set
```

Optimization 18/20

Brute-force search of a password:

```
def check_password( pwd ):
    if pwd == 'pas':
        return True
    else:
        return False
chars = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'+\
        'abcdefghijklmnopgrstuvwxyz0123456789'
for pair in itertools.product( chars, repeat=3 ):
    pair = ''.join( pair )
    if check password( pair ):
        print( pair )
```

Optimization 19/20

Brute-force search of a password:

$$2 \times n(\text{alphabet}) + n(\text{digits}) + n(\text{special})$$

= $2 \times 26 + 10 + \{24-32\}$
= $\{86-94\}$

per letter! This gets very big very quickly!

Optimization 20/20