Quiz Section Week 3
April 11, 2017
Today’s goals

• Stats review: p-values, null distributions, and significance testing

• Command line tips and tricks

• Python: Precedence, For loops and functions
P-values!

• P-values tell you about expectations under the null hypothesis
  • they say nothing about the alternative hypothesis or how probable it is

• Null hypothesis: usually the boring default, devil’s advocate position – what you want to see if you can disprove
P-values!

• P-values tell you about expectations under the null hypothesis
  • they say nothing about the alternative hypothesis or how probable it is

• Null hypothesis: usually the boring default, devil’s advocate position – what you want to see if you can disprove

There is no difference between treatment groups
Life expectancy is not changing over time
This coin is not weighted
These two sequences are unrelated
Historic example: R.A. Fisher and the tea-tasting test

8 cups of tea, randomly chosen to either have tea poured over milk or milk poured over tea

Null hypothesis? How many would she guess correctly if she were picking randomly?
Null distribution: What we suppose the data might look like if the null hypothesis is true

• This could be based on a parameterized probability distribution
  • E.g. Poisson: number of successes in x tries with y% probability of success

• Or you can generate an *empirical* null based on your real data
  • E.g. Shuffle the labels of the variable you want to test

• Defining the most appropriate null distribution is a relevant and tough problem in a lot of computational biology research!
Multiple testing can be dangerous!


- Nutrition & lifestyle questionnaires from 54 individuals

Our shocking new study finds that ...

<table>
<thead>
<tr>
<th>EATING OR DRINKING</th>
<th>IS LINKED TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw tomatoes</td>
<td>Judaism</td>
</tr>
<tr>
<td>Egg rolls</td>
<td>Dog ownership</td>
</tr>
<tr>
<td>Energy drinks</td>
<td>Smoking</td>
</tr>
<tr>
<td>Potato chips</td>
<td>Higher score on SAT math vs. verbal</td>
</tr>
<tr>
<td>Soda</td>
<td>Weird rash in the past year</td>
</tr>
<tr>
<td>Shellfish</td>
<td>Right-handedness</td>
</tr>
<tr>
<td>Lemonade</td>
<td>Belief that “Crash” deserved to win best picture</td>
</tr>
<tr>
<td>Fried/breaded fish</td>
<td>Democratic Party affiliation</td>
</tr>
<tr>
<td>Beer</td>
<td>Frequent smoking</td>
</tr>
<tr>
<td>Coffee</td>
<td>Cat ownership</td>
</tr>
<tr>
<td>Table salt</td>
<td>Positive relationship with Internet service provider</td>
</tr>
<tr>
<td>Steak with fat trimmed</td>
<td>Lack of belief in a god</td>
</tr>
<tr>
<td>Iced tea</td>
<td>Belief that “Crash” didn’t deserve to win best picture</td>
</tr>
<tr>
<td>Bananas</td>
<td>Higher score on SAT verbal vs. math</td>
</tr>
<tr>
<td>Cabbage</td>
<td>Innie bellybutton</td>
</tr>
</tbody>
</table>

Source: FFQ & Fivethirtyeight Supplement
Multiple testing can be dangerous!


- Nutrition & lifestyle questionnaires from 54 individuals

<table>
<thead>
<tr>
<th>EATING OR DRINKING</th>
<th>IS LINKED TO</th>
<th>P-VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw tomatoes</td>
<td>Judaism</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>Egg rolls</td>
<td>Dog ownership</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>Energy drinks</td>
<td>Smoking</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>Potato chips</td>
<td>Higher score on SAT math vs. verbal</td>
<td>0.0001</td>
</tr>
<tr>
<td>Soda</td>
<td>Weird rash in the past year</td>
<td>0.0002</td>
</tr>
<tr>
<td>Shellfish</td>
<td>Right-handedness</td>
<td>0.0002</td>
</tr>
<tr>
<td>Lemonade</td>
<td>Belief that “Crash” deserved to win best picture</td>
<td>0.0004</td>
</tr>
<tr>
<td>Fried/breaded fish</td>
<td>Democratic Party affiliation</td>
<td>0.0007</td>
</tr>
<tr>
<td>Beer</td>
<td>Frequent smoking</td>
<td>0.0013</td>
</tr>
<tr>
<td>Coffee</td>
<td>Cat ownership</td>
<td>0.0018</td>
</tr>
<tr>
<td>Table salt</td>
<td>Positive relationship with Internet service provider</td>
<td>0.0014</td>
</tr>
<tr>
<td>Steak with fat trimmed</td>
<td>Lack of belief in a god</td>
<td>0.0030</td>
</tr>
<tr>
<td>Iced tea</td>
<td>Belief that “Crash” didn’t deserve to win best picture</td>
<td>0.0043</td>
</tr>
<tr>
<td>Bananas</td>
<td>Higher score on SAT verbal vs. math</td>
<td>0.0073</td>
</tr>
<tr>
<td>Cabbage</td>
<td>Innie bellybutton</td>
<td>0.0097</td>
</tr>
</tbody>
</table>

SOURCE: FFQ & FIVETHIRTYEIGHT SUPPLEMENT
Multiple testing can be dangerous!


- Nutrition & lifestyle questionnaires from 54 individuals

This is exactly the same as testing for alignments between thousands of sequences

<table>
<thead>
<tr>
<th>EATING OR DRINKING</th>
<th>IS LINKED TO</th>
<th>P-VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw tomatoes</td>
<td>Judaism</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>Egg rolls</td>
<td>Dog ownership</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>Energy drinks</td>
<td>Smoking</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>Potato chips</td>
<td>Higher score on SAT math vs. verbal</td>
<td>0.0001</td>
</tr>
<tr>
<td>Soda</td>
<td>Weird rash in the past year</td>
<td>0.0002</td>
</tr>
<tr>
<td>Shellfish</td>
<td>Right-handedness</td>
<td>0.0002</td>
</tr>
<tr>
<td>Lemonade</td>
<td>Belief that “Crash” deserved to win best picture</td>
<td>0.0004</td>
</tr>
<tr>
<td>Fried/breaded fish</td>
<td>Democratic Party affiliation</td>
<td>0.0007</td>
</tr>
<tr>
<td>Beer</td>
<td>Frequent smoking</td>
<td>0.0013</td>
</tr>
<tr>
<td>Coffee</td>
<td>Cat ownership</td>
<td>0.0018</td>
</tr>
<tr>
<td>Table salt</td>
<td>Positive relationship with Internet service provider</td>
<td>0.0014</td>
</tr>
<tr>
<td>Steak with fat trimmed</td>
<td>Lack of belief in a god</td>
<td>0.0030</td>
</tr>
<tr>
<td>Iced tea</td>
<td>Belief that “Crash” didn’t deserve to win best picture</td>
<td>0.0043</td>
</tr>
<tr>
<td>Bananas</td>
<td>Higher score on SAT verbal vs. math</td>
<td>0.0073</td>
</tr>
<tr>
<td>Cabbage</td>
<td>Innie bellybutton</td>
<td>0.0097</td>
</tr>
</tbody>
</table>

SOURCE: FFQ & FIVETHIRTYEIGHT SUPPLEMENT
Bonferroni correction: just raise the threshold depending on the total # of tests

- For 1000 tests: Use a threshold 1000x stricter
  - Does not require tests to have a particular relationship with each other
  - Ensures that the probability of rejecting a true null hypothesis is still less than your original desired p-value threshold

- Suppose they did 1000 tests for this study
- (50 lifestyle qs and 200 foods)
  - What’s the corresponding E-value for p=0.0001?

Our shocking new study finds that...

<table>
<thead>
<tr>
<th>EATING OR DRINKING</th>
<th>IS LINKED TO</th>
<th>P-VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw tomatoes</td>
<td>Judaism</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>Egg rolls</td>
<td>Dog ownership</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>Energy drinks</td>
<td>Smoking</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>Potato chips</td>
<td>Higher score on SAT math vs. verbal</td>
<td>0.0001</td>
</tr>
<tr>
<td>Soda</td>
<td>Weird rash in the past year</td>
<td>0.0002</td>
</tr>
<tr>
<td>Shellfish</td>
<td>Right-handedness</td>
<td>0.0002</td>
</tr>
<tr>
<td>Lemonade</td>
<td>Belief that “Crash” deserved to win best picture</td>
<td>0.0004</td>
</tr>
<tr>
<td>Fried/breaded fish</td>
<td>Democratic Party affiliation</td>
<td>0.0007</td>
</tr>
<tr>
<td>Beer</td>
<td>Frequent smoking</td>
<td>0.0013</td>
</tr>
</tbody>
</table>

FYI: Sometimes this is too harsh, and false discovery rate corrections can be more useful
Programming

Note: If you are new to coding and still spending lots of time on lots of small errors, this is 100% normal!! Keep at it and it will get easier.
Useful command line tools

- `ls []` = list files
  - `ls -lh` = list files with extra detail
- `cd []` = change directory
  - `cd ..` = change to the parent directory
- `mkdir []` = make a new directory
- `rm []` = delete files
- `mv []` = move a file to a different name or location
- `cp []` = copy a file
- `pwd []` = print the current directory
- `head []` = print first lines of a file, `tail []` = print last lines of file
- `cat []` = print contents of a file
- `less []` = print contents of a file
- `grep [] []` = search for a string in a file

A pretty good starter reference:

- up arrow = previous command history
- tab = autocomplete file name
- | pipe = use output as input to next command

- You can install Python modules from the command line using `pip`
- Much much more...
Practice:

- Navigate to wherever you saved your homework 1 python script
- take a look at the first lines of it using “head”
- Use grep to search for the “%” operator
- then rename the file using “mv”
Order of operations/precedence

• Same rules apply as for mathematical expressions generally, plus some other conventions

• Without other specifications, evaluation happens left-to-right

What’s the value of each expression?

2 ** 7 == 127 + 1
2 + 2 ** 7 == 128 + 1
2 + (2 ** 7 == 128)+1
2 == 2 or 3 == 2 == False
2 == 2 or 3 in [3,4] == False
(2 == 2 or 3 in [3,4]) == False

Reference: https://docs.python.org/2/reference/expressions.html#operator-precedence
Order of operations/precedence

• Same rules apply as for mathematical expressions generally, plus some other conventions

• Without other specifications, evaluation happens left-to-right

What’s the value of each expression?

2 ** 7 == 127 + 1
2 + 2 ** 7 == 100
(2 + 2 ** 7 == 100)+1
2 == 2 or 3 == 2 == False
2 == 2 or 3 in [3,4] == False
(2 == 2 or 3 in [3,4]) == False

When in doubt: Use parentheses!!!

Reference: https://docs.python.org/2/reference/expressions.html#operator-precedence
For loops let you repeatedly apply the same lines of code to each element in a list

```python
x = [1, 2, 3]
for i in x:
    print i
print 'done!
```

i takes on the value of each element in the list for each iteration of the code inside the for loop block
For loops let you repeatedly apply the same lines of code to each element in a list

```python
x = [1, 2, 3]
for i in x:
    print i
print 'done!'
```

i takes on the value of each element in the list for each iteration of the code inside the for loop block
For loops also work for strings!

```python
x = 'actg'
for i in x:
    print i
done!
```
Compute the sum of the numbers in list x!

```python
x = [1, 2, 4, 5]

sum = 0
for v in x:
    sum = sum + v
print 'The sum is:', sum

The sum is: 12```
How about the product?

\[ x = [1, 2, 4, 5] \]

```
print 'The product is:', product
```

The product is: 40
How about the product?

\[ x = [1, 2, 4, 5] \]

```python
product = 1
for v in x:
    product = product * v
print 'The product is:', product
```

The product is: 40
Powerful strategy: Combining for loops and if/else statements

```
x = [12, 3, 4.4, 6]

Output how many numbers in the list x with values greater than 5

    count = 0
    for v in x:
        if v > 5:
            count = count + 1
    print count
```

*Watch the indents!!*
Example: reverse complement

s = 'ATCG'

reverse_complement = ''
for nuc in s:
    # Find the complement of the nucleotide
    reverse_complement = nuc + reverse_complement

print reverse_complement

'CGAT'
Example: reverse complement

`s = 'ATCG'

reverse_complement = ''
for nuc in s:
    # Find the complement of the nucleotide
    if nuc == 'A':
        nuc = 'T'
    elif nuc == 'T':
        nuc = 'A'
    elif nuc == 'C':
        nuc = 'G'
    elif nuc == 'G':
        nuc = 'C'
    # Add the complement to the beginning of new string
    reverse_complement = nuc + reverse_complement

print reverse_complement
    'CGAT'`
Functions are sub-programs that you can call in one line

```python
>>> x = [1, 2, 3]
>>> print len(x)
3
```

- Used as a single word (no spaces) followed by "()", where the input to the function goes within the parentheses
- The function will run, and it will evaluate to the output of the function

```python
>>> print len( ‘hello!’ )
```

```
Functions are sub-programs that you can call in one line

- Python has many built-in functions (e.g. len)
- Modules contain definitions for additional functions
- Next week we will talk about defining and writing your own functions
Useful list functions

# Initializing a sequence of integers
x = range(0,4)
print x
[0, 1, 2, 3]

# Adding to the end of a list
x.append('four')
print x
[0, 1, 2, 3, four]
Iterate through a list with range() and indices

```python
english = ['zero', 'one', 'two']
spanish = ['cero', 'uno', 'dos']

for i in range(0, len(english)):
    print english[i], spanish[i]
```
Iterate through a list with range() and indices

```python
english = ['zero', 'one', 'two']
spanish = ['cero', 'uno', 'dos']

for i in range(0, len(english)):
    print(english[i], spanish[i])

print('zero cero')
print('one uno')
print('two dos')```
Useful string functions

These are defined to manipulate a specific string variable, so we use the . to reflect that (more on this later)

```python
>>> s = "GATTACA"
>>> s.find("ATT")
1
>>> s.count("T")
2
>>> s.lower()
'gattaca'
>>> s+s
'GATTACAGATTACA'
>>> s.upper()
'GATTACA'
>>> s.replace("G", "U")
'UATTACA'
>>> s.replace("C", "U")
'GATTAUA'
>>> s.replace("AT", "**")
'G**TACA'
```
Extra practice:

https://interactivepython.org/runestone/static/thinkcspy/index.html
Sections 4 and 6