```
  Course: Statistical_Inference
  Lesson: Multiple_Testing


- Class: text
  Output: "Multiple_Testing. (Slides for this and other Data Science courses may be
  found at github https://github.com/DataScienceSpecialization/courses/. If you care to
  use them, they must be downloaded as a zip file and viewed locally. This lesson
  corresponds to 06_Statistical_Inference/12_MultipleTesting.)"

- Class: text
  Output: In this lesson, we'll discuss multiple testing. You might ask, "What's that?"

- Class: text
  Output: Given that data is valuable and we'd like to get the most out of it, we might
  use it to test several hypotheses. If we have an alpha level of .05 and we test 20
  hypotheses, then on average, we expect one error, just by chance.

- Class: text
  Output: Another potential problem is that after running several tests,  only the
  lowest p-value might be reported OR all p-values under some threshold might be
  considered significant. Undoubtedly, some of these would be false.

- Class: text
  Output: Luckily, we have clever ways of minimizing errors in this situation. That's
  what we'll address.    We'll define specific error measures and then statistical ways
  of correcting or limiting them.


- Class: text
  Output: Multiple testing is particularly relevant now in this age of BIG data.
  Statisticians are tasked with questions such as "Which variables matter among the
  thousands measured?" and "How do you relate unrelated information?"

- Class: mult_question
  Output: Since multiple testing addresses compensating for errors let's review what we
  know about them. A Type I error is
  AnswerChoices: rejecting a false hypothesis; failing to reject a false hypothesis;
  rejecting a true hypothesis; failing to reject a true hypothesis
  CorrectAnswer: rejecting a true hypothesis
  AnswerTests: omnitest(correctVal='rejecting a true hypothesis')
  Hint: Eliminate the two choices that are not errors. A Type I error involves rejection.

- Class: mult_question
  Output: In an American court, an example of a Type I error is
  AnswerChoices: convicting an innocent person; acquitting a guilty person; letting the
  indicted off on a technicality
  CorrectAnswer: convicting an innocent person
  AnswerTests: omnitest(correctVal='convicting an innocent person')
  Hint: In an American court, the null hypothesis is that the accused is innocent. If
  he is convicted when he really is innocent then the null hypothesis is rejected
  incorrectly.


- Class: mult_question
  Output:  A Type II error is
  AnswerChoices: rejecting a false hypothesis; failing to reject a false hypothesis;
  rejecting a true hypothesis; failing to reject a true hypothesis
  CorrectAnswer: failing to reject a false hypothesis
  AnswerTests: omnitest(correctVal='failing to reject a false hypothesis')
  Hint: Eliminate the two choices that are not errors. A Type II error involves failing
  to  reject.

- Class: mult_question
  Output: In an American court, an example of a Type II error is
  AnswerChoices: convicting an innocent person; acquitting a guilty person; letting the
  indicted off on a technicality
  CorrectAnswer: acquitting a guilty person
  AnswerTests: omnitest(correctVal='acquitting a guilty person')
```

```
  Hint: In an American court, the null hypothesis is that the accused is innocent. If
  this is accepted (not rejected) by the jury and the defendant is in fact guilty a
  Type II error has been made.

- Class: mult_question
  Output: Good. Let's continue reviewing. The null hypothesis
  AnswerChoices: represents the status_quo and is assumed true; tells us the origins of
  the number 0; is never true;  is a big nothing that statisticians like to gossip about
  CorrectAnswer: represents the status_quo and is assumed true
  AnswerTests: omnitest(correctVal='represents the status_quo and is assumed true')
  Hint: Really? Only one choice seems reasonable.

- Class: text
  Output: The p-value is "the probability under the null hypothesis of obtaining
  evidence as or more extreme than  your test statistic (obtained from your observed
  data) in the direction of the alternative hypothesis." Of course p-values are related
  to significance or alpha levels, which are set before the test is conducted (often at
  0.05).

- Class: mult_question
  Output: If a p-value is found to be less than alpha (say 0.05), then the test result
  is considered statistically significant, i.e., surprising and unusual, and the null
  hypothesis (the status quo) is ?
  AnswerChoices: accepted; rejected; revised; renamed the aleph null hypothesis
  CorrectAnswer: rejected
  AnswerTests: omnitest(correctVal='rejected')
  Hint: Accepted (failed to be rejected) or rejected are the only real choices here. A
  low p-value is a low probability. This means your data is unusual and is closer to
  the alternative hypothesis than the null.

- Class: figure
  Output:  Now consider this chart copied from
  http://en.wikipedia.org/wiki/Familywise_error_rate. Suppose we've tested m null
  hypotheses, m_0 of which are actually true, and  m-m_0  are actually false. Out of
  the m tests R have been declared significant, that is, the associated p-values were
  less than alpha, and m-R were nonsignificant, or boring results.
  Figure: errorMat.R
  FigureType: new

- Class: mult_question
  Output: Looking at the chart, which variables are known?
  AnswerChoices: m and R; m_0, and m; S,T,U,V; A,B,C
  CorrectAnswer:  m and R
  AnswerTests: omnitest(correctVal='m and R')
  Hint: The number of hypotheses tested (m) and the number declared significant (R) are
  known. The variable m_0 represents the unknowable, the number of true hypotheses. S,
  T, U, and V are unobservable random variables.

- Class: mult_question
  Output: In testing the m_0 true null hypotheses, V results were declared significant,
  that is, these tests favored the alternative hypothesis. What type of error does this
  represent?
  AnswerChoices: Type I; Type II; Type III; a serious one
  CorrectAnswer: Type I
  AnswerTests: omnitest(correctVal='Type I')
  Hint: By declaring the test result significant the true null hypothesis was rejected,
  like convicting an innocent person.

- Class: text
  Output:  Another name for a Type I error is False Positive, since it is falsely
  claiming a significant (positive) result.

- Class: mult_question
  Output: Of the m-m_0 false null hypotheses, T were declared nonsignificant. This
  means that these T null hypotheses were accepted (failed to be rejected). What type
  of error does this represent?
  AnswerChoices: Type I; Type II; Type III; a serious one
  CorrectAnswer: Type II
  AnswerTests: omnitest(correctVal='Type II')
```

```
  Hint: By declaring the test result nonsignificant the false null hypothesis was
  accepted (failed to be rejected), like letting a guilty person go free.

- Class: text
  Output:  Another name for a Type II error is False Negative, since it is falsely
  claiming a nonsignificant (negative) result.

- Class: text
  Output: A rose by any other name, right? Consider the fraction V/R.

- Class: text
  Output:  The observed R represents the number of test results declared significant.
  These are 'discoveries', something different from the status quo. V is the number of
  those falsely declared significant, so V/R is the ratio of FALSE discoveries. Since V
  is a random variable (i.e., unknown until we do an experiment) we call the expected
  value of the ratio, E(V/R), the False Discovery Rate (FDR).

- Class: text
  Output: A rose by any other name, right? How about the fraction V/m_0? From the
  chart, m_0 represents the number of true H_0's and m_0 is unknown. V is the number of
  those falsely declared significant, so V/m_0 is the ratio of FALSE positives. Since V
  is a random variable (i.e., unknown until we do an experiment) we call the expected
  value of the ratio, E(V/m_0), the FALSE POSITIVE rate.

- Class: mult_question
  Output: Another good name for the false positive rate would be
  AnswerChoices: false alarm rate; the Type II rate; a rose; a thorn
  CorrectAnswer: false alarm rate
  AnswerTests: omnitest(correctVal='false alarm rate')
  Hint: False positives are Type I errors so one of the only two sensible answers is
  incorrect.

- Class: mult_question
  Output: The false positive rate would be closely related to
  AnswerChoices: the Type I error rate; the Type II error rate; a thorny rose;
  CorrectAnswer: the Type I error rate
  AnswerTests: omnitest(correctVal='the Type I error rate')
  Hint: False positives are Type I errors so one of the only two sensible answers is
  incorrect.

- Class: text
  Output: We call the probability of at least one false positive, Pr(V >= 1) the Family
  Wise Error Rate (FWER).

- Class: text
  Output: So how do we control the False Positive Rate?

- Class: text
  Output: Suppose we're really smart, calculate our p-values correctly, and declare
  all tests with p < alpha as significant. This means that our false positive rate is
  at most alpha, on average.

- Class: mult_question
  Output: Suppose we  perform 10,000 tests and alpha = .05. How many false positives do
  we expect on average?
  AnswerChoices: 500; 5000; 50; 50000
  CorrectAnswer: 500
  AnswerTests: omnitest(correctVal='500')
  Hint: Multiply 10000 by .05 to get the correct answer.

- Class: text
  Output: You got it! 500 false positives seems like a lot. How do we avoid so many?

- Class: text
  Output:  We can try to control the family-wise error rate (FWER), the probability of
  at least one false positive, with the Bonferroni correction, the oldest multiple
  testing correction.

- Class: text
```

```
    Output: It's very straightforward. We do m tests and want to control the FWER at
    level alpha so that Pr(V >= 1) < alpha. We simply reduce alpha dramatically. Set
    alpha_fwer to be alpha/m. We'll only call a test result significant if its p-value <
    alpha_fwer.

  - Class: mult_question
    Output: Sounds good, right? Easy to calculate. What would be a drawback with this
    method?
    AnswerChoices: too many results will pass; too many results will fail; requires too
    much math
    CorrectAnswer: too many results will fail
    AnswerTests: omnitest(correctVal='too many results will fail')
    Hint: Dividing alpha by m makes your cutoff value very small so you might not get any
    significant results, much less false ones.

  - Class: text
    Output: Another way to limit the false positive rate is to control the false
    discovery rate (FDR). Recall this is E(V/R). This is the most popular correction when
    performing lots of tests. It's used in lots of areas such as genomics, imaging,
    astronomy, and other signal-processing disciplines.

  - Class: text
    Output:  Again, we'll do m tests but now we'll set the FDR, or E(V/R) at level alpha.
    We'll calculate the p-values as usual and order them from smallest to largest,  p_1,
    p_2,...p_m. We'll call significant any result with p_i <= (alpha*i)/m. This is the
    Benjamini-Hochberg method (BH). A p-value is compared to a value that depends on its
    ranking.

  - Class: text
    Output: This is equivalent to finding the largest k such that p_k <= (k * alpha)/m,
    (for a given alpha) and then rejecting all the null hypotheses for i=1,...,k.



  - Class: text
    Output: Like the Bonferroni correction, this is easy to calculate and it's much less
    conservative. It might let more false positives through and it may behave strangely
    if the tests aren't independent.

  - Class: figure
    Output:  Now consider this chart copied from the slides. It shows the p-values for 10
    tests performed at the alpha=.2 level and three cutoff lines. The p-values are shown
    in order from left to right along the x-axis. The red line is the threshold for No
    Corrections (p-values are compared to alpha=.2), the blue line is the Bonferroni
    threshold, alpha=.2/10 = .02, and the gray line shows the BH correction. Note that it
    is not horizontal but has a positive slope as we expect.
    Figure: corrMat1.R
    FigureType: new

  - Class: mult_question
    Output: With no correction, how many results are declared significant?
    AnswerChoices: 2; 4; 6 ;8
    CorrectAnswer: 4
    AnswerTests: omnitest(correctVal='4')
    Hint: How many points fall below the red line?

  - Class: mult_question
    Output: With the Bonferroni correction, how many tests are declared significant?
    AnswerChoices: 2; 4; 6; 8
    CorrectAnswer: 2
    AnswerTests: omnitest(correctVal='2')
    Hint: How many points fall below the blue line?

  - Class: mult_question
    Output: So the Bonferroni passed only half the results that the No Correction
    (comparing p-values to alpha) method passed. Now look at the BH correction. How many
    tests are significant with this scale?
    AnswerChoices: 1; 3; 5; 7
    CorrectAnswer: 3
```

```
    AnswerTests: omnitest(correctVal='3')
    Hint: How many points fall below the gray line?.

  - Class: text
    Output: So the BH correction which limits the FWER is between the No Correction and
    the Bonferroni. It's more conservative (fewer significant results) than the No
    Correction but less conservative (more significant results) than the Bonferroni. Note
    that with this method the threshold is proportional to the ranking of the values so
    it slopes positively while the other two thresholds are flat.

  - Class: text
    Output: Notice how both the Bonferroni and BH methods adjusted the threshold (alpha)
    level of rejecting the null hypotheses. Another equivalent corrective approach is to
    adjust the p-values, so they're not classical p-values anymore, but they can be
    compared directly to the original alpha.

  - Class: text
    Output:  Suppose the p-values are p_1, ... , p_m.  With the Bonferroni method you
    would adjust these by setting p'_i = max(m  * p_i, 1)  for each p-value. Then if you
    call all p'_i < alpha significant you will control the FWER.


  - Class: figure
    Output: To demonstrate some of these concepts, we've created an array of p-values for
    you. It is 1000-long and the result of a linear regression performed on random normal
    x,y pairs so there is no true significant relationship between the x's and y's.
    Figure: genNoTrue.R
    FigureType: new

  - Class: cmd_question
    Output:  Use the R command head to see the first few entries of the array pValues.
    CorrectAnswer: head(pValues)
    AnswerTests: omnitest(correctExpr='head(pValues)')
    Hint: Type head(pValues) at the command prompt.

  - Class: cmd_question
    Output:  Now count the number of entries in the array that are less than the value
    .05. Use the R command sum, and the appropriate Boolean expression.
    CorrectAnswer: sum(pValues < 0.05)
    AnswerTests: omnitest(correctExpr='sum(pValues < 0.05)')
    Hint: Type sum(pValues < 0.05) at the command prompt.

  - Class: cmd_question
    Output:  "So we got around 50 false positives, just as we expected (.05*1000=50). The
    beauty of R is that it provides a lot of built-in statistical functionality. The
    function p.adjust is one example. The first argument is the array of pValues. Another
    argument is the method of adjustment. Once again, use the R function sum and a
    boolean expression using p.adjust with method=\"bonferroni\" to control the FWER."
    CorrectAnswer: sum(p.adjust(pValues,method="bonferroni") < 0.05)
    AnswerTests: omnitest(correctExpr='sum(p.adjust(pValues,method=\"bonferroni\") <
    0.05)')
    Hint: Type sum(p.adjust(pValues,method="bonferroni") < 0.05) at the command prompt.

  - Class: cmd_question
    Output: "So the correction eliminated all the false positives that had passed the
    uncorrected alpha test. Repeat the same experiment, this time using the method \"BH\"
    to control the FDR."
    CorrectAnswer: sum(p.adjust(pValues,method="BH") < 0.05)
    AnswerTests: omnitest(correctExpr='sum(p.adjust(pValues,method="BH") < 0.05)')
    Hint: Type sum(p.adjust(pValues,method="BH") < 0.05) at the command prompt.

  - Class: figure
    Output: So the BH method also eliminated all the false positives. Now we've generated
    another 1000-long array of p-values, this one called pValues2. In this data, the
    first half ( 500 x/y pairs) contains x and y values that are random and the second
    half contain x and y pairs that are related, so running a linear regression model on
    the 1000 pairs should find some significant  (not random) relationship.
    Figure: gen50True.R
    FigureType: new
```

```
240
241 - Class: cmd_question
242   Output:  We also created a 1000-long array of character strings, trueStatus. The
      first 500 entries are "zero" and the last are "not zero". Use the R function tail to
      look at the end of trueStatus.
243   CorrectAnswer: tail(trueStatus)
244   AnswerTests: omnitest(correctExpr='tail(trueStatus)')
245   Hint: Type tail(trueStatus) at the command prompt.
246
247 - Class: cmd_question
248   Output: Once again we can use R's greatness to count and tabulate for us. We can call
      the R function table with two arguments, a boolean such as pValues2<.05, and the
      array trueStatus. The boolean obviously has two outcomes and each entry of trueStatus
      has one of two possible values. The function table aligns the two arguments and
      counts how many of each combination (TRUE,"zero"), (TRUE,"not zero"), (FALSE,"zero"),
      and (FALSE,"not zero") appear. Try it now.
249   CorrectAnswer: table(pValues2 < 0.05, trueStatus)
250   AnswerTests: omnitest(correctExpr='table(pValues2 < 0.05, trueStatus)')
251   Hint: Type table(pValues2 < 0.05, trueStatus) at the command prompt.
252
253 - Class: text
254   Output: "We see that without any correction all 500 of the truly significant
      (nonrandom) tests were correctly identified in the \"not zero\" column. In the zero
      column (the truly random tests), however, 24 results were flagged as significant."
255
256 - Class: cmd_question
257   Output: What is the percentage of false positives in this test?
258   CorrectAnswer: 24/500
259   AnswerTests: equiv_val(.048)
260   Hint: "Divide 24 by 500 to get the percentage."
261
262 - Class: text
263   Output: "Just as we expected - around 5% or .05*100."
264
265 - Class: cmd_question
266   Output: "Now run the same table function, however, this time use the call to p.adjust
      with the \"bonferroni\" method in the boolean expression. This will control the FWER."
267   CorrectAnswer: "table(p.adjust(pValues2,method=\"bonferroni\") < 0.05, trueStatus)"
268   AnswerTests: omnitest(correctExpr='table(p.adjust(pValues2,method=\"bonferroni\") <
      0.05, trueStatus)')
269   Hint: "Type table(p.adjust(pValues2,method=\"bonferroni\") < 0.05, trueStatus) at the
      command prompt."
270
271 - Class: text
272   Output: Since the Bonferroni correction method is more conservative than just
      comparing p-values to alpha all the truly random tests are correctly identified in
      the zero column. In other words, we have no false positives. However, the threshold
      has been adjusted so much that 23 of the truly significant results have been
      misidentified in the not zero column.
273
274 - Class: cmd_question
275   Output: "Now run the same table function one final time. Use the call to p.adjust
      with \"BH\" method in the boolean expression. This will control the false discovery
      rate."
276   CorrectAnswer: "table(p.adjust(pValues2,method=\"BH\") < 0.05, trueStatus)"
277   AnswerTests: omnitest(correctExpr='table(p.adjust(pValues2,method=\"BH\") < 0.05,
      trueStatus)')
278   Hint: "Type table(p.adjust(pValues2,method=\"BH\") < 0.05, trueStatus) at the command
      prompt."
279
280 - Class: text
281   Output: "Again, the results are a compromise between the No Corrections and the
      Bonferroni. All the significant results were correctly identified in the \"not zero\"
      column but in the random (\"zero\") column 13 results were incorrectly identified.
      These are the false positives. This is roughly half the number of errors in the other
      two runs."
282
283 - Class: figure
284   Output: Here's a plot of the two sets of adjusted p-values, Bonferroni on the left
```

```
    and BH on the right. The x-axis indicates the original p-values. For the Bonferroni,
    (adjusting by multiplying by 1000, the number of tests), only a few of the adjusted
    values are below 1. For the BH, the adjusted values are slightly larger than the
    original values.
285   Figure: plot2.R
286   FigureType: new
287
288 - Class: text
289   Output: We'll conclude by saying that multiple testing is an entire subfield of
    statistical inference. Usually a basic Bonferroni/BH correction is good enough to
    eliminate false positives, but if there is strong dependence between tests there may
    be problems. Another correction method to consider is "BY".
290
291 - Class: text
292   Output: Congrats! We hope you liked the multiple concepts and questions you saw in
    this lesson.
293
294 - Class: mult_question
295   Output: "Would you like to receive credit for completing this course on
296     Coursera.org?"
297   CorrectAnswer: NULL
298   AnswerChoices: Yes;No
299   AnswerTests: coursera_on_demand()
300   Hint: ""
301
```