

Since September 2013 *Nature Methods* has been publishing a monthly column on statistics called "Points of Significance."



**Importance of being uncertain** - How samples are used to estimate population statistics and what this means in terms of uncertainty.



**Error Bars** - The use of error bars to represent uncertainty and advice on how to interpret them.



**Significance, P values and t-tests** - Introduction to the concept of statistical significance and the one-sample t-test.



**Power and sample size** - Use of statistical power to optimize study design and sample numbers.



**Visualizing samples with box plots** - Introduction to box plots and their use to illustrate the spread and differences of samples. See also: [Kick the bar chart habit](#) and [BoxPlotR](#), a web tool for generation of box plots



**Comparing samples—part I** - How to use the two-sample t-test to compare either uncorrelated or correlated samples.



**Comparing samples—part II** - Adjustment and reinterpretation of P values when large numbers of tests are performed.



**Nonparametric tests** - Use of nonparametric tests to robustly compare skewed or ranked data.



**Designing comparative experiments** - The first of a series of columns that tackle experimental design shows how a paired design achieves sensitivity and specificity requirements despite biological and technical variability.



**Analysis of variance and blocking** - Introduction to ANOVA and the importance of blocking in good experimental design to mitigate experimental error and the impact of factors not under study.



**Replication** - Technical replication reveals technical variation while biological replication is required for biological inference.



**Nested designs** - Use the relative noise contribution of each layer in nested experimental designs to optimally allocate experimental resources using ANOVA.



**Two-factor designs** - It is common in biological systems for multiple experimental factors to produce interacting effects on a system. A study design that allows these interactions can increase sensitivity.



**Sources of variation** - To generalize experimental conclusions to a population, it is critical to sample its variation while using experimental control, randomization, blocking and replication for replicable and meaningful results.

Source: <http://www.nature.com/collections/qghqmq/pointsofsignificance>

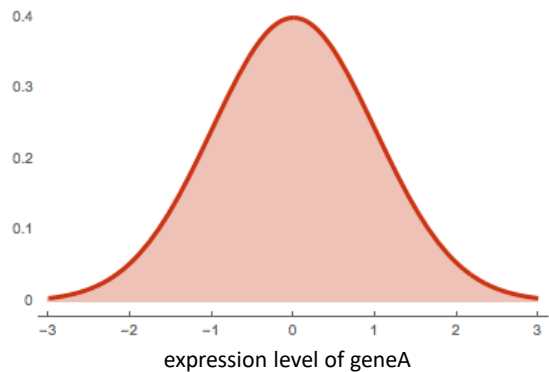
## The population

A **statistical population** is the set of **ALL** the **individuals** about which we want to make **inferences (or predictions)**. These inferences are made about one or more **variables** of an individual.

- population: "all the people in the world", "all men older than 55", "all genes in the human genome", ...
- individual: "one person", "a man older than 55", "a gene in the human genome", ...
- inference = deducing properties: "average age", "is the average blood pressure of men older than 55 significantly higher", "can the expression level of a human gene diagnose disease?"
- variable: "age", "blood pressure", "expression level"

## The probability density function (PDF)

- We can **describe** the values of a variable of a population by a **probability density function (PDF)**.
- A **probability density function (PDF)**, or density of a (continuous) variable, is a **function**, whose value at any given sample (or point) in the sample space (the set of possible values taken by the variable) can be interpreted as providing a relative likelihood that the value of the variable would equal that sample.
- The PDF of the expression levels for geneA has a very common symmetric (bell-like) shape and is known as a **normal (or Gaussian) distribution**.



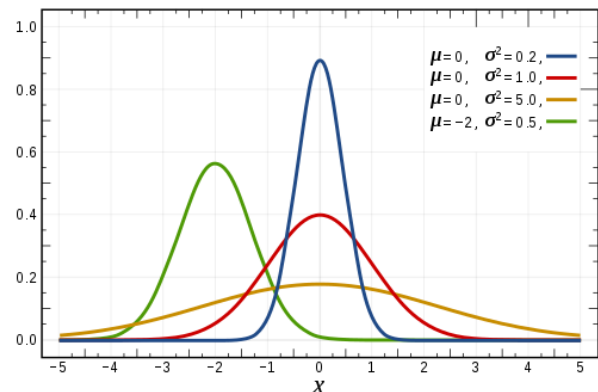
## The normal (PDF)

- The PDF of normal distribution is defined as:

$$PDF(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

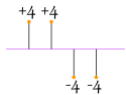
- With the following population **parameters**:

$$\mu = \frac{1}{N} \sum_{i=1}^N x_i \quad \sigma^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2$$



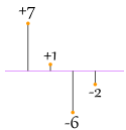
## Intermezzo: sum of squares

- Why use the sum of the squared difference from the mean (**sum of squares**)?
- Imagine we have the following 4 differences from the mean (indicated by the purple line) and we use the absolute value of the difference from mean to compute the variance:



$$\frac{1}{3} \cdot (|4| + |4| + |-4| + |-4|) = 5.3$$

- Now consider 4 differences that are more spread out (higher variance):



$$\frac{1}{3} \cdot (|7| + |1| + |-6| + |-2|) = 5.3$$

- If we use the squared differences we get a better result:

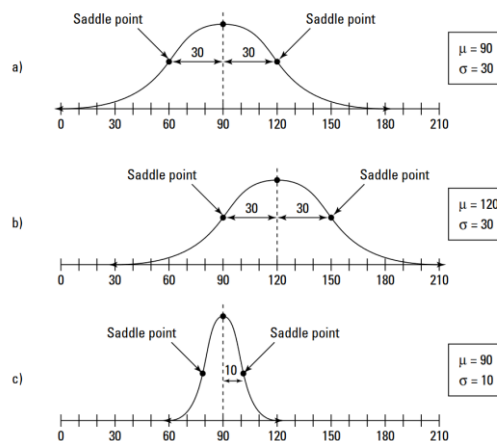
$$\frac{1}{3} \cdot (4^2 + 4^2 + (-4)^2 + (-4)^2) = 21.3 \quad < \quad \frac{1}{3} \cdot (7^2 + 1^2 + (-6)^2 + (-2)^2) = 30$$

## Other population parameters

- The **standard deviation** of a population variable is a more intuitive measure of spread of a variable and is computed as the square root of the variance:

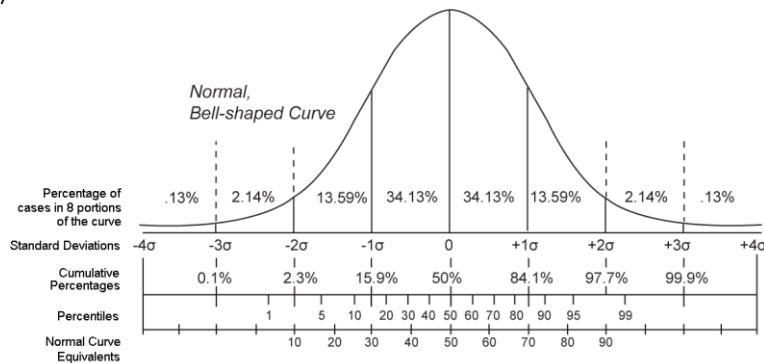
$$\sigma = \sqrt{\sigma^2}$$

- It is also the distance from the mean out to either saddle point in a normal PDF. The saddle points on each graph are where the PDF changes from concave down to concave up.



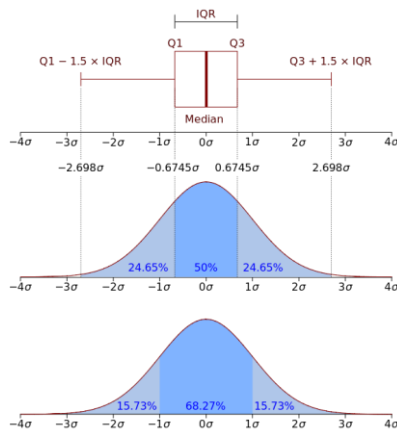
## Other population parameters

- About 68% of the individuals have a value within one standard deviation from the mean, while 95% of the individuals have a value within two standard deviations.
- The **median** is the value separating the higher half of a PDF from the lower half.
- The **quantiles** are cutpoints dividing the range of a PDF into continuous intervals with equal probabilities (e.g. quartiles, percentiles).



## Other population parameters

- A **boxplot** is a graphical representation of the quartiles Q1, Q2 and Q3 where the range of the data is shown with the assumption that every value lower than  $Q1 - 1.5 \times IQR$  or higher than  $Q3 + 1.5 \times IQR$  is considered an outlier.
- Boxplots are more practical when comparing the shapes of PDFs.



## The sample and uncertainty

Most of the time fiscal and practical constraints **limit** our access to the population.

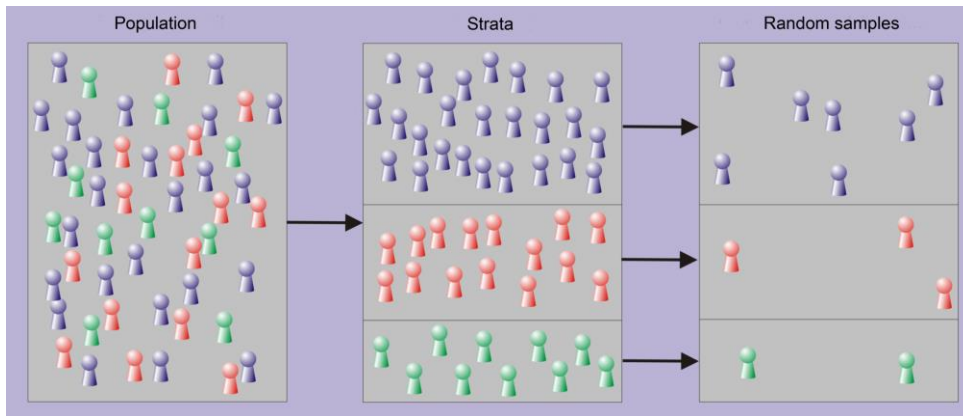
So, we have to make inferences about the population from a small subset of the population called a **sample (data)**.

- The **observations** in the sample must be **representative** of the population.
- Because of chance, working with a sample implies **uncertainty** about the inference.

## Types of sampling

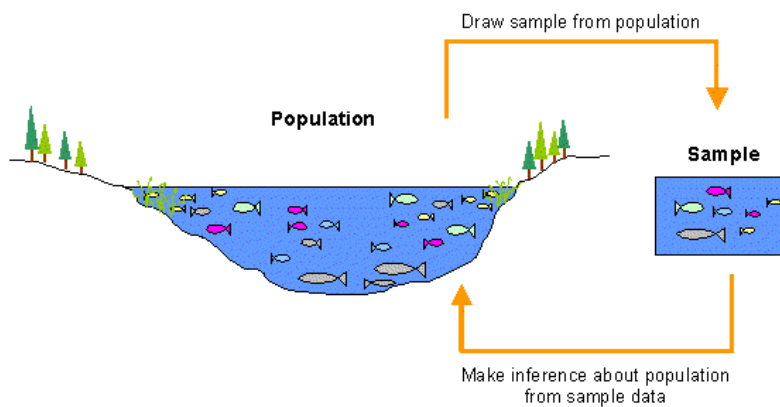
- **Simple random sampling**: each individual in the population has equal probability of being observed.
- **Stratified random sampling**: partition the population into "**strata**", then choose a simple random sample from each stratum with **equal proportions**. It is a mini-reproduction of the population.
- **Convenience sampling** is a matter of taking what you can get. It is an **accidental** sample, e.g. hospital data.

Beautiful google search image



fernuni-hagen.de

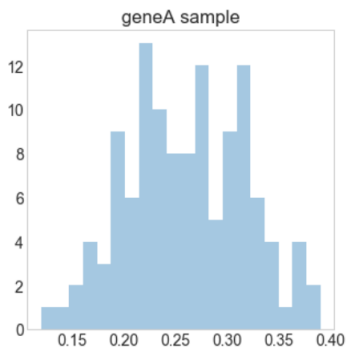
Beautiful google search image



<http://labs.geog.uvic.ca/geog226/frLab3.html>

## The distribution: what do we know about the PDF?

```
plt.figure(figsize=(5,5))
sns.distplot(geneA_sample,kde=False,bins=20)
plt.title("geneA sample")
plt.show()
```



- A **histogram** is a graphical representation of the **distribution** of (continuous) variable.
- To construct a histogram, the first step is to **bin** the range of values—that is, divide the entire range of values into a series of intervals—and then **count** how many values fall into each interval. The x-axis represents the bins, the y-axis represents the number of data points in a bin.
- The histogram reveals **probabilities** of ranges of the geneA expression level.

## The sample statistics

$$\bar{x} = \frac{1}{n-1} \sum_{i=1}^n x_i$$

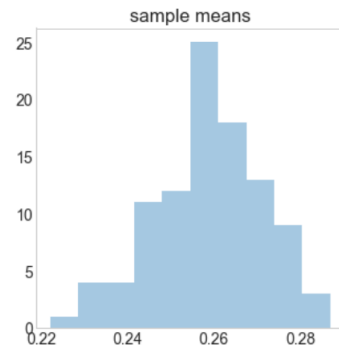
$$s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \mu)^2$$

- If we want to make inferences about the population using the sample we need to take into account the **uncertainty** that is implied by a **limited sample size  $n$** .

## Parameters and statistics

- The mean and the variance of the population are examples of a **parameter**. Parameters are summary measures of a population and as such are **fixed**.
- The sample mean and variance are known as **descriptive statistics** and are **variable** summary measures of a sample.
- This is illustrated by drawing 100 samples and computing the mean for each sample:

```
n=16
sample_means = []
for i in range(100):
    sample = np.random.normal(size=n, loc=0.26, scale=0.05)
    sample_means.append(np.mean(sample))
```



## Parameters and statistics

- The shape of this distribution is **again normal** and the mean is close to the population mean:

```
print "mean of sample means: %f" % np.mean(sample_means)
```

```
mean of sample means: 0.258972
```

- If we **increase** the number of samples we draw from the population then the mean of the sample means gets very close to the sample mean:

```
n=16
sample_means = []
for i in range(200000):
    sample = np.random.normal(size=n, loc=0.26, scale=0.05)
    sample_means.append(np.mean(sample))
print "mean of sample means: %f" % np.mean(sample_means)
```

```
mean of sample means: 0.260025
```



## The confidence interval

- The shape of the distribution of the sample means also has a standard deviation:

```
np.std(sample_means)
```

```
0.012559814200256193
```

- Suppose we would be able to draw all possible samples with fixed sample size  $n$  then the standard deviation the sample means is known as the **standard error of the mean (SEM)** and is equal to the standard deviation of the population divided by the square root of the sample size  $n$ .

```
print "SEM using population: %f" % (0.05/np.sqrt(n))
```

```
SEM using population: 0.012500
```

## The confidence interval

- Suppose we would be able to draw all possible samples with fixed sample size  $n$  then the standard deviation the sample means is known as the **standard error of the mean (SEM)** and is equal to the standard deviation of the population divided by the square root of the sample size  $n$ .

```
print "SEM using population: %f" % (0.05/np.sqrt(n))
```

```
SEM using population: 0.012500
```

- For one sample the SEM is estimated as  $SEM = \frac{s}{\sqrt{n}}$

```
sample = np.random.normal(size=n, loc=0.26, scale=0.05)
print "SEM: %f" % (np.std(sample)/np.sqrt(n))
```

```
SEM: 0.011680
```

## The confidence interval

- For one sample the SEM is estimated as:  $SEM = \frac{s}{\sqrt{n}}$
- SEM increases as the sample standard deviation  $s$  increases
- SEM decreases as the sample size  $n$  increases
- The square root of the sample size decreases the impact of the sample size  $n$  on the SEM as  $n$  increases.
- The SEM is used to compute a **confidence interval (CI)** for the mean of a population.

## The confidence interval

- The CI is an interval of values computed from the sample that is almost sure (set by a **level of confidence**) to cover the true population value.
- For instance, at a level of confidence of 95% the CI of the mean of a population is an interval of values computed from a sample drawn from this population that is 95% sure to cover the true population mean.
- The CI is compute using the SEM as:

$$[\bar{x} - t^*SEM, \bar{x} + t^*SEM]$$

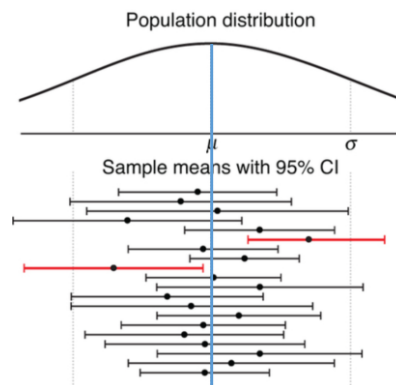
- where  $t^*$  is a **T-statistic** that follows a **t-distribution** with  $n-1$  degrees of freedom.
- Which t-distribution to use depends on the confidence level of the CI.

## The confidence interval

- Which t-distribution to use depends on the confidence level of the CI.

df	Upper-tail probability p											
	.25	.20	.15	.10	.05	.025	.02	.01	.005	.0025	.001	.0005
1	1.000	1.376	1.963	3.078	6.314	12.71	15.89	31.82	63.66	127.3	318.3	636.6
2	0.816	1.061	1.386	1.886	2.920	4.303	4.849	6.965	9.925	14.09	22.33	31.60
3	0.765	0.978	1.250	1.638	2.353	3.182	3.462	4.541	5.841	7.453	10.21	12.92
4	0.741	0.941	1.190	1.533	2.132	2.776	2.999	3.747	4.604	5.598	7.173	8.610
5	0.727	0.920	1.156	1.476	2.015	2.571	2.757	3.365	4.032	4.773	5.893	6.869
6	0.718	0.906	1.134	1.440	1.943	2.447	2.612	3.143	3.707	4.317	5.208	5.959
7	0.711	0.896	1.119	1.415	1.895	2.365	2.517	2.998	3.499	4.029	4.785	5.408
8	0.706	0.889	1.108	1.397	1.860	2.306	2.449	2.896	3.355	3.833	4.501	5.041
9	0.703	0.883	1.100	1.383	1.833	2.262	2.398	2.821	3.250	3.690	4.297	4.781
10	0.700	0.879	1.093	1.372	1.812	2.228	2.359	2.764	3.169	3.581	4.144	4.587
11	0.697	0.876	1.088	1.363	1.796	2.201	2.328	2.718	3.106	3.497	4.025	4.437
12	0.695	0.873	1.083	1.356	1.782	2.179	2.303	2.681	3.055	3.428	3.930	4.318
13	0.694	0.870	1.079	1.350	1.771	2.160	2.282	2.650	3.012	3.372	3.852	4.221
14	0.692	0.868	1.076	1.345	1.761	2.145	2.264	2.624	2.977	3.326	3.787	4.140
15	0.691	0.866	1.074	1.341	1.753	2.131	2.249	2.602	2.947	3.286	3.733	4.073
16	0.690	0.865	1.071	1.337	1.746	2.120	2.235	2.583	2.921	3.252	3.686	4.015
17	0.689	0.863	1.069	1.333	1.740	2.110	2.224	2.567	2.898	3.222	3.646	3.965
18	0.688	0.862	1.067	1.330	1.734	2.101	2.214	2.552	2.878	3.197	3.611	3.922
19	0.688	0.861	1.066	1.328	1.729	2.093	2.205	2.539	2.861	3.174	3.579	3.883
20	0.687	0.860	1.064	1.325	1.725	2.086	2.197	2.528	2.845	3.153	3.552	3.850
21	0.686	0.859	1.063	1.323	1.721	2.080	2.189	2.518	2.831	3.135	3.527	3.819
22	0.686	0.858	1.061	1.321	1.717	2.074	2.183	2.508	2.819	3.119	3.505	3.792
23	0.685	0.858	1.060	1.319	1.714	2.069	2.177	2.500	2.807	3.104	3.485	3.768
24	0.685	0.857	1.059	1.318	1.711	2.064	2.172	2.492	2.797	3.091	3.467	3.745
25	0.684	0.856	1.058	1.316	1.708	2.060	2.167	2.485	2.787	3.078	3.450	3.725
26	0.684	0.856	1.058	1.315	1.706	2.056	2.162	2.479	2.779	3.067	3.435	3.707
27	0.684	0.855	1.057	1.314	1.703	2.052	2.158	2.473	2.771	3.057	3.421	3.690
28	0.683	0.855	1.056	1.313	1.701	2.048	2.154	2.467	2.763	3.047	3.408	3.674
29	0.683	0.854	1.055	1.311	1.699	2.045	2.150	2.462	2.756	3.038	3.396	3.659
30	0.683	0.854	1.055	1.310	1.697	2.042	2.147	2.457	2.750	3.030	3.385	3.646
40	0.681	0.851	1.050	1.303	1.684	2.021	2.123	2.423	2.704	2.971	3.307	3.551
50	0.679	0.849	1.047	1.299	1.676	2.009	2.109	2.403	2.678	2.937	3.261	3.496
60	0.679	0.848	1.045	1.296	1.671	2.000	2.099	2.390	2.660	2.915	3.232	3.460
80	0.678	0.846	1.043	1.292	1.664	1.990	2.088	2.374	2.639	2.887	3.195	3.416
100	0.677	0.845	1.042	1.290	1.660	1.984	2.081	2.364	2.626	2.871	3.174	3.390
1000	0.675	0.842	1.037	1.282	1.646	1.962	2.056	2.330	2.581	2.813	3.098	3.300
z*	0.674	0.841	1.036	1.282	1.645	1.960	2.054	2.326	2.576	2.807	3.091	3.291
Confidence level C												
	50%	60%	70%	80%	90%	95%	96%	98%	99%	99.5%	99.8%	99.9%

## The confidence interval



## The confidence interval

```

confidence = 0.95
num_samples = 1000
n = 16

num_wrong = 0
for i in range(0, num_samples):
    m, l, h = CI_mean(np.random.normal(size=n, loc=0.26, scale=0.05), confidence)
    if (l >= 0.26) | (h <= 0.26):
        num_wrong += 1
prob = (1. - (float(num_wrong)/num_samples))
print "Probability that population mean is within sample CI: %f" % prob

```

Probability that population mean is within sample CI: 0.953000

A random sample of 300 diastolic blood pressure measurements are taken. Suppose a 99% confidence interval for the population mean diastolic blood pressure is 68 to 73 mm Hg. If a 95% confidence interval is also calculated, then

- (a) The 95% confidence interval will be wider than the 99%.
- (b) The 95% confidence interval will be narrower than the 99%.
- (c) 95% and 99% confidence interval will be the same.
- (d) One cannot make a general statement about whether the 95% confidence interval would be narrower, wider or the same as the 99%.

## Types of statistical analysis

- **Descriptive statistics** quantitatively describe or summarize one or more variables in a sample. They are typically the first kind of analysis performed on the data set.
- **Inferential statistics** quantitatively describe or summarize parameters of the population using a sample. Again, this is based mainly on assumptions made about the population.
- **Predictive modeling** is about using statistics to predict outcomes (typically of future events). In this case each observation is described by an **independent variable** (the variable to be predicted) and one or more **dependent variables** (used to compute the predictions).
- **Exploratory data analysis** summarizes the main characteristics of the variables in a sample, often with visual methods. It is an approach to analyze data sets to find **previously unknown relationships**.

## inferential statistics

## Hypothesis testing

- In inferential statistics we make a **claim** about the population and use one or more samples to estimate the likelihood of this claim by making assumptions about the sample(s).
- Consider a lab where protein mixtures with known concentration for each protein are produced.
- Each day the machine is tested by sampling  $n=9$  mixtures and measuring their protein concentrations.
- Let's focus on one protein proteinA. The mean proteinA concentration should be 0.3 with a variance between the mixtures smaller than 0.01.
- We make the following assumptions:
  - the proteinA concentration distribution in the mixtures is normal
  - the mixtures are drawn independently (e.g. by simple random sampling)
- Now we make the following claim: **the mean proteinA concentration of the population from which the sample is drawn equals 0.3.**

## Hypothesis testing

- Now we make the following claim: **the mean proteinA concentration  $\mu$  of the population from which the sample is drawn equals  $\mu_0 = 0.3$ .**
- We draw a sample at random from a distribution with  $\mu=0.3$  and  $\sigma=0.05$ :

```
n=9
#loc = mean, scale = standard deviation
x = np.random.normal(size=n,loc=0.3,scale=0.05)
```

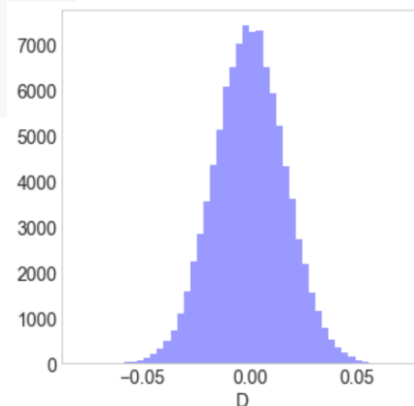
```
mu0 = 0.3
D_observed = np.mean(x)-mu0
print "observed mean difference: %f" % D_observed
```

```
observed mean difference: -0.010158
```

- Is this difference **significant**?

## Hypothesis testing

```
m = 100000
mean_diffs = []
for i in range(m):
    x_tmp = np.random.normal(size=n, loc=0.3, scale=0.05)
    D = np.mean(x_tmp) - mu0
    mean_diffs.append(D)
plt.figure(figsize=(5,5))
sns.distplot(mean_diffs, color="b", kde=False)
plt.xlabel("D")
plt.show()
```

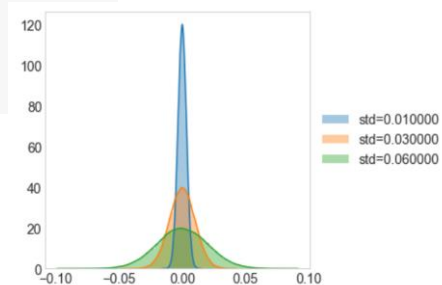


- The distribution of  $D$  is called the **null distribution** for the one sample t-test.

## Hypothesis testing: null distribution

- The distribution of  $D$  is called the **null distribution** for the one sample t-test.
- The shape of the null distribution depends strongly on the **sample size**  $n$  and the **variance**  $\sigma^2$  of the population from which the samples were drawn (so not on the actual value of  $\mu$  itself).

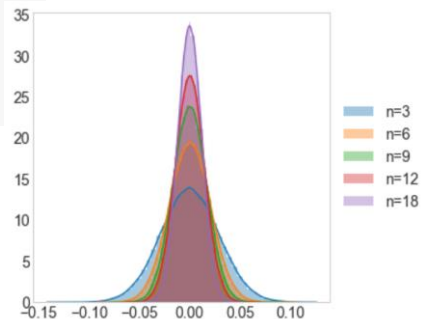
```
plt.figure(figsize=(5,5))
for sigma_new in [0.01, 0.03, 0.06]:
    mean_diffs = []
    for i in range(m):
        x_tmp = np.random.normal(size=n, loc=0.3, scale=sigma_new)
        D = np.mean(x_tmp) - mu0
        mean_diffs.append(D)
    sns.distplot(mean_diffs, label="std=%f"%sigma_new)
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))
plt.show()
```



## Hypothesis testing: null distribution

- The distribution of  $D$  is called the **null distribution** for the one sample t-test.
- The shape of the null distribution depends strongly on the **sample size**  $n$  and the **variance**  $\sigma^2$  of the population from which the samples were drawn (so not on the actual value of  $\mu$  itself).

```
plt.figure(figsize=(5,5))
for n in [3,6,9,12,18]:
    mean_diffs = []
    for i in range(m):
        x_tmp = np.random.normal(size=n, loc=0.3, scale=0.05)
        D = np.mean(x_tmp) - mu0
        mean_diffs.append(D)
    sns.distplot(mean_diffs, label="n=%i"%n)
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))
plt.show()
```



## Hypothesis testing: one sample t-test

- The one sample t-test claims that the mean of the population from which a sample was drawn equals some value  $\mu_0$ .
  - It tests the null hypothesis  $H_0$  which states that the population mean equals  $\mu_0$ .
  - This test is performed by computing a **test statistic** and evaluating the probability (p-value) of observing the test statistic value under the null distribution of this test statistic.
  - If the p-value is below a **certain significance level alpha** the null hypothesis is rejected and the alternative hypothesis  $H_1$  that the population mean does not equal  $\mu_0$  is accepted.
    - The null hypothesis ( $H_0$ ) states that  $\mu = \mu_0$ .
    - The two-tailed hypothesis ( $H_1$ ) states that  $\mu \neq \mu_0$ .
- There are also two possible one-tailed hypothesis:
- The upper-tailed hypothesis ( $H_1$ ) states that  $\mu > \mu_0$ .
  - The lower-tailed hypothesis ( $H_1$ ) states that  $\mu < \mu_0$ .



## Hypothesis testing: one sample t-test

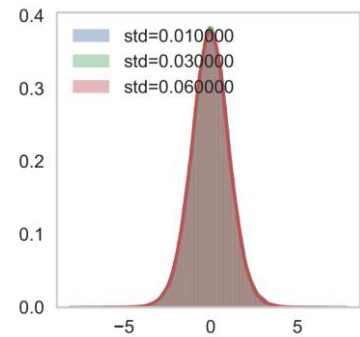
- The test statistic computed by the one sample t-test to test the null hypothesis is called the T-statistic and is computed as

$$T = \frac{\bar{x} - \mu}{s_{n-1}^2 / \sqrt{n}}$$

- If the null hypothesis is true then the T-statistic follows a Student's t-distribution with  $n-1$  degrees of freedom.
- A p-value is computed from the Student's t-distribution and a significance level alpha is used to make the final conclusion.
- The one sample t-test assumes that
  - the population is normal
  - the sample observations are drawn independently

```
import scipy.stats as stats
print stats.ttest_1samp(np.array(x),mu0)
```

```
Ttest_1sampResult(statistic=-0.4467647978644402, pvalue=0.6668917378539923)
```

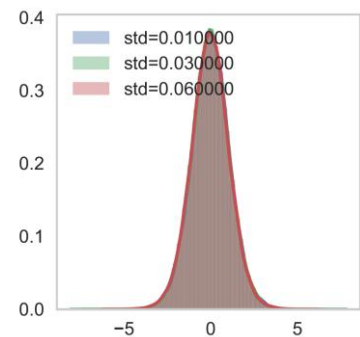


## Hypothesis testing: one sample t-test

```
x_bad = np.random.normal(size=n, loc=0.37, scale=0.05)
print "Mean bad sample: %f" % np.mean(x_bad)
print stats.ttest_1samp(np.array(x_bad), mu0)
```

```
Mean bad sample: 0.370894
```

```
Ttest_1sampResult(statistic=7.432071395989882, pvalue=9.800010644648027e-07)
```



## Hypothesis testing: (unpaired) two sample t-test

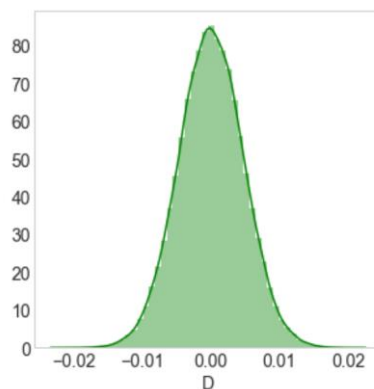
- The unpaired two sample t-test for equal means is a hypothesis test that claims that the means of two samples  $x_1$  with sample size  $n_1$  and  $x_2$  with sample size  $n_2$  are equal.
  - The null hypothesis  $H_0$  states that  $\bar{x}_1 = \bar{x}_2$ .
  - The two-tailed alternative hypothesis  $H_1$  states that  $\bar{x}_1 \neq \bar{x}_2$ .
- The T-statistic that is used to test the null hypothesis is

$$T = \frac{\bar{x}_1 - \bar{x}_2}{SE(\bar{x}_1 - \bar{x}_2)} \quad SE(\bar{x}_1 - \bar{x}_2) = s_p \sqrt{\frac{1}{n_1} + \frac{1}{n_2}} \quad s_p = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}}$$

- If the null hypothesis is true then this T-statistic follows a Student's t-distribution with  $n_1 + n_2 - 2$  degrees of freedom.

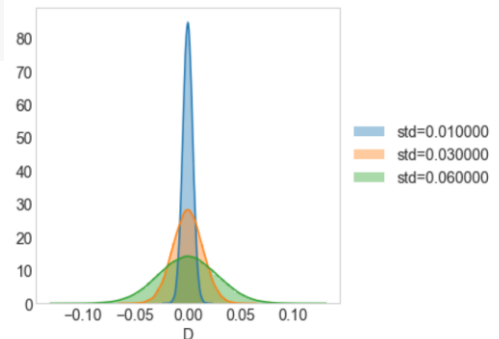
## Hypothesis testing: (unpaired) two sample t-test

```
n = 9
null_distribution = []
for i in range(m):
    x1 = np.random.normal(size=n, loc=0.26, scale=0.01)
    x2 = np.random.normal(size=n, loc=0.26, scale=0.01)
    D = (np.mean(x1) - np.mean(x2))
    null_distribution.append(D)
plt.figure(figsize=(5,5))
sns.distplot(np.array(null_distribution), color="g")
plt.xlabel("D")
plt.show()
```



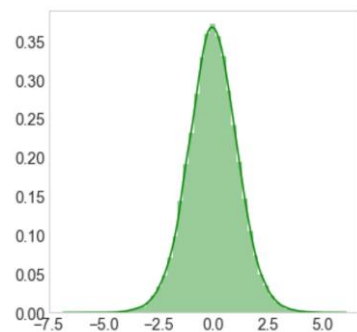
## Hypothesis testing: (unpaired) two sample t-test

```
plt.figure(figsize=(5,5))
for sigma_gene_new in [0.01,0.03,0.06]:
    null_distribution_new = []
    for i in range(m):
        x1 = np.random.normal(size=n,loc=0.26,scale=sigma_gene_new)
        x2 = np.random.normal(size=n,loc=0.26,scale=sigma_gene_new)
        D = (np.mean(x1)-np.mean(x2))
        null_distribution_new.append(D)
    sns.distplot(np.array(null_distribution_new),label="std=%f"%sigma_gene_new)
plt.xlabel("D")
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))
plt.show()
```



## Hypothesis testing: (unpaired) two sample t-test

```
n = 9
null_distribution = []
for i in range(m):
    x1 = np.random.normal(size=n,loc=0.26,scale=0.01)
    x2 = np.random.normal(size=n,loc=0.26,scale=0.01)
    null_distribution.append(compute_T_statistic(x1,x2))
plt.figure(figsize=(5,5))
sns.distplot(np.array(null_distribution),color="g")
plt.show()
```



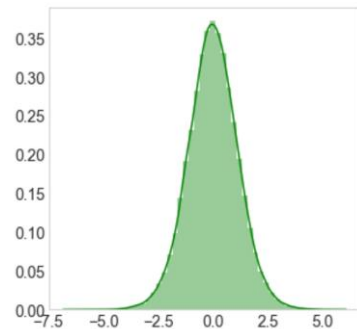
## Hypothesis testing: (unpaired) two sample t-test

```
geneA = np.random.normal(size=n,loc=0.26,scale=0.01)
print "Mean geneA: %f" % np.mean(geneA)
geneB = np.random.normal(size=n,loc=0.26,scale=0.01)
print "Mean geneB: %f" % np.mean(geneB)
print "Mean difference: %f" % (np.mean(geneA)-np.mean(geneB))
T_observed = compute_T_statistic(geneA,geneB)
print "T-statistic: %f" % T_observed
```

```
Mean geneA: 0.254477
Mean geneB: 0.263379
Mean difference: -0.008902
T-statistic: -1.898559
```

```
print "p-value=%f"%stats.ttest_ind(geneA,geneB)[1]
```

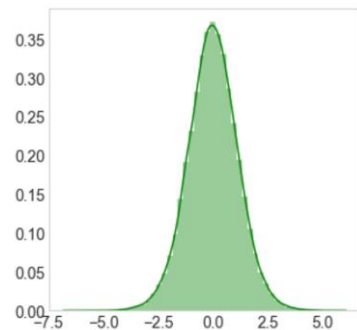
```
p-value=0.092398
```



## Hypothesis testing: (unpaired) two sample t-test

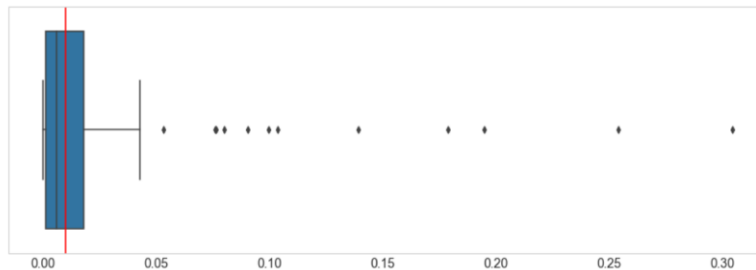
```
geneA = np.random.normal(size=n,loc=0.26,scale=0.01)
print "Mean geneA: %f" % np.mean(geneA)
geneB = np.random.normal(size=n,loc=0.275,scale=0.01)
print "Mean geneB: %f" % np.mean(geneB)
print "Mean difference: %f" % (np.mean(geneA)-np.mean(geneB))
T_observed = compute_T_statistic(geneA,geneB)
print "T-statistic: %f" % T_observed
print "p-value=%f"%stats.ttest_ind(geneA,geneB)[1]
```

```
Mean geneA: 0.260193
Mean geneB: 0.275073
Mean difference: -0.014880
T-statistic: -6.343262
p-value=0.000019
```



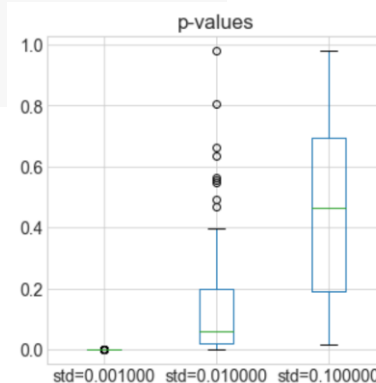
## Hypothesis testing: (unpaired) two sample t-test

```
pvalues = []
for i in range(100):
    geneA = np.random.normal(size=n, loc=0.26, scale=0.01)
    geneB = np.random.normal(size=n, loc=0.275, scale=0.01)
    pvalues.append(stats.ttest_ind(geneA, geneB).pvalue)
df = pd.DataFrame()
df["p-values"] = pvalues
plt.figure(figsize=(15,5))
sns.boxplot(df, showfliers=True)
plt.axvline(x=0.01, c='r')
plt.show()
```



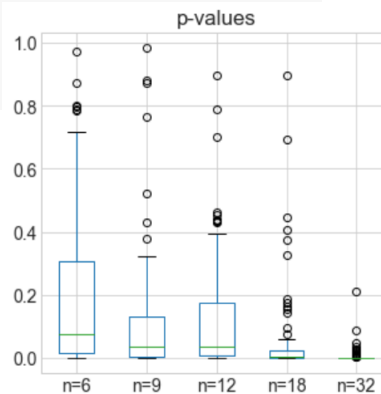
## Hypothesis testing: (unpaired) two sample t-test

```
df = pd.DataFrame()
for sigma in [0.001, 0.01, 0.1]:
    pvalues = []
    for i in range(100):
        geneA = np.random.normal(size=n, loc=0.26, scale=sigma)
        geneB = np.random.normal(size=n, loc=0.27, scale=sigma)
        pvalues.append(stats.ttest_ind(geneA, geneB).pvalue)
    df["std=%f"%sigma] = pvalues
plt.figure(figsize=(5,5))
df.boxplot()
plt.title("p-values")
plt.show()
```



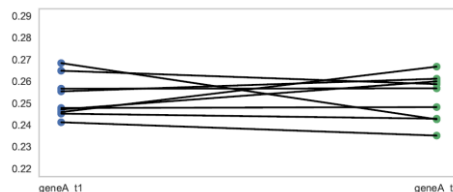
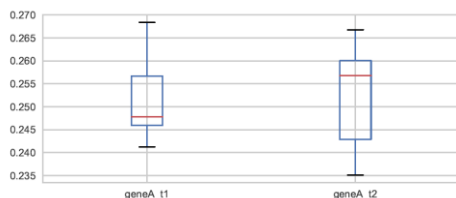
## Hypothesis testing: (unpaired) two sample t-test

```
df = pd.DataFrame()
for n_tmp in [6,9,12,18,32]:
    pvalues = []
    for i in range(100):
        geneA = np.random.normal(size=n_tmp,loc=0.26,scale=0.01)
        geneB = np.random.normal(size=n_tmp,loc=0.27,scale=0.01)
        pvalues.append(stats.ttest_ind(geneA,geneB).pvalue)
    df["n=%i"%n_tmp] = pvalues
plt.figure(figsize=(5,5))
df.boxplot()
plt.title("p-values")
plt.show()
```



## Hypothesis testing: paired two sample t-test

- The paired two sample t-test for equal means is a hypothesis test that claims that the means of two samples  $x_1$  with sample size  $n_1$  and  $x_2$  with sample size  $n_2$  are equal in the case where observations in one sample can be paired with observations in the other sample.



```
print stats.ttest_ind(geneA_t1,geneA_t2)
```

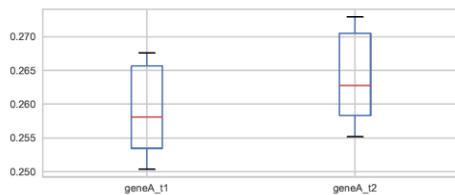
```
Ttest_indResult(statistic=0.53023275217923627, pvalue=0.60323220614606166)
```

```
print stats.ttest_rel(geneA_t1,geneA_t2)
```

```
Ttest_relResult(statistic=0.5260615713801049, pvalue=0.6131126885750664)
```

## Hypothesis testing: paired two sample t-test

- The paired two sample t-test for equal means is a hypothesis test that claims that the means of two samples  $x_1$  with sample size  $n_1$  and  $x_2$  with sample size  $n_2$  are equal in the case where observations in one sample can be paired with observations in the other sample.



```
print "unpaired test: %f" % stats.ttest_ind(geneA_t1,geneA_t2).pvalue
print "paired test: %f" % stats.ttest_rel(geneA_t1,geneA_t2).pvalue
```

```
unpaired test: 0.447869
paired test: 0.000000
```

## Hypothesis testing: paired two sample t-test

- The paired two sample t-test for equal means is a hypothesis test that claims that the means of two samples  $x_1$  with sample size  $n_1$  and  $x_2$  with sample size  $n_2$  are equal in the case where observations in one sample can be paired with observations in the other sample.

$$T = \frac{\bar{d}}{SE(\bar{d})} \quad SE(\bar{d}) = \frac{s_d}{\sqrt{n}}$$

- If the null hypothesis is true then this T-statistic follows a Student's t-distribution with  $n - 1$  degrees of freedom.

## Hypothesis testing: the Mann-Whitney U-Test

- The Mann-Whitney U-test is a **non-parametric** hypothesis test that claims that the medians of two samples are equal.
- Non-parametric tests are also called **distribution-free** tests because they don't assume that your data follow a specific distribution. However the test does assume that the two distributions are similar in shape.
- The test statistic is called the U-statistic that again follows a known distribution if the null hypothesis is true.

## Hypothesis testing: the Mann-Whitney U-Test

- The **U-statistic** is computed as follows.

Suppose we have a sample of  $n_x$  observations  $\{x_1, x_2, \dots, x_n\}$  in one group (i.e. from one population) and a sample of  $n_y$  observations  $\{y_1, y_2, \dots, y_n\}$  in another group (i.e. from another population).

1. Arrange all the observations in order of magnitude.
2. Under each observation, write down  $X$  or  $Y$  (or some other relevant symbol) to indicate which sample they are from.
3. Under each  $x$  write down the number of  $y$ s which are to the left of it (i.e. smaller than it); this indicates  $x_i > y_j$ . Under each  $y$  write down the number of  $x$ s which are to the left of it (i.e. smaller than it); this indicates  $y_j > x_i$
4. Add up the total number of times  $x_i > y_j$  — denote by  $U_x$ . Add up the total number of times  $y_j > x_i$  — denote by  $U_y$ . Check that  $U_x + U_y = n_x n_y$ .
5. Calculate  $U = \min(U_x, U_y)$
6. Use statistical tables for the Mann-Whitney U test to find the probability of observing a value of  $U$  or lower. If the test is one-sided, this is your p-value; if the test is a two-sided test, double this probability to obtain the p-value.

Age	11	12	16	17	19	20	22	24	29
M/F	F	F	M	F	M	F	M	M	M
$M > F$			2		3		4	4	4
$F > M$	0	0		1		2			

$$U_M = 2 + 3 + 4 + 4 + 4 = 17$$

$$U_F = 0 + 0 + 1 + 2 = 3$$

$$U = \min(U_M, U_F) = 3$$



## Hypothesis testing: the Mann-Whitney U Test

**Table 3** Critical values of U (5% significance).

$n_1 \backslash n_2$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1																				
2								0	0	0	0	1	1	1	1	1	2	2	2	2
3					0	1	1	2	2	3	3	4	4	5	5	6	6	7	7	8
4				0	1	2	3	4	4	5	6	7	8	9	10	11	11	12	13	13
5			0	1	2	3	5	6	7	8	9	11	12	13	14	15	17	18	19	20
6			1	2	3	5	6	8	10	11	13	14	16	17	19	21	22	24	25	27
7			1	3	5	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
8		0	2	4	6	8	10	13	15	17	19	22	24	26	29	31	34	36	38	41
9		0	2	4	7	10	12	15	17	20	23	26	28	31	34	37	39	42	45	48
10		0	3	5	8	11	14	17	20	23	26	29	33	36	39	42	45	48	52	55
11		0	3	6	9	13	16	19	23	26	30	33	37	40	44	47	51	55	58	62
12		1	4	7	11	14	18	22	26	29	33	37	41	45	49	53	57	61	65	69
13		1	4	8	12	16	20	24	28	33	37	41	45	50	54	59	63	67	72	76
14		1	5	9	13	17	22	26	31	36	40	45	50	55	59	64	67	74	78	83
15		1	5	10	14	19	24	29	34	39	44	49	54	59	64	70	75	80	85	90
16		1	6	11	15	21	26	31	37	42	47	53	59	64	70	75	81	86	92	98
17		2	6	11	17	22	28	34	39	45	51	57	63	67	75	81	87	93	99	105
18		2	7	12	18	24	30	36	42	48	55	61	67	74	80	86	93	99	106	112
19		2	7	13	19	25	32	38	45	52	58	65	72	78	85	92	99	106	113	119
20		2	8	13	20	27	34	41	48	55	62	69	76	83	90	98	105	112	119	127

## Hypothesis testing: the Mann-Whitney U-Test

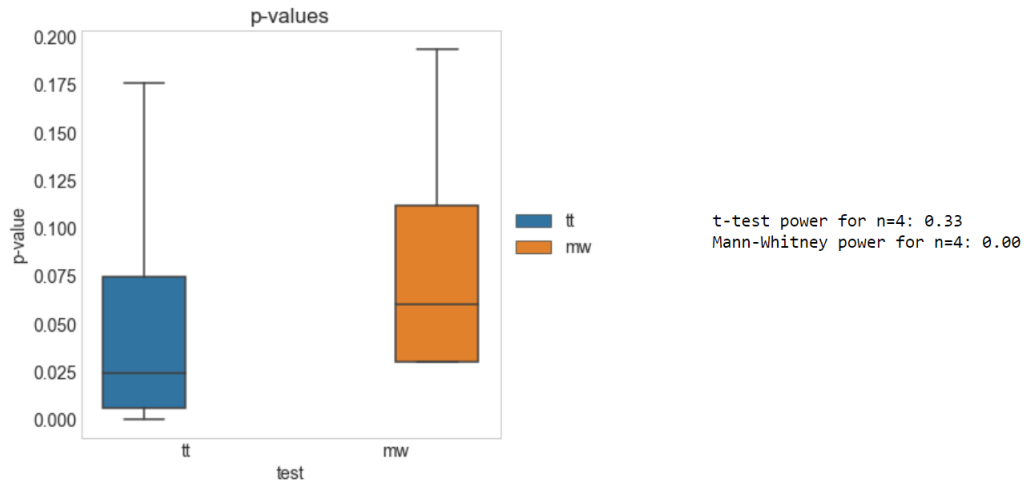
- So, it there exists a non-parametric test then why use a parametric test?
- The answer is that the parametric test will have more statistical power, especially for small sample sizes.

```

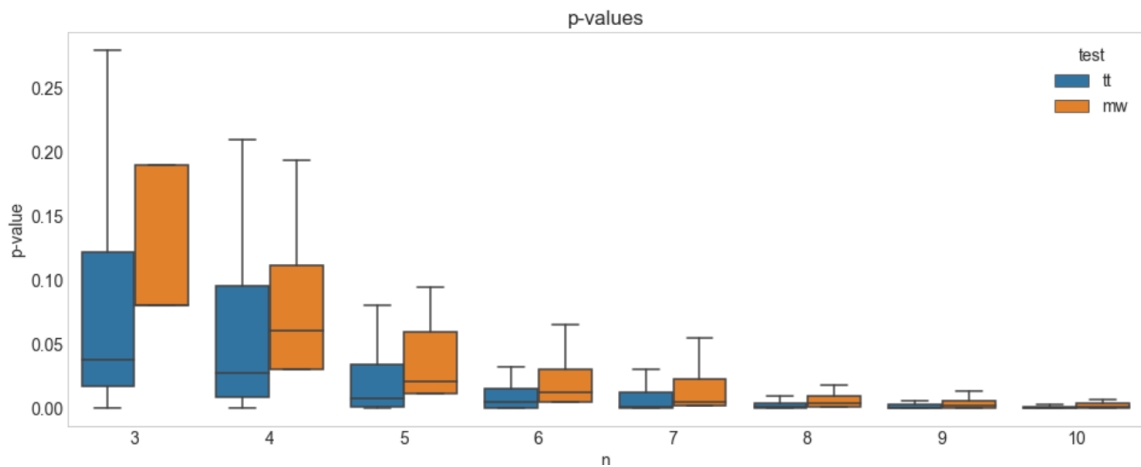
n = 4
alpha = 0.01
df = pd.DataFrame()
pvalues = []
tests = []
correct_tt = 0
correct_mw = 0
for i in range(1000):
    p1 = np.random.normal(size=n, loc=0.26, scale=0.01)
    p2 = np.random.normal(size=n, loc=0.28, scale=0.01)
    pvalue = stats.ttest_ind(p1, p2).pvalue
    pvalues.append(pvalue)
    if pvalue <= alpha:
        correct_tt += 1
    tests.append("tt")
    pvalue = stats.mannwhitneyu(p1, p2, alternative="two-sided").pvalue
    pvalues.append(pvalue)
    if pvalue <= alpha:
        correct_mw += 1
    tests.append("mw")
power_tt = correct_tt/1000.
print "t-test power for n=%i: %.2f" % (n, power_tt)
power_mw = correct_mw/1000.
print "Mann-Whitney power for n=%i: %.2f" % (n, power_mw)
df = pd.DataFrame()
df["p-value"] = pvalues
df["test"] = tests
plt.figure(figsize=(6,6))
sns.boxplot(x="test", y="p-value", hue="test", data=df, showfliers=False)
plt.title("p-values")
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))
plt.show()

```

## Hypothesis testing: the Mann-Whitney U-Test



## Hypothesis testing: the Mann-Whitney U-Test



## One-way Analysis Of Variance (ANOVA)

- One-way ANOVA is a statistical method to test the claim that the means of **two or more ( $k$ )** samples  $x_i$  with sample size  $n_i$  ( $i=1 \dots k$ ) are equal.
- The test makes the following assumptions:
  - the shape of the population from which the samples are drawn should be normal
  - all samples have equal variance
  - all samples should be drawn independently
- Again there are two hypothesis:
  - The null hypothesis  $H_0$  states that the means  $\bar{x}_i$  with  $i = 1 \dots k$  are equal.
  - The alternative hypothesis  $H_1$  states that the means  $\bar{x}_i$  with  $i = 1 \dots k$  are not equal.
- One-way ANOVA is a generalization of the two sample t-test.

## One-way ANOVA is a generalization of the two sample t-test.

- One-way ANOVA is a generalization of the two sample t-test.

```
n = 9
geneA = np.random.normal(size=n,loc=0.26,scale=0.01)
geneB = np.random.normal(size=n,loc=0.28,scale=0.01)
print "two sample t-test:"
print stats.ttest_ind(geneA,geneB)
print "one-way ANOVA:"
print stats.f_oneway(geneA,geneB)
```

two sample t-test:

Ttest\_indResult(statistic=-4.022327404376181, pvalue=0.0009847119072361069)

one-way ANOVA:

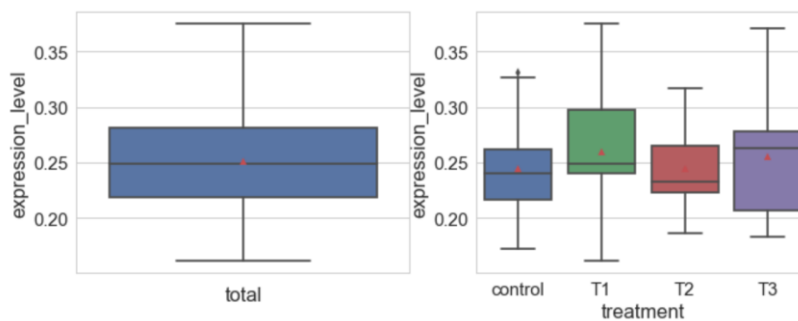
F\_onewayResult(statistic=16.179117747995637, pvalue=0.0009847119072361058)

## One-way Analysis Of Variance (ANOVA)

```
n = 4
sample_control = np.random.normal(size=n, loc=0.26, scale=0.05)
sample_T1 = np.random.normal(size=n, loc=0.26, scale=0.05)
sample_T2 = np.random.normal(size=n, loc=0.26, scale=0.05)
sample_T3 = np.random.normal(size=n, loc=0.26, scale=0.05)
```

	treatment	expression_level
0	control	0.276610
1	control	0.268073
2	control	0.256852
3	control	0.249141
4	T1	0.252675
5	T1	0.247875
6	T1	0.280871
7	T1	0.261644
8	T2	0.281502
9	T2	0.257326
10	T2	0.271810
11	T2	0.281779
12	T3	0.256650
13	T3	0.270311
14	T3	0.249154
15	T3	0.246365

## One-way Analysis Of Variance (ANOVA)



	treatment	expression_level
0	control	0.276610
1	control	0.268073
2	control	0.256852
3	control	0.249141
4	T1	0.252675
5	T1	0.247875
6	T1	0.280871
7	T1	0.261644
8	T2	0.281502
9	T2	0.257326
10	T2	0.271810
11	T2	0.281779
12	T3	0.256650
13	T3	0.270311
14	T3	0.249154
15	T3	0.246365

## One-way Analysis Of Variance (ANOVA)

- ANOVA computes a **F-statistic** that is simply a ratio of two variances.
- Where the two sample t-test T-statistic looks at the difference in mean between two samples, the ANOVA F-statistics looks at the ratio of two variances.
- The first variance in the F-statistic is the variation between the  $k=4$  sample means. It is computed as what is known as the treatment sum of squares SST:

$$SST = \sum_{i=1}^k n_i (\bar{x}_i - \bar{x}_{total})$$

- The second variation is the variation within the sample. This is computed as the sum of squares of the error SSE:

$$SSE = \sum_{i=1}^k \sum_{j=1}^{n_i} (x_{ij} - \bar{x}_i)$$

## One-way Analysis Of Variance (ANOVA)

- The variance of the expression level in the dataset (not taking treatment into account) is computed by the total sum of squares TSS:

$$TSS = \sum_{i=1}^k \sum_{j=1}^{n_i} (x_{ij} - \bar{x}_{total})^2$$

- So, ANOVA partitions the total variance (TSS) into a part caused by the treatment (SST) and a part the is not explained (SSE), i.e.  $TSS = SST + SEE$ .
- explained / unexplained
- between group / within group

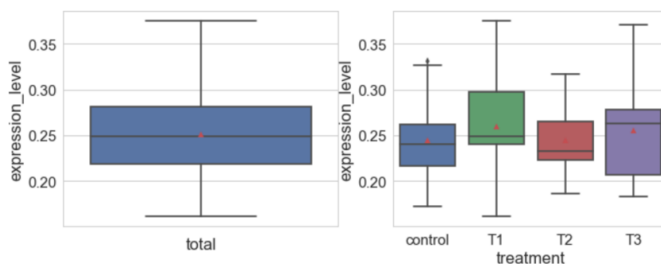
## One-way Analysis Of Variance (ANOVA)

- The F-statistic computed by ANOVA is the ratio of SST and SSE (with some variance normalization):

$$F = \frac{SST/(k - 1)}{SSE/(n_t - k)}$$

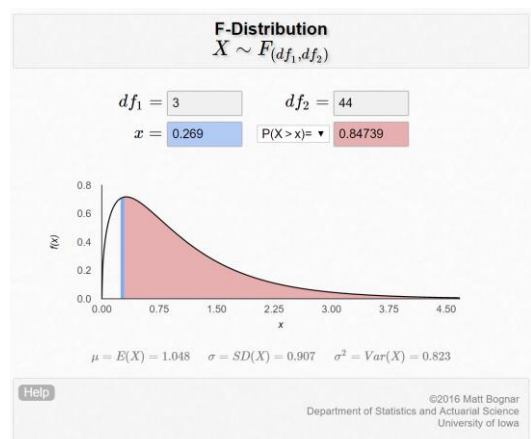
- So, ANOVA partitions the total variance (TSS) into a part caused by the treatment (SST) and a part that is with unknown source (SSE).
- (explained / unexplained) or (between group / within group)
- If the null hypothesis is true then the F-statistic follows an F-distribution with  $df_1=k-1$  and  $df_2=n_t-k$  degrees of freedom.

## One-way Analysis Of Variance (ANOVA)

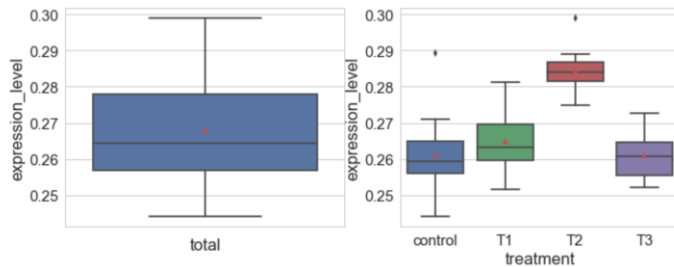


```
import statsmodels.api as sm
from statsmodels.formula.api import ols
mod = ols('expression_level ~ treatment',
          data=df).fit()
aov_table = sm.stats.anova_lm(mod, typ=2)
print aov_table
```

	sum_sq	df	F	PR(>F)
treatment	0.001968	3.0	0.269032	0.847367
Residual	0.107293	44.0	NaN	NaN

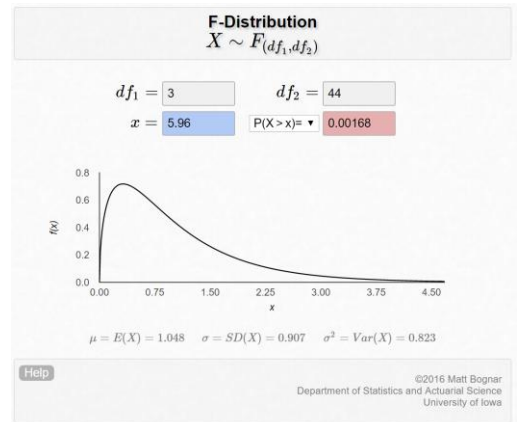


## One-way Analysis Of Variance (ANOVA)



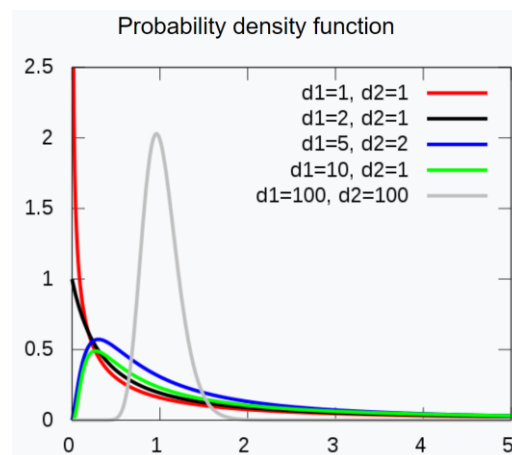
```
mod = ols('expression_level ~ treatment',
          data=df).fit()
aov_table = sm.stats.anova_lm(mod, typ=2)
print aov_table
```

	sum_sq	df	F	PR(>F)
treatment	0.001384	3.0	5.945948	0.001702
Residual	0.003413	44.0	NaN	NaN



<http://homepage.divms.uiowa.edu/~mbognar/applets/f.html>

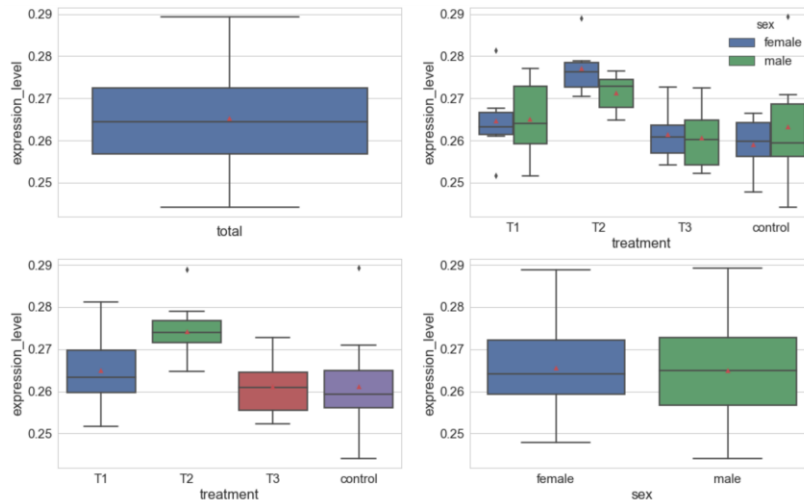
## One-way Analysis Of Variance (ANOVA)



<https://en.wikipedia.org/wiki/F-distribution>

## Two-way Analysis Of Variance (ANOVA)

- In two-way ANOVA there are two factors.



	treatment	sex	expression_level
0	control	female	0.276610
1	control	male	0.268073
2	control	female	0.256852
3	control	male	0.249141
4	T1	female	0.252675
5	T1	male	0.247875
6	T1	female	0.280871
7	T1	male	0.261644
8	T2	female	0.281502
9	T2	male	0.257326
10	T2	female	0.271810
11	T2	male	0.281779
12	T3	female	0.256650
13	T3	male	0.270311
14	T3	female	0.249154
15	T3	male	0.246365

## Two-way Analysis Of Variance (ANOVA)

```
mod = ols('expression_level ~ treatment',
          data=df).fit()
aov_table = sm.stats.anova_lm(mod, typ=2)
print aov_table
```

	sum_sq	df	F	PR(>F)
treatment	0.001384	3.0	5.945948	0.001702
Residual	0.003413	44.0	NaN	NaN

```
mod = ols('expression_level ~ sex',
          data=df).fit()
aov_table = sm.stats.anova_lm(mod, typ=2)
print aov_table
```

	sum_sq	df	F	PR(>F)
sex	0.000003	1.0	0.033092	0.85645
Residual	0.004793	46.0	NaN	NaN

	treatment	sex	expression_level
0	control	female	0.276610
1	control	male	0.268073
2	control	female	0.256852
3	control	male	0.249141
4	T1	female	0.252675
5	T1	male	0.247875
6	T1	female	0.280871
7	T1	male	0.261644
8	T2	female	0.281502
9	T2	male	0.257326
10	T2	female	0.271810
11	T2	male	0.281779
12	T3	female	0.256650
13	T3	male	0.270311
14	T3	female	0.249154
15	T3	male	0.246365



## Two-way Analysis Of Variance (ANOVA)

- In two-way ANOVA both factors are taken into account when computing the F-statistic:

$$TSS = SS_{treatment} + SS_{sex} + SSE$$

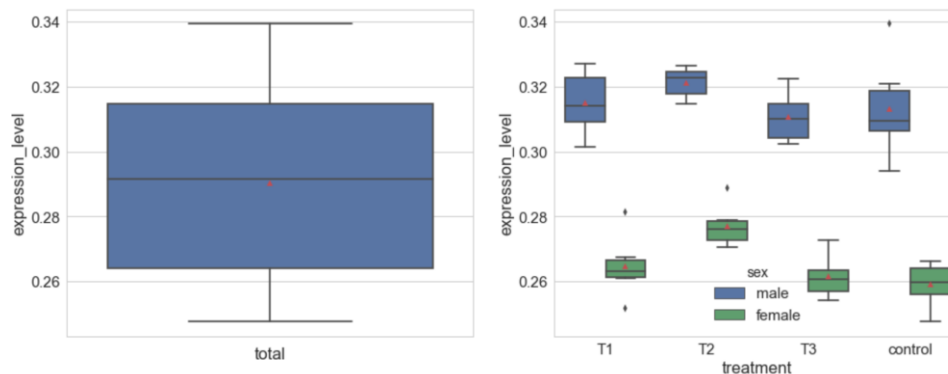
```
mod = ols('expression_level ~ treatment + sex',
          data=df).fit()
aov_table = sm.stats.anova_lm(mod, typ=2)
print aov_table
```

	sum_sq	df	F	PR(>F)
treatment	0.001384	3.0	5.816689	0.001986
sex	0.000003	1.0	0.043488	0.835794
Residual	0.003410	43.0	NaN	NaN

	treatment	sex	expression_level
0	control	female	0.276610
1	control	male	0.268073
2	control	female	0.256852
3	control	male	0.249141
4	T1	female	0.252675
5	T1	male	0.247875
6	T1	female	0.280871
7	T1	male	0.261644
8	T2	female	0.281502
9	T2	male	0.257326
10	T2	female	0.271810
11	T2	male	0.281779
12	T3	female	0.256650
13	T3	male	0.270311
14	T3	female	0.249154
15	T3	male	0.246365

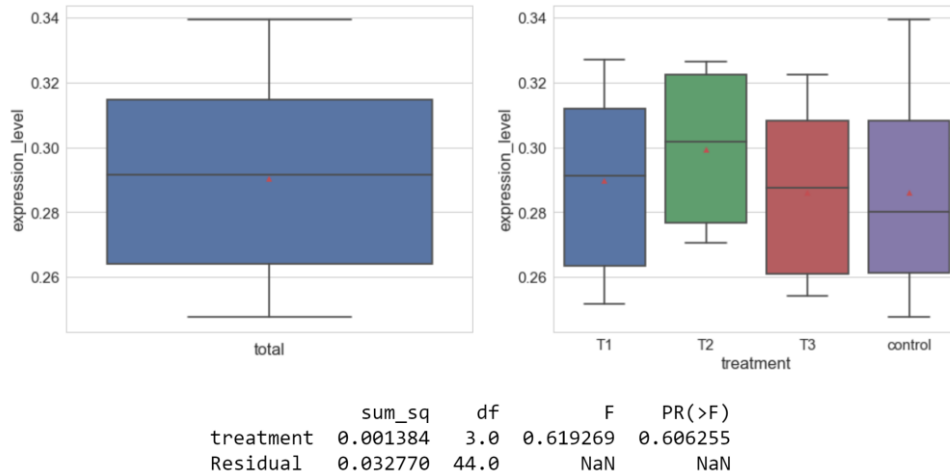
## Two-way Analysis Of Variance (ANOVA)

- A large effect of factor “sex” on the gene expression level is added:



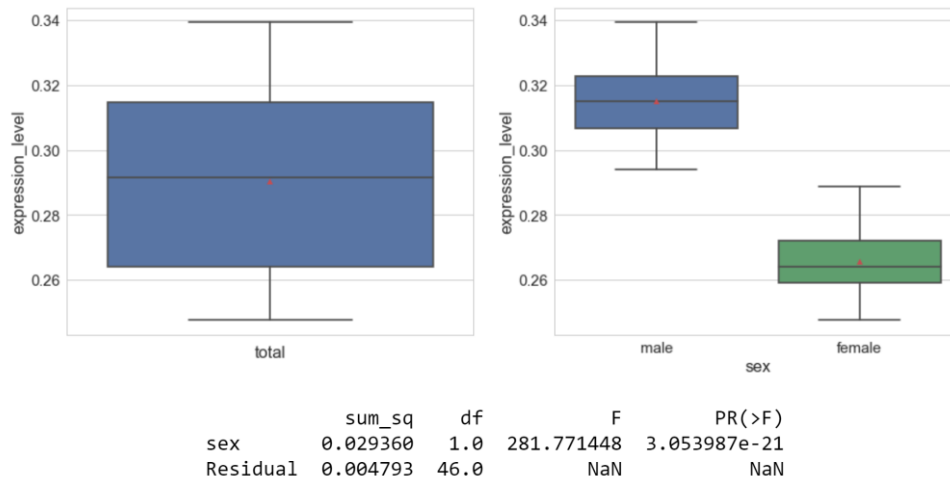
## Two-way Analysis Of Variance (ANOVA)

- A large effect of factor “sex” on the gene expression level is added:



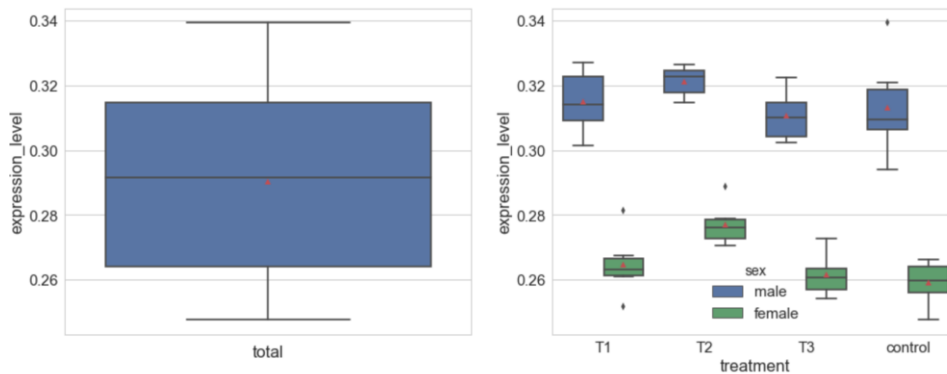
## Two-way Analysis Of Variance (ANOVA)

- A large effect of factor “sex” on the gene expression level is added:



## Two-way Analysis Of Variance (ANOVA)

- A large effect of factor “sex” on the gene expression level is added:

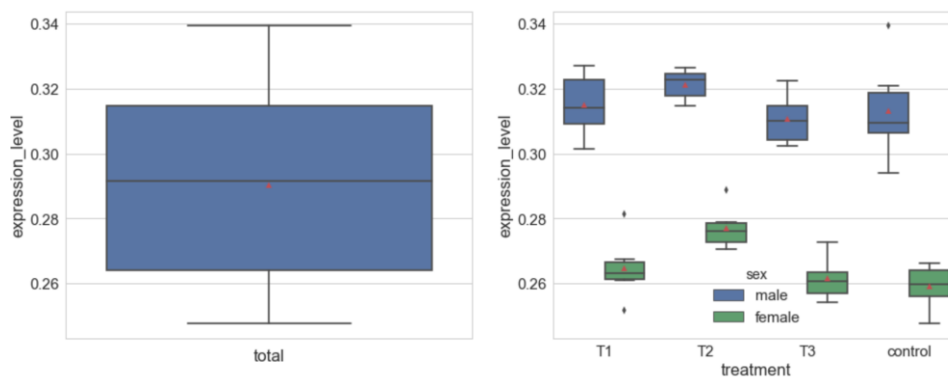


	sum_sq	df	F	PR(>F)
treatment	0.001384	3.0	5.816689	1.986050e-03
sex	0.029360	1.0	370.284849	9.454352e-23
Residual	0.003410	43.0	NaN	NaN

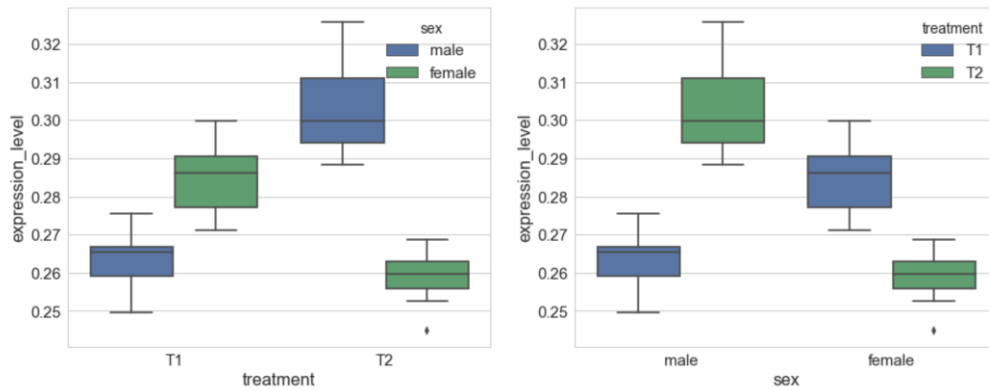
## Two-way Analysis Of Variance (ANOVA)

- The main difference with one-way ANOVA is the computation of the sum of squares of the error SSE which is computed in two-way ANOVA as

$$SSE = \sum_{l=1}^r \sum_{j=1}^{k_s} \sum_{i=1}^{k_t} (x_{ijl} - \bar{x}_{ij})^2.$$

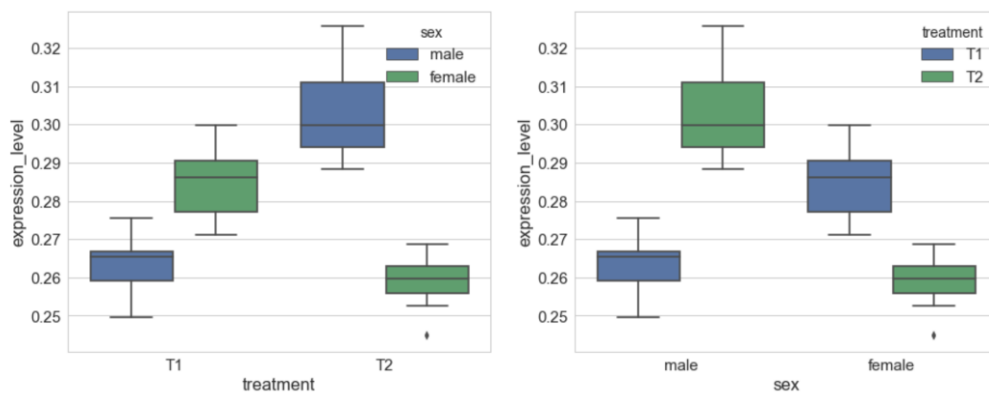


## Two-way Analysis Of Variance (ANOVA)

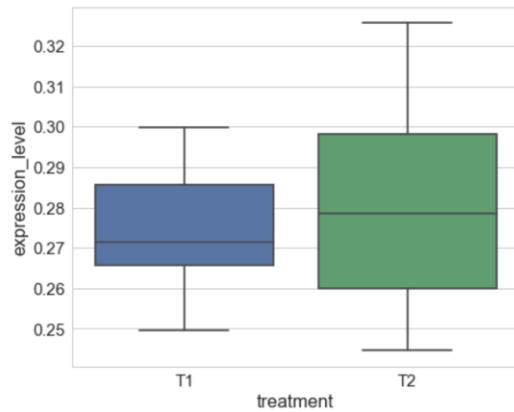


## Two-way Analysis Of Variance (ANOVA)

$$TSS = SS_{treatment} + SS_{sex} + SS_{treatment:sex} + SSE$$



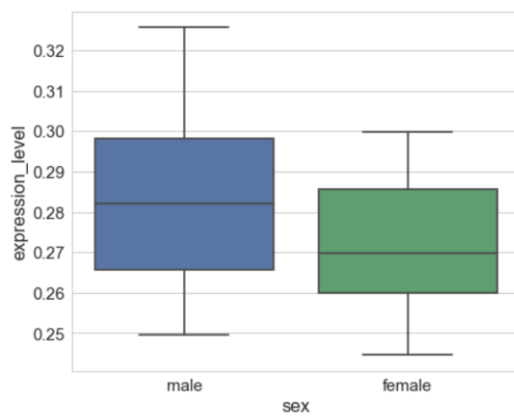
## Two-way Analysis Of Variance (ANOVA)



one-way ANOVA for treatment:

	sum_sq	df	F	PR(>F)
treatment	0.000575	1.0	1.37193	0.247512
Residual	0.019265	46.0	NaN	NaN

## Two-way Analysis Of Variance (ANOVA)

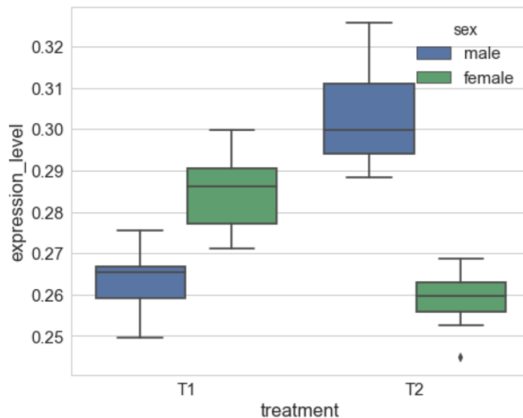


one-way ANOVA for sex:

	sum_sq	df	F	PR(>F)
sex	0.001612	1.0	4.068529	0.049547
Residual	0.018228	46.0	NaN	NaN

## Two-way Analysis Of Variance (ANOVA)

- If there is an interaction effect we should be very critical about the conclusions we make.



two-way ANOVA without interaction term:

	sum_sq	df	F	PR(>F)
treatment	0.000575	1.0	1.464674	0.232507
sex	0.001612	1.0	4.109628	0.048588
Residual	0.017653	45.0	NaN	NaN

two-way ANOVA with interaction term:

	sum_sq	df	F	PR(>F)
treatment	0.000575	1.0	6.116762	1.732354e-02
sex	0.001612	1.0	17.162605	1.532358e-04
treatment:sex	0.013520	1.0	143.928763	1.823620e-15
Residual	0.004133	44.0	NaN	NaN

## Power analysis

- The **power** of a hypothesis test is defined as the probability of rejecting the null hypothesis in case the alternative hypothesis is true.
- In power analysis the power is defined as  $1 - \beta$  where  $\beta$  is the probability of accepting the null hypothesis even though the alternative hypothesis is true.
- The **effect size**. It is the minimum size of the difference between the null hypothesis and the alternative hypothesis that we hope to detect.

## Power analysis: population variance

```
m = 1000
alpha = 0.01
n = 9
diff = 0.02
for sigma in [0.01,0.015,0.02,0.025,0.03,0.04,0.05]:
    power = 0.
    for i in range(m):
        geneA = np.random.normal(size=n,loc=0.26,scale=sigma)
        geneB = np.random.normal(size=n,loc=0.26+diff,scale=sigma)
        pvalue = stats.ttest_ind(geneA,geneB).pvalue
        if pvalue < alpha:
            power += 1.
    print "sample size: %i / effect size: %.2f / population std: %.3f / power: %.2f" % \
          (n,diff,sigma,power/m)
```

```
sample size: 9 / effect size: 0.02 / population std: 0.010 / power: 0.88
sample size: 9 / effect size: 0.02 / population std: 0.015 / power: 0.52
sample size: 9 / effect size: 0.02 / population std: 0.020 / power: 0.26
sample size: 9 / effect size: 0.02 / population std: 0.025 / power: 0.13
sample size: 9 / effect size: 0.02 / population std: 0.030 / power: 0.10
sample size: 9 / effect size: 0.02 / population std: 0.040 / power: 0.06
sample size: 9 / effect size: 0.02 / population std: 0.050 / power: 0.03
```

## Power analysis: sample size

```
sigma = 0.01
diff = 0.02
for n in range(2,15):
    power = 0.
    for i in range(m):
        geneA = np.random.normal(size=n,loc=0.26,scale=sigma)
        geneB = np.random.normal(size=n,loc=0.26+diff,scale=sigma)
        pvalue = stats.ttest_ind(geneA,geneB).pvalue
        if pvalue < alpha:
            power += 1.
    print "sample size: %i / effect size: %.2f / population std: %.3f / power: %.2f" % \
          (n,diff,sigma,power/m)
```

```
sample size: 2 / effect size: 0.02 / population std: 0.010 / power: 0.05
sample size: 3 / effect size: 0.02 / population std: 0.010 / power: 0.16
sample size: 4 / effect size: 0.02 / population std: 0.010 / power: 0.33
sample size: 5 / effect size: 0.02 / population std: 0.010 / power: 0.47
sample size: 6 / effect size: 0.02 / population std: 0.010 / power: 0.65
sample size: 7 / effect size: 0.02 / population std: 0.010 / power: 0.73
sample size: 8 / effect size: 0.02 / population std: 0.010 / power: 0.84
sample size: 9 / effect size: 0.02 / population std: 0.010 / power: 0.91
sample size: 10 / effect size: 0.02 / population std: 0.010 / power: 0.93
sample size: 11 / effect size: 0.02 / population std: 0.010 / power: 0.96
sample size: 12 / effect size: 0.02 / population std: 0.010 / power: 0.98
sample size: 13 / effect size: 0.02 / population std: 0.010 / power: 0.99
sample size: 14 / effect size: 0.02 / population std: 0.010 / power: 0.99
```

## Power analysis: effect size

```
sigma = 0.01
n= 9
for diff in [0.001,0.005,0.01,0.02,0.03,0.04]:
    power = 0.
    for i in range(m):
        geneA = np.random.normal(size=n,loc=0.26,scale=sigma)
        geneB = np.random.normal(size=n,loc=0.26+diff,scale=sigma)
        pvalue = stats.ttest_ind(geneA,geneB).pvalue
        if pvalue < alpha:
            power += 1.
    print "sample size: %i / effect size: %.2f / population std: %.3f / power: %.2f" % \
        (n,diff,sigma,power/m)
```

```
sample size: 9 / effect size: 0.00 / population std: 0.010 / power: 0.01
sample size: 9 / effect size: 0.01 / population std: 0.010 / power: 0.06
sample size: 9 / effect size: 0.01 / population std: 0.010 / power: 0.24
sample size: 9 / effect size: 0.02 / population std: 0.010 / power: 0.89
sample size: 9 / effect size: 0.03 / population std: 0.010 / power: 1.00
sample size: 9 / effect size: 0.04 / population std: 0.010 / power: 1.00
```

## Power analysis

- The power of a hypothesis test depends on these four numbers:
  - the effect size
  - the population variance
  - the sample size
  - the significance level
- Given the effect size, the population variance, the significance level and the expected minimum power we can compute the minimum samples size required to reach that power.



## Sample size calculations

- Sample size calculation for unpaired t-test:

Suppose you want to compare the mean in one group to the mean in another (i.e. carry out an unpaired t-test). The number,  $n$ , required in each group is given by

$$n = f(\alpha, \beta) \cdot \frac{2s^2}{\delta^2}$$

Where:

$\alpha$  is the significance level (using a two-sided test) — i.e. your cut-off for regarding the result as statistically significant.

$1 - \beta$  is the power of your test.

$f(\alpha, \beta)$  is a value calculated from  $\alpha$  and  $\beta$

$\delta$  is the smallest difference in means that you regard as being important to be able to detect.

$s$  is the standard deviation of whatever it is we're measuring — this will need to be estimated from previous studies.

$f(\alpha, \beta)$  for the most commonly used values for  $\alpha$  and  $\beta$

$\alpha$	$\beta$			
	0.05	0.1	0.2	0.5
0.05	13.0	10.5	7.9	3.8
0.01	17.8	14.9	11.7	6.6

## Power analysis

```
import statsmodels.stats.power as smp
sigma = 0.01
diff = 0.02
for power in [0.99,0.95,0.9,0.8]:
    print "Minimum sample size required for power = %f: %f" \
          % (power,smp.TTestIndPower().solve_power(diff/sigma, power=power, alpha=alpha, alternative='two-sided'))
```

Minimum sample size required for power = 0.990000: 13.805334

Minimum sample size required for power = 0.950000: 10.710971

Minimum sample size required for power = 0.900000: 9.251530

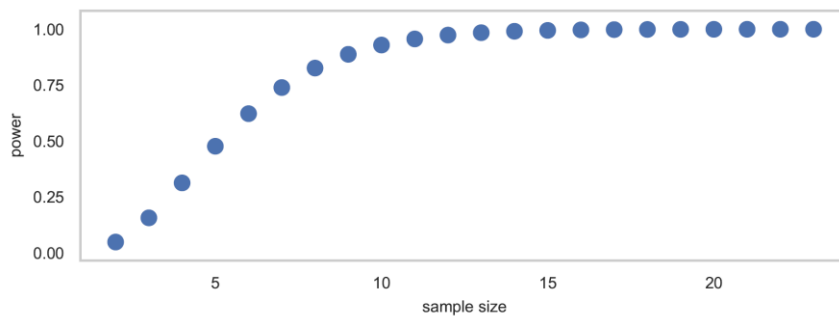
Minimum sample size required for power = 0.800000: 7.660540

## Power analysis

```
n = 11
power = 0.
for i in range(m):
    geneA = np.random.normal(size=n,loc=0.26,scale=0.01)
    geneB = np.random.normal(size=n,loc=0.26+diff,scale=0.01)
    pvalue = stats.ttest_ind(geneA,geneB).pvalue
    if pvalue < alpha:
        power += 1.
print "Power for sample size %i, mean difference %f and population sigma %f: %f" % \
(n,diff,sigma,power/(float(m)))
```

Power for sample size 11, mean difference 0.020000 and population sigma 0.010000: 0.956000

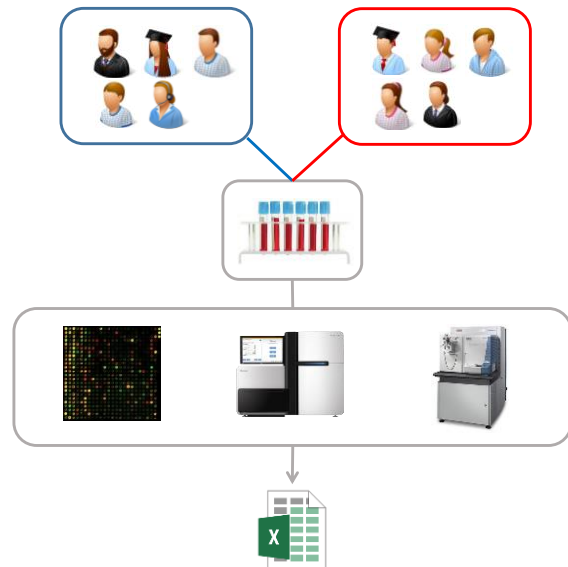
## Power analysis



## Statistics in biomedical research

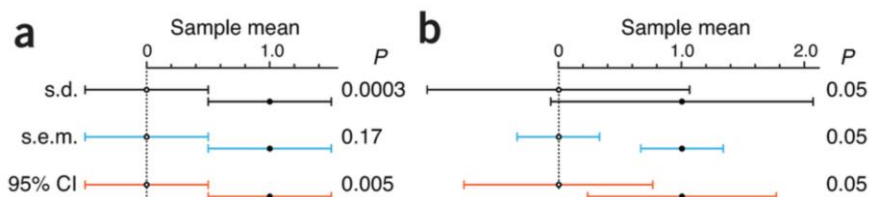
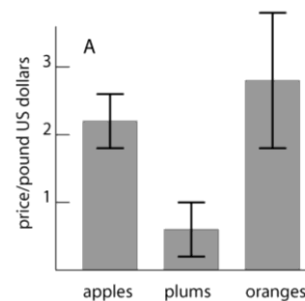
In many areas of biology, and especially in clinical research, the statistician

- is restricted to **small sample sizes**
  - difficult to find representative patients
  - time/money considerations
- is faced with enormous **biological variability**
  - many factors influence gene/protein expression
  - mouse models
- is faced with **technical variability**
  - sample preparation
  - machine measurement imprecisions
- is not able to control all relevant variables
- is interested in **small effects**



## Intermezzo: error bars

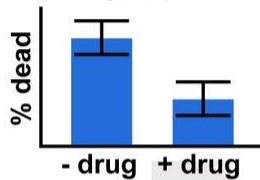
- The uncertainty in estimates is often represented using error bars.
- These can represent the standard deviation, the standard error of the mean or a confidence interval. This needs to be stated in the legend.
- Error bars don't necessarily relate to statistical significance. E.g. is the average price of apples different from the average price of oranges?
- Don't use error bars for small sample sizes (e.g. 3 data points).



## Statistical Significance Tests

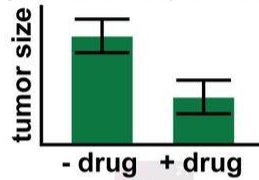
### Fisher's Exact Test

x-axis: 2 categories (- or + drug)  
y-axis: 2 categories (dead or alive)



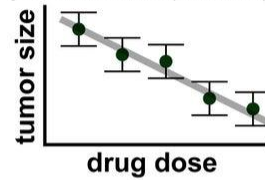
### Students T-Test

x-axis: 2 categories (- or + drug)  
y-axis: continuous (tumor size)



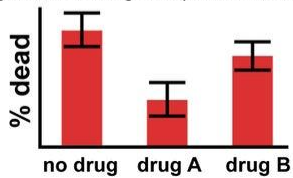
### Regression Analysis

x-axis: continuous (drug dose)  
y-axis: continuous (tumor size)



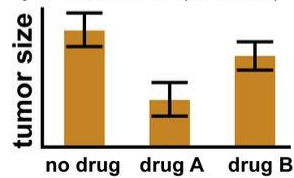
### Chi-Square Test

x-axis: >2 categories (drug A, B, etc.)  
y-axis: >2 categories (dead or alive)



### ANOVA

x-axis: >2 categories (drug A, B, etc.)  
y-axis: continuous (tumor size)



### Multiple Regression

x/z-axis: >2 continuous (drug dose)  
y-axis: continuous (tumor size)

