Statistical Analysis of Data

- Given a set of measurements of a value, how certain can we be of the value?
- Given a set of measurements of two values, how certain can we be that the values are different?
- Given a measured outcome and several condition or treatment values, how can we remove the effect of unwanted conditions or treatments on the outcome?

Measuring CPU Time

I made the 37 measurements of the CPU time required to compute

$$\binom{10000}{500}$$

in Common Lisp on `darwin.cs.orst.edu`.

0.27	0.25	0.23	0.24	0.26	0.24	0.26	0.25	0.24	0.25
0.25	0.24	0.25	0.24	0.25	0.26	0.24	0.25	0.25	0.25
0.25	0.25	0.24	0.25	0.24	0.25	0.25	0.24	0.25	0.25
0.24	0.25	0.24	0.24	0.25	0.25	0.26			

What is the true CPU cost of this computation?

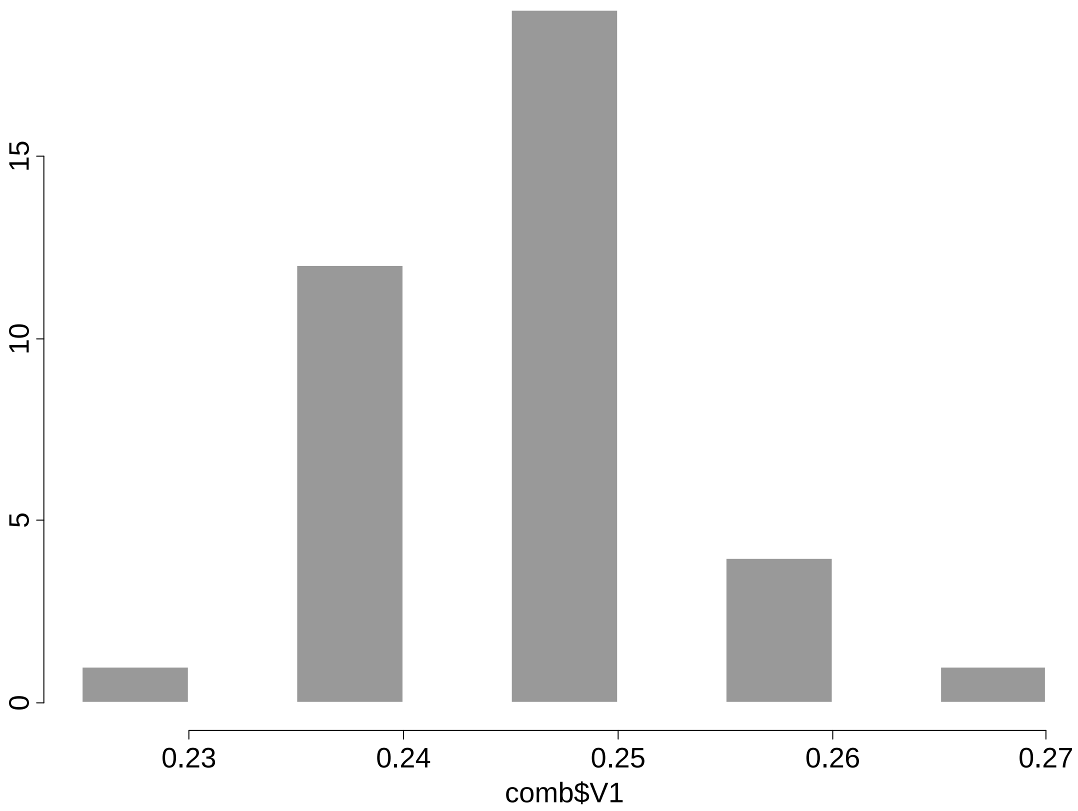
Before doing any calculations with the data,

Always Visualize Your Data

Histogram

Using Splus (on ada), we can issue the commands

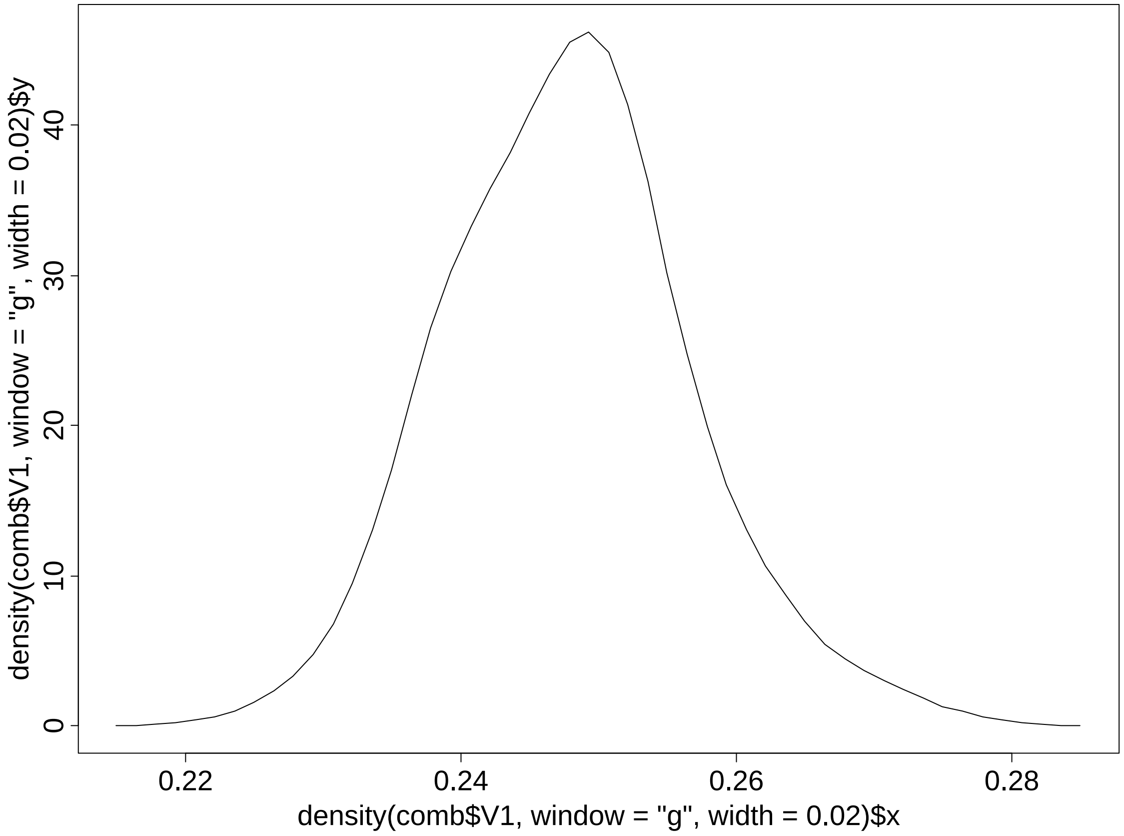
```
% Splus
> comb <- read.table("comb.data")
> hist(comb$V1)
```



Kernel Density Estimate

We can also construct a kernel density estimate. A kernel density estimate places a small normal distribution (the "kernel") at each observed data point and sums them up.

```
> plot(density(comb$V1, width=0.02),type="l")
```



Sample Mean

Based on this visualization, it is reasonable to compute the mean of this distribution:

$$\text{mean} = \overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Mean = 0.248

But how confident can we be that this is the true value? We would like to have a **confidence interval** that would tell us the following:

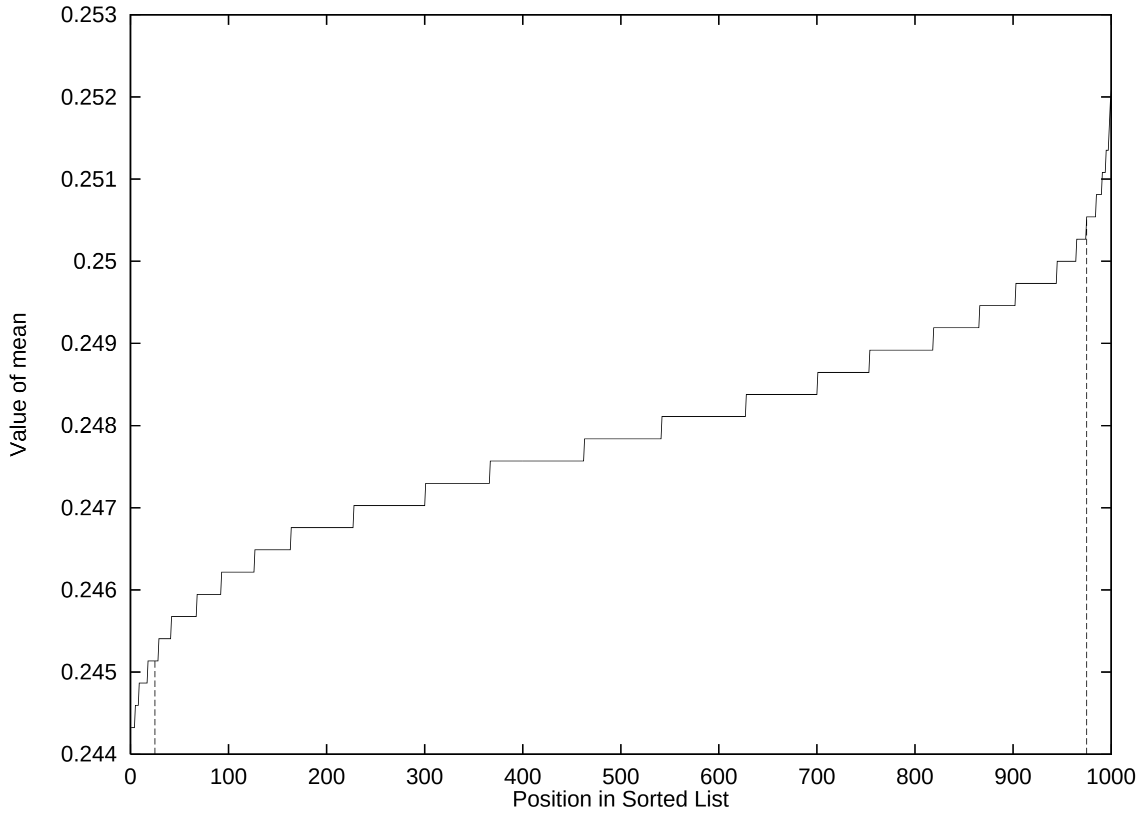
> If we drew random samples of size 37 and took the mean, 95% of the time, the mean would lie between a *lower bound* and an *upper bound*.

Confidence Intervals Via Resampling

Using a computer, we can simulate this. We draw 1000 random subsamples (with replacement) from our original 37 points and compute the mean. Then we sort these means and choose the 26th and 975th values as our lower and upper bounds.

Results: In 950 trials (out of 1000),

$$0.2451 \leq \overline{x} \leq 0.2505$$



Confidence Intervals Via Distributional Theory

If we plot a histogram of the 1000 bootstrap trials, we see that it is very nearly normally distributed. This is called the **sampling distribution of the mean**. The **Central Limit Theorem** says that the sampling distribution of the mean is normally distributed.

The normal distribution has two parameters,

- the *mean* (denoted μ)
- the *standard deviation* (denoted σ).

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

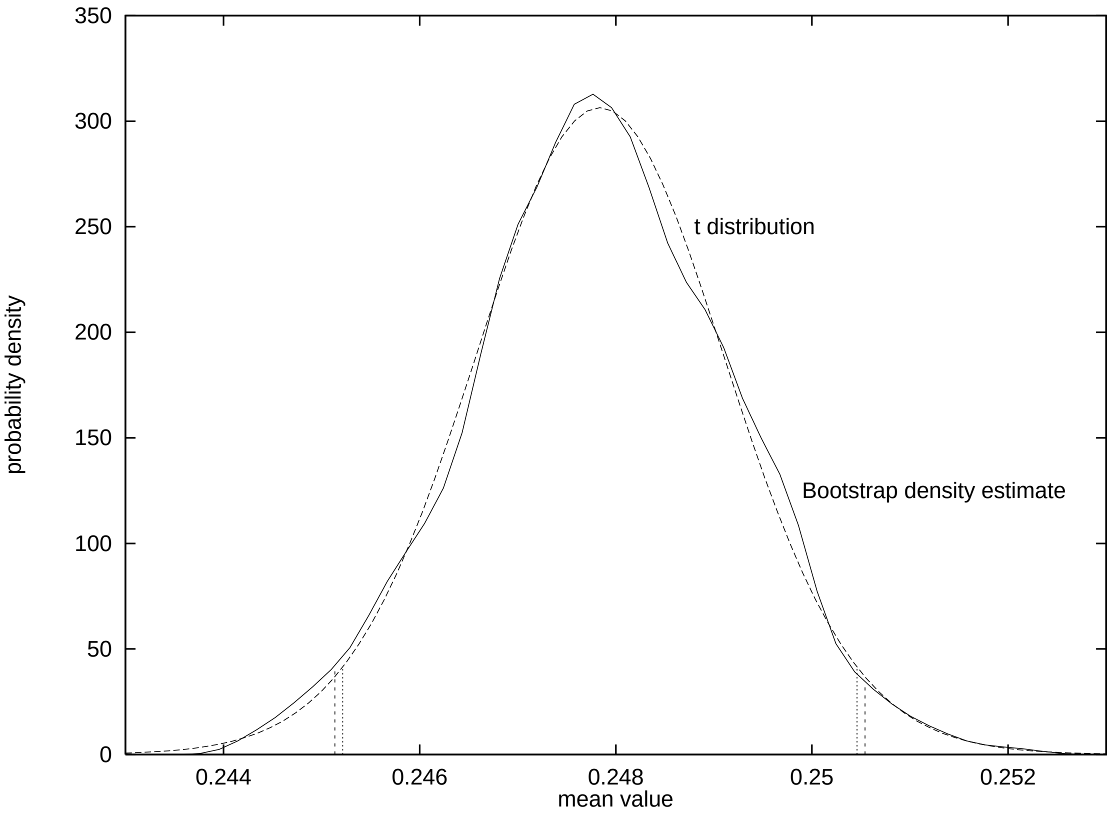
If the original CPU times were distributed with mean μ and standard deviation σ, then the *means* will be distributed with mean μ and standard deviation $\sigma/\sqrt{n}$. (Here $n = 37$.) Unfortunately, we must know the true standard deviation of the CPU times in order to apply the central limit theorem. We don't know this.

The Sample Standard Deviation

$$\text{standard deviation} = s = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n-1}}$$

The t distribution

Instead of the normal distribution, we can use the t distribution. The t distribution has three parameters: the mean (μ), the standard deviation (σ), and the degrees of freedom ($d.f. = n - 1$).



The 95% confidence limits are slightly tighter according to the central limit theorem using the t distribution.

Distributional confidence intervals

A 95% confidence interval for the mean can be computed via the t distribution as follows:

Let $\overline{x}$ be the sample mean.
Let s be the sample standard deviation.
Let n be the sample size.
Let $t_{0.025}(n-1)$ be the value of t with $n-1$ degrees of freedom such that the probability that $x < t_{0.025}(n-1)$ is 0.975.

Then,

$$\overline{x} - t_{0.025}(n-1)s/\sqrt{n} \leq \mu \leq \overline{x} + t_{0.025}(n-1)s/\sqrt{n}$$

Where μ is the true mean of the CPU times.

The t values can be looked up in a table, or you can use Splus:

```
> qt(0.975,36)
[1] 2.028094
```

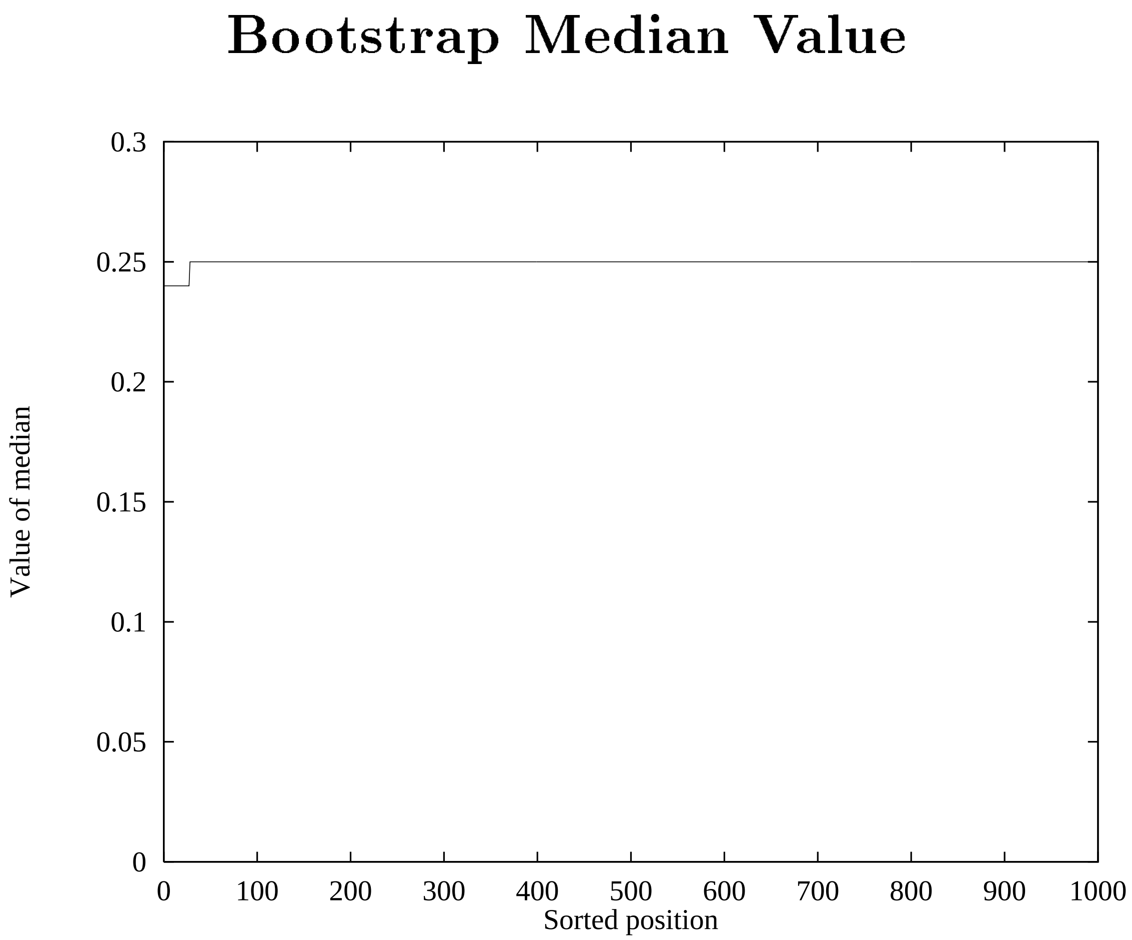
Bootstrapping on the Median

Suppose our goal was to measure the *median* CPU time required for this computation, rather than the average.

We would like to know that 95% of the time, the observed median is within some *bound* of the true median.

While distribution theory can't help us here, we can still apply the bootstrap method:

> Choose 1000 random samples (with replacement) of size 37 from our original 37 points. Take the median value of each sample. Sort and take the value at the 25th and 975th positions.


Bootstrap Median Value
Value of median
Sorted position
0
0.05
0.1
0.15
0.2
0.25
0.3
0
100
200
300
400
500
600
700
800
900
1000

When the Bootstrap Doesn't Work Well

The bootstrap is good for the mean, the median, and other statistics involving the "middle" of a distribution. The bootstrap is not good for estimating the minimum, the maximum, or other statistics involving the "tails" of the distribution.

Measuring Number of Occurrences of Events

In many CS experiments, we count the number of events that occur in n trials. For example, in machine learning, suppose we constructed a decision tree and then evaluated it on a test set of 100 examples and observed 88 correct classifications. We would report the *proportion of correctly classified* test examples as 0.88.

But how uncertain is this quantity? How much might it vary due to the random choice of the test set?

We will say $\hat{\theta} = 0.88$, where θ is the true proportion of correct classifications that our decision tree would make (on an infinite test set).

A Bootstrap Confidence Interval

We can again perform a bootstrapping experiment. Let n be number of test examples.

Repeat 1000 trials:

Draw a random sample of size n with replacement from the test set.

Measure p_i = the proportion correctly classified by the decision tree.

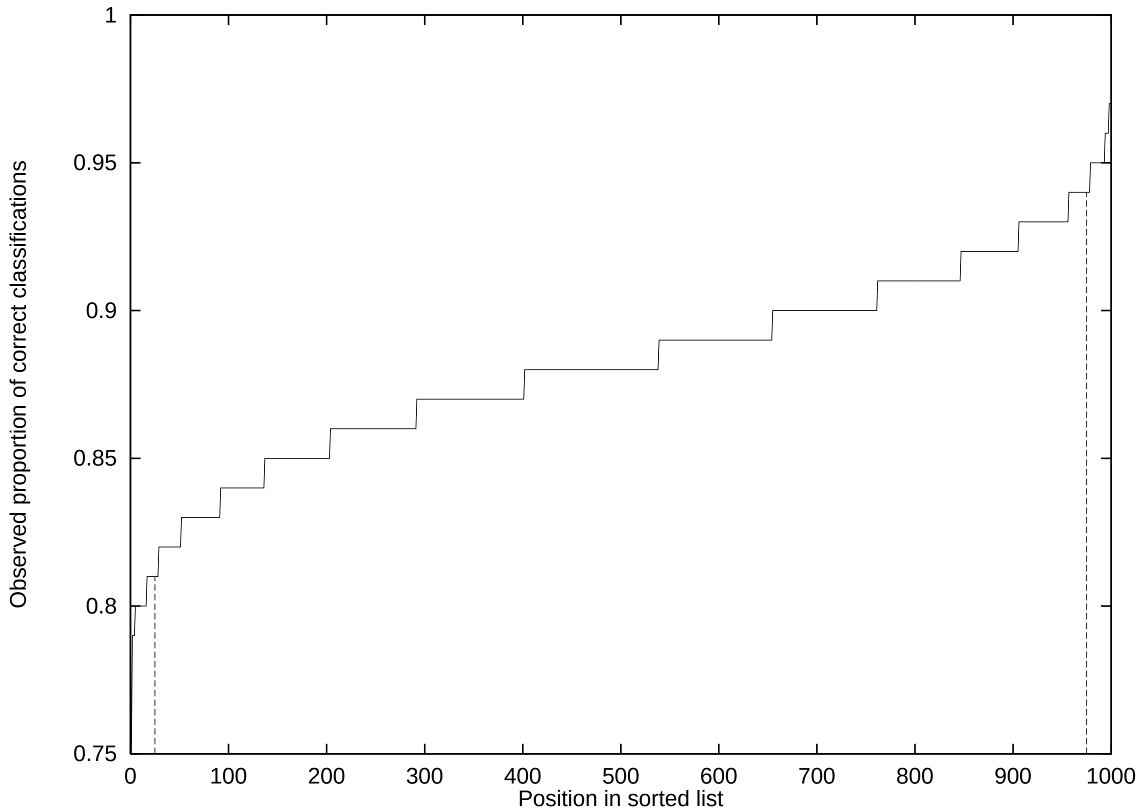
Sort the p_i in increasing order.

Choose lb and ub to be the 26th and 975th elements.

Then, we would say in 1000 trials, the probability is 0.95 that we would observe $lb \leq \hat{\theta} \leq ub$.

Results: $0.81 \leq \hat{\theta} \leq 0.94$ with confidence 0.95.

Bootstrap Graph



Binomial Confidence Interval From Distributional Theory

Suppose we have a biased coin with probability of heads θ. Suppose we take a sample of size n and measure the proportion of successes $\hat{\theta}$. From the central limit theorem, this quantity is approximately normally distributed with mean $\hat{\theta}$ and standard deviation $\sqrt{\hat{\theta}(1-\hat{\theta})/n}$.

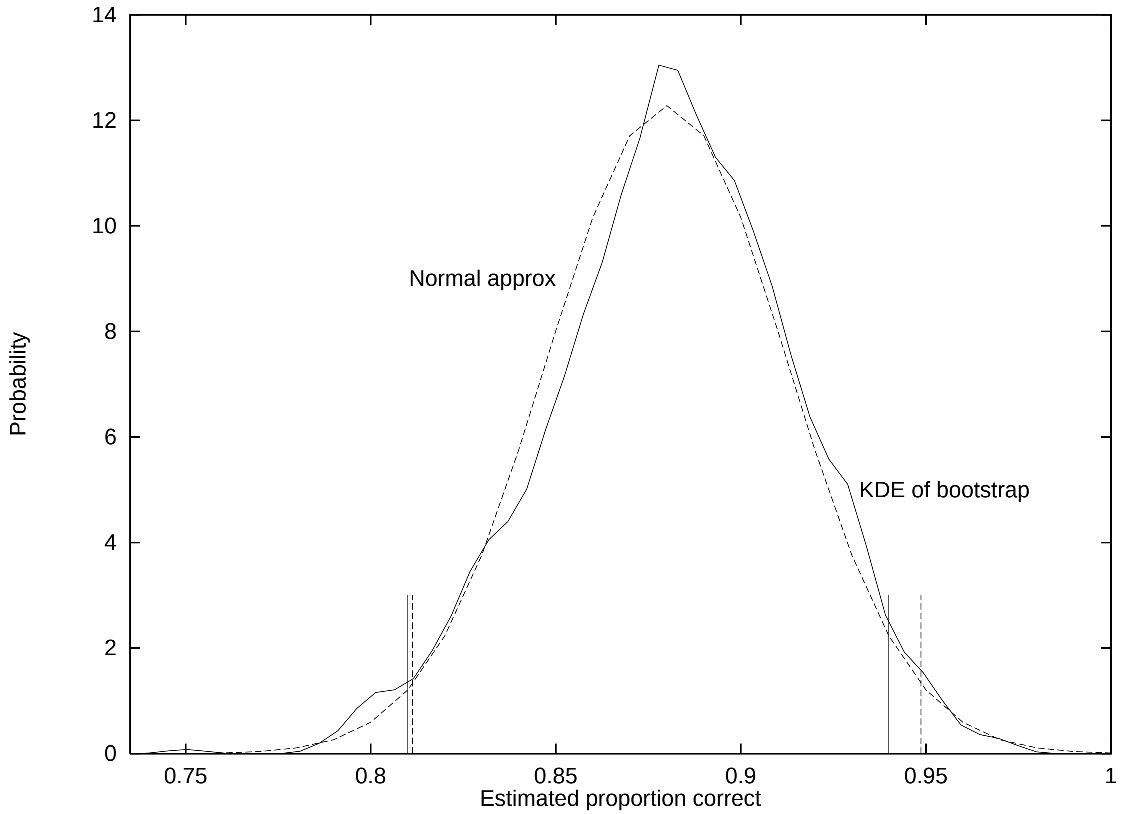
We can therefore use a 95% confidence interval for the mean of the normal distribution to compute a confidence interval for the binomial distribution. We make a slight change (called the "continuity correction") to correct for the discrete nature of the binomial distribution.

$$\hat{\theta} - \left[z_{0.975}\sqrt{\hat{\theta}(1-\hat{\theta})/n} + 1/(2n)\right] \leq \hat{\theta} \leq \hat{\theta} + \left[z_{0.975}\sqrt{\hat{\theta}(1-\hat{\theta})/n} + 1/(2n)\right]$$

Here $z_{0.975}$ is the value of a normally distributed variable z such that $P(z \leq z_{0.975}) = 0.975$. Specifically, $z_{0.975} = 1.96$.

Results: $0.811 \leq \hat{\theta} \leq 0.949$.

Bootstrap and Normal Distributions



The Normal approximation is always symmetrical, so it does not work very well when $\hat{\theta}$ is near 0.0 or 1.0.

Comparing Two Measurements

I performed 33 trials of

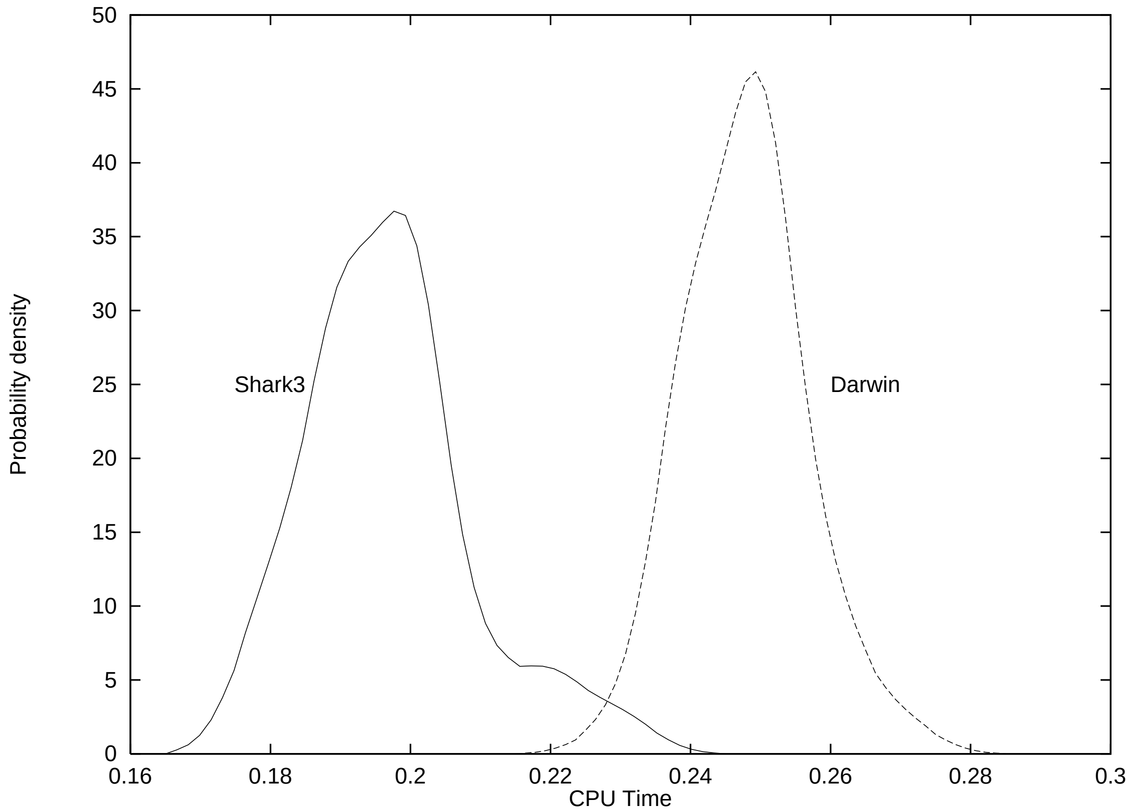
$$\binom{10000}{500}$$

in Common Lisp on `shark3.cs.orst.edu`.

0.21	0.20	0.20	0.19	0.20	0.19	0.18	0.20	0.19	0.19
0.19	0.19	0.20	0.18	0.19	0.20	0.22	0.20	0.20	0.20
0.19	0.20	0.18	0.19	0.19	0.20	0.20	0.22	0.18	0.19
0.21	0.23	0.20							

Can we conclude that shark3 is faster than darwin?

Visualizing



Comparable kernel density estimation plots. Visually, shark3 is much faster than darwin.

Bootstrap Test

Conduct 1000 trials of the following:

Draw bootstrap sample from Darwin, compute mean $\overline{x}_d$

Draw bootstrap sample from Shark3, compute mean $\overline{x}_s$

Count number of times $\overline{x}_d > \overline{x}_s$.

If this is greater than 950, then we can be 95% confident.

Result: All 1000 trials give darwin slower than shark3.

We can also compute a 95% bootstrap confidence interval on the difference $\overline{x}_d - \overline{x}_s$:

$$0.0461 \leq \overline{x}_d - \overline{x}_s \leq 0.0553.$$

Distributional Test

If two random variables are normally distributed, then their difference is normally distributed with mean $\mu = \mu_1 - \mu_2$ and standard deviation $\sigma = \sqrt{\sigma_1^2 + \sigma_2^2}$.

Now the sampling distribution of the mean $\overline{x}$ is approximately normally distributed (according to the central limit theorem). So we know $\overline{x}_1 - \overline{x}_2$ is also normally distributed. However, because we don't know σ_1 or σ_2, we must use the t distribution instead.

If the two samples have sizes n_1 and n_2, then $\overline{x}_1 - \overline{x}_2$ is has a t distribution with mean $\overline{x}_1 - \overline{x}_2$ and standard deviation

$$s = \sqrt{\left(\frac{\Sigma_{i=1}^{n_1}(x_{1,i} - \overline{x}_1)^2}{n_1 - 1} + \frac{\Sigma_{i=1}^{n_2}(x_{2,i} - \overline{x}_2)^2}{n_2 - 1}\right)\left(\frac{1}{n_1} + \frac{1}{n_2}\right)}$$

and $n_1 + n_2 - 2$ degrees of freedom.

Using the data above, we obtain

$\overline{x}_1 - \overline{x}_2 = 0.0509$

$s = 0.0023$

$df = 68$.

A 95% confidence interval for the difference is (0.0463,0.0555).

Hypothesis Testing

Suppose we want to know whether the true difference between the two machines is zero or non-zero. We can formalize this as a statistical decision:

H_0: $\mu_1 - \mu_2 = 0$
H_1: $\mu_1 - \mu_2 \neq 0$

H_0 is the "null hypothesis" and H_1 is the "alternative hypothesis". An hypothesis test determines probablistically whether we can reject H_0 in favor of H_1 by asking "Suppose H_0 is true, what is the probability that we would have observed the given data?"

Specifically, we want to know what is the probability that we would observe $\overline{x}_1 - \overline{x}_2 \geq 0.0509$ when the true difference was zero. This can be determined from the t distribution.

The computed value of t is

$$t = \frac{\overline{x}_1 - \overline{x}_2}{s} = 21.69$$

The probability of seeing a t value greater than or equal to this is virtually 0.0. $t_{0.99999}(68) = 4.59$.

Paired Differences

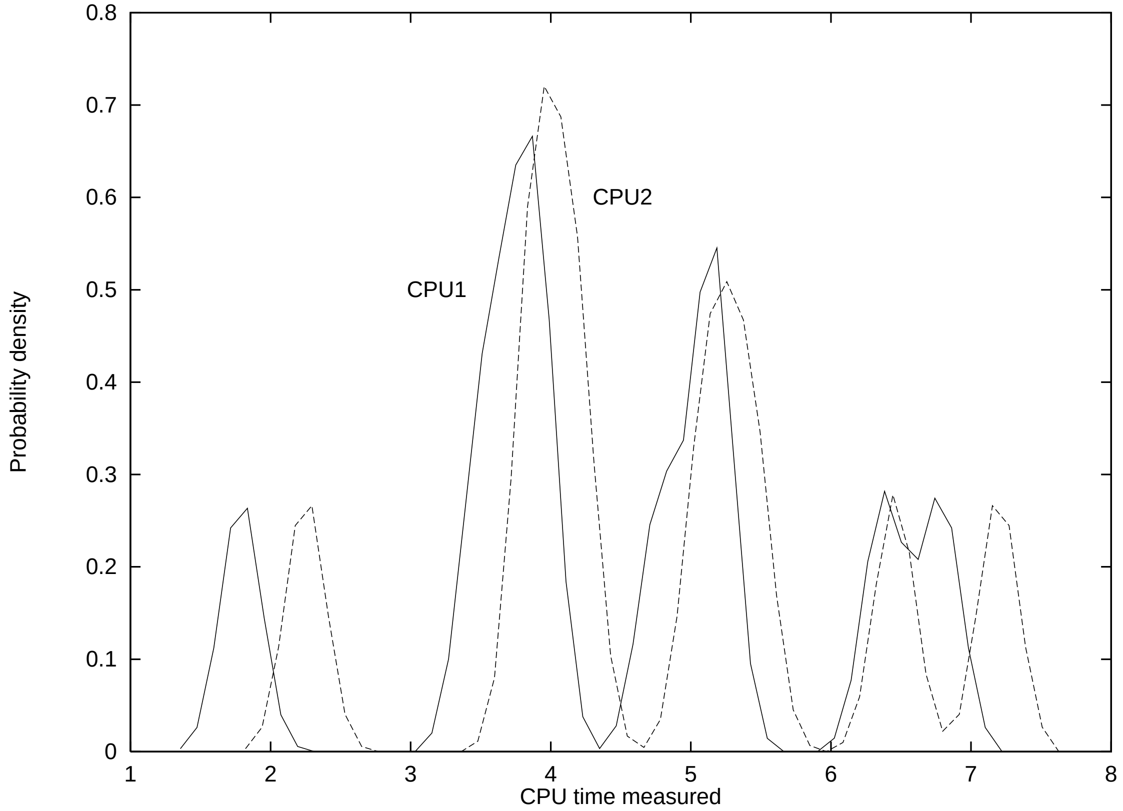
Suppose we had a set of benchmark programs that we were going to run on two machines. We will run each program on each machine to obtain the following data:

Program	CPU1	CPU2
P1	3.482514	3.896850
P2	3.677492	3.866780
P3	3.877525	4.206775
P4	6.787100	7.197257
P5	1.789549	2.250253
P6	5.156133	5.457694
P7	4.777698	5.075136
P8	3.906618	4.095468
P9	6.374434	6.456649
P10	5.152357	5.257691

Notice that the different programs have very different run times (e.g., ranging from 1.78 to 6.79 on CPU1).

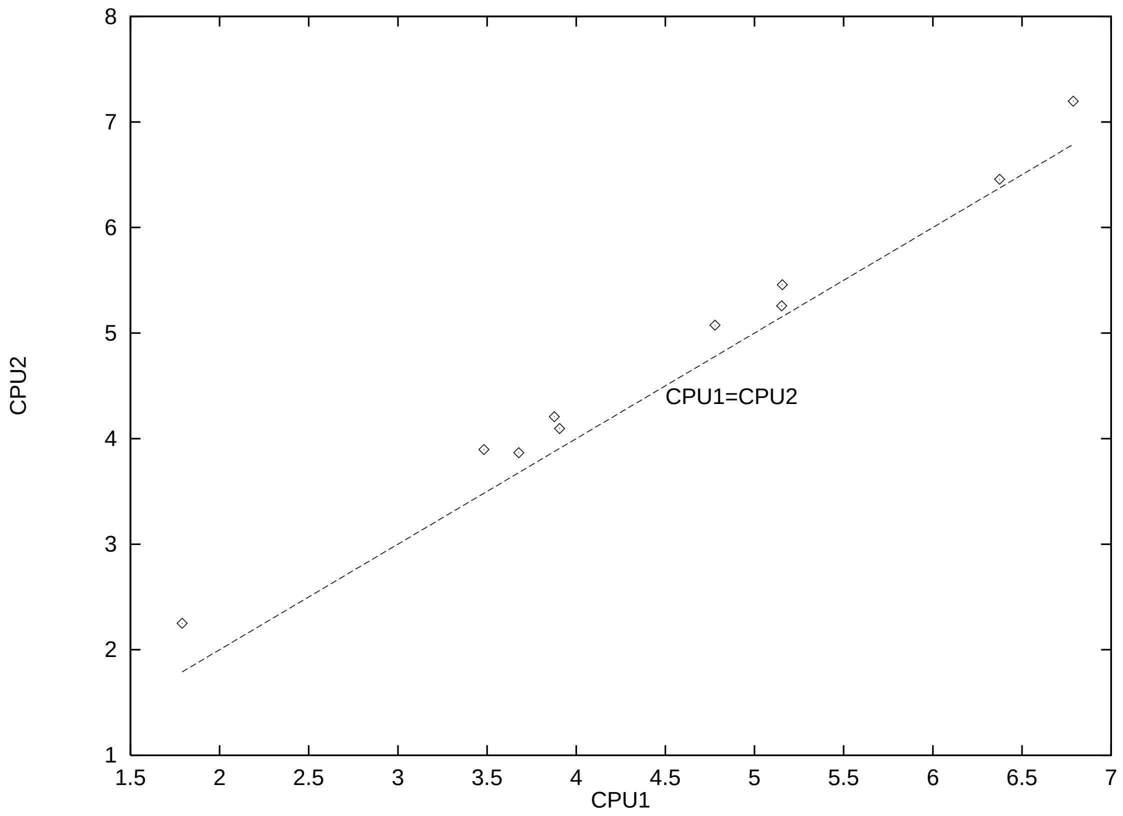
Visualization

There are many ways to visualize the data. We can superimpose a kernel density estimate for each of the CPU's:



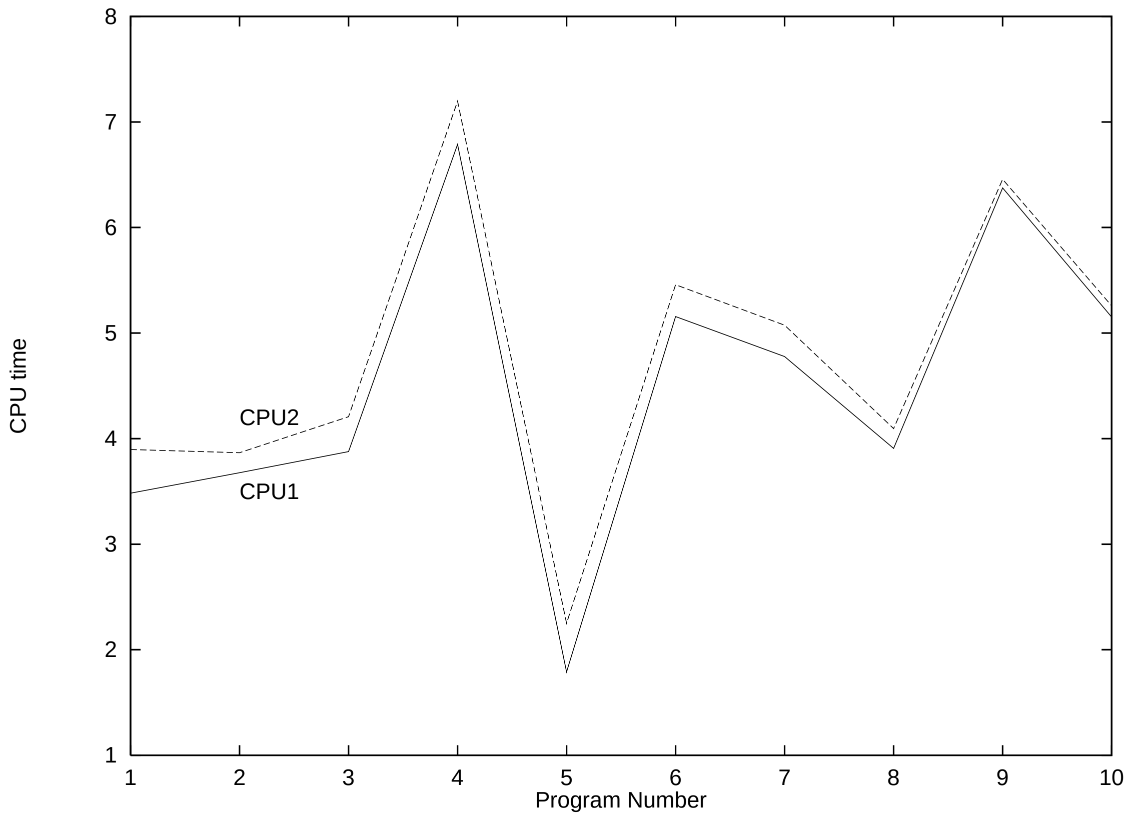
This suggests that CPU1 is systematically offset from CPU2.

Visualization (2)



This plots CPU1 versus CPU2 and also plots the line $y = x$. Notice that the performance of CPU1 is correlated with the performance of CPU2. Notice that all points lie above the line, suggesting that CPU2 is always bigger than CPU1.

Visualization (3)



Here we have plotted the data in sequential order (by program). We can see even more strongly that the CPU times of the programs co-vary.

Analysis of Paired Data

Construct points by subtracting $CPU1_i - CPU2_i$, and analyze this just like the univariate data we analyzed last time.

mean = 0.2779

standard deviation = 0.1321

degrees of freedom = 9

value of $t = 6.6549$

The probability of seeing this value (or greater) if the true mean were 0 is 0.0000466, so we can reject the null hypothesis that the mean is zero in favor of the alternative hypothesis that the mean is greater than zero with confidence at least 0.9999534.

However, in the absence of prior expectation that CPU2 is slower than CPU1, we should use a "two-tailed test". To do this, we must compute the probability that we would have seen a value $t \geq 6.6549$ or $t \leq -6.6549$. Because the distribution is symmetric, this probability is 0.0000932, so we can reject the null hypothesis in favor of the hypothesis that the mean is non-zero with confidence 0.9999068.

Analysis as Unpaired Data

If we had used our previous technique for unpaired data, we would not be able to detect the difference between the two CPU's.

$\overline{x}_1 - \overline{x}_2 = 0.2779$

$s = 0.6473$

$df = 18$

$t = 0.4293$

The probability of observing this t value (or greater) if the true difference is zero is 0.3364 (for a one-tailed test). For a 2-tailed test, it is 0.6728. So we cannot reject the null hypothesis using this analysis.

Bootstrap Analysis

1000-fold Bootstrap 95% Confidence Interval for $\overline{CPU2 - CPU1}$:
$0.2086 \leq \overline{(CPU2 - CPU1} \leq 0.3580$.

1000-fold Bootstrap 95% Confidence Interval for $\overline{CPU2} - \overline{CPU1}$:
$-1.4533 \leq \overline{CPU2} - \overline{CPU1} \leq 1.0277$. (This contains 0, so we cannot reject the null hypothesis.)

Differences in Proportions

In Machine Learning, we often need to test differences of two proportions. For example, in the ID3 vs. Backprop comparison paper, we compared ID3 and Backprop on the same test set to see which algorithm was better. This is a case of paired differences.

To perform the paired differences test, we need the 2x2 table:

		Backpropagation: Correct	Backpropagation: Incorrect	
ID3	Correct	4239 (58.5%)	512 (7.1%)	Disagree: 1385 (19.2%)
ID3	Incorrect	873 (12.1%)	1618 (22.3%)	Agree: 5857 (80.9%)

We will call the cells in this table $cell_{11}$, $cell_{10}$, $cell_{01}$, and $cell_{00}$.

Bootstrap Confidence Interval on Difference of Two Proportions

To construct one bootstrap sample,

let $n = 4239 + 512 + 873 + 1618$

let $cell_{00} = cell_{01} = cell_{10} = cell_{11} = 0$

Repeat n times:

Draw a random number r between 0 and $n - 1$ and

If $r < 4239$ then $cell_{11} += 1$

else if $r < 4239 + 512$ then $cell_{10} += 1$

else if $r < 4239 + 512 + 873$ then $cell_{01} += 1$

else $cell_{00} += 1$

Difference in proportions on this trial is $(cell_{10} - cell_{01})/n$

Repeat this 1000 times, sort them in order, and construct a confidence interval from the 26th and 975th elements in the list.

Results for NETtalk: $-0.0594 \leq p_{ID3} - p_{BP} \leq -0.0398$.

Bootstrap For Difference of Non-Paired Proportions

Suppose we have a sample of size n_1 with p_1 proportion of successes and a sample of size n_2 with p_2 proportion of successes.

Repeat 1000 times:

  Flip n_1 random coins with probability of success p_1

    let $p_{1,i}$ be the observed probability of success.

  Flip n_2 random coins with probability of success p_2

    let $p_{2,i}$ be the observed probability of success.

  Let $\hat{p}_i = p_{1,i} - p_{2,i}$

Sort the list of $\hat{p}_i$ values and choose the 26th and 975th elements.

If we treat the NETtalk data as un-paired, this test gives a confidence interval of $-0.0645 \leq p_{ID3} - p_{BP} \leq -0.0338$. This is much wider than the paired-differences interval.

Distributional Tests of the Paired Difference of Two Proportions

For paired differences, the quantity

$$\chi^2 = \frac{(|cell_{10} - cell_{01}| - 1)^2}{cell_{10} + cell_{01}}$$

is distributed according to the χ^2 distribution with 1 degree of freedom. 93.5747

For NETtalk, this value is 93.57. The probability of seeing a value at least that large under the null hypothesis (that the two proportions are identical) is < 0.0001. So we can reject the null hypothesis with confidence at least 0.9999.

Distributional Confidence Interval for the Paired Difference of Two Proportions

To construct a 95% confidence interval for the difference of two proportions, we can work as follows:

Let $p_{10} = cell_{10}/n$ and $p_{01} = cell_{01}/n$.
Let $SE = \sqrt{[p_{10} + p_{01} - (p_{10} - p_{01})^2]/n}$
Let $p_{a-b} = p_a - p_b = (cell_{10} - cell_{01})/n$
Then

$$p_{a-b} - [1.96SE + 1/(2n)] \leq p_{a-b} \leq p_{a-b} + [1.96SE + 1/(2n)].$$

For the NETtalk example: $-0.0599 \leq p_{ID3-BP} \leq -0.0398$.

This interval is tighter than the bootstrap interval, but it is based on the Central Limit Theorem.

Tests for Unpaired Differences of Two Proportions

Let p_1 be the proportion of successes in n_1 trials.
Let p_2 be the proportion of successes in n_2 trials.
Let $p = (n_2 p_1 + n_1 p_2)/(n_1 + n_2)$ be the pooled proportion of successes.
Then

$$z = \frac{p_1 - p_2}{\sqrt{p(1-p)(1/n_1 + 1/n + 2)}}$$

is approximately standard normally distributed.

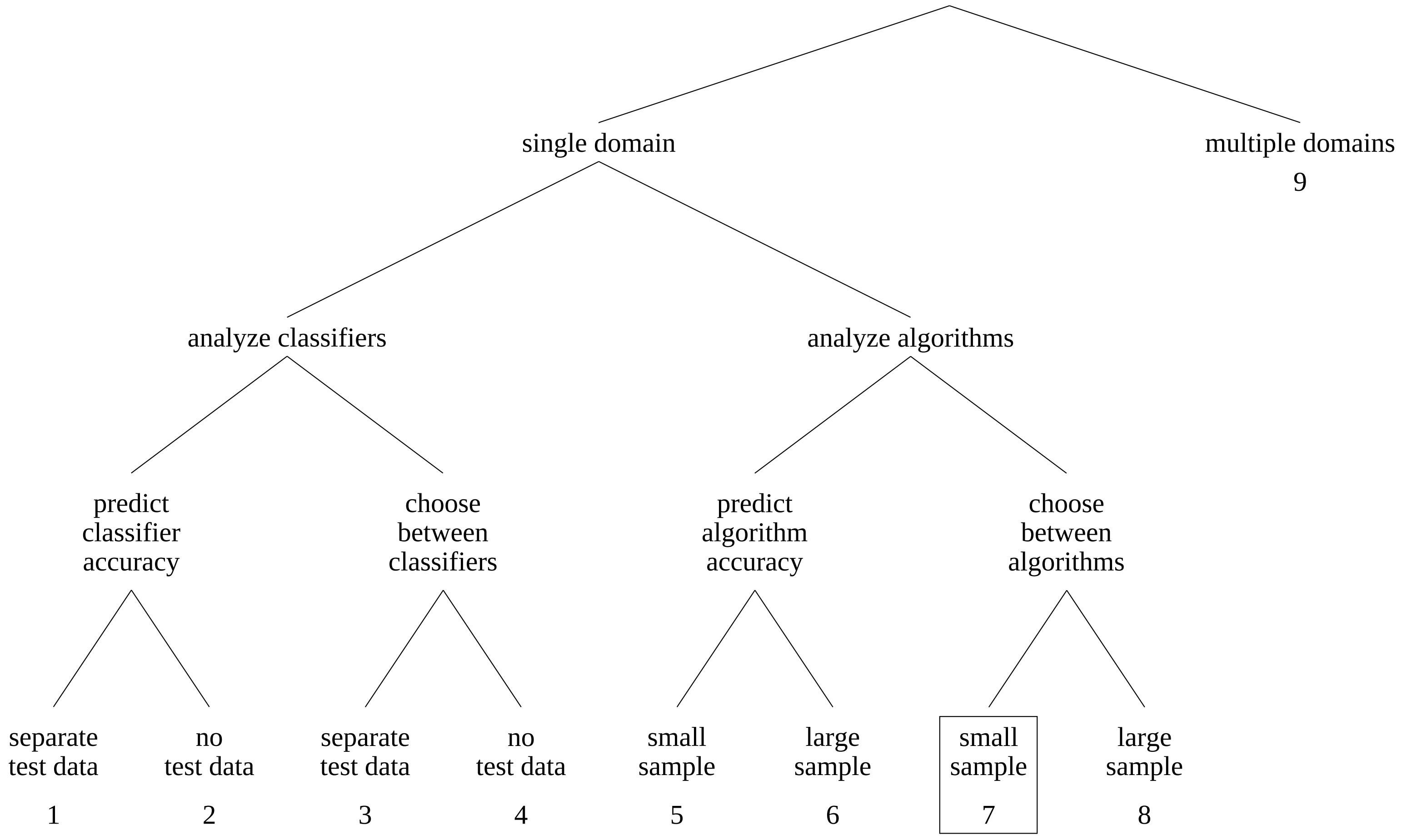
We can obtain a confidence interval for the difference in the two proportions by letting $SE = \sqrt{p_1(1-p_1)/n_1 + p_2(1-p_2)/n_2}$ and computing

$$p_1 - p_2 - [1.96SE + 1/(n_1+n_2)] \leq p_1 - p_2 \leq p_1 - p_2 + [1.96SE + 1/(n_1+n_2)]$$

For NETtalk, this gives $-0.0651 \leq p_{ID3} - p_{BP} \leq -0.0346$

There is a method known as Fisher's Exact Test that gives better estimates for small samples ($n_1 + n_2 < 20$ or $20 \leq n_1 + n_2 \leq 40$ and smallest expected cell has less than 5 examples in it).

Evaluation and Comparison of Learning Algorithms



Special Problems with Learning Algorithms

There are multiple sources of variation in learning algorithms:

- **Test set choice.** Randomly-chosen test sets may be unrepresentative.
- **Training set choice.** Randomly-chosen training sets may be unrepresentative.
- **Algorithm instability.** The classifier produced by an algorithm can vary substantially for even minor changes in the training set. Algorithms may also have internal sources of randomness (e.g., random initial weights for neural network algorithms).

Estimating Performance for Small Samples: 10-Fold Cross-Validation

Protocol:

- **Split** S **into** $n = 10$ **disjoint subsets** $S_1, \ldots, S_{10}$**.**
- **Repeat 10 times:**
  - **Let** $S_{train}^{(i)} = S - S_i$ **and** $S_{test}^{(i)} = S_i$**.**
  - **Run the algorithm**
    $C_A^{(i)} = A(S_{train}^{(i)})$
  - **Apply classifer to test set** $S_{test}^{(i)}$ **and count number of errors** $err^{(i)}$**.**
- **Compute statistic**

$$\text{error rate} = \frac{1}{|S|} \sum_i err^{(i)}$$

  We can compute a binomial confidence interval for this number in the usual way.

To stabilize this estimate, it can be repeated many times and averaged.

Comparing Two Learning Algorithms: The 5x2cv test

Protocol:

- **Repeat for $j = 1, \ldots, 5$:**
  - **Split S into $n = 2$ disjoint subsets S_1 and S_2.**
  - **Train each algorithm on each subset:**
    $C_A^{(1)} = A(S_1) \quad C_A^{(2)} = A(S_2)$
    $C_B^{(1)} = B(S_1) \quad C_B^{(2)} = B(S_2)$
  - **Test each classifier on the other subset and compute differences:**
    $\delta_j^{(1)} = \epsilon_A^{(1)} - \epsilon_B^{(1)}$.
    $\delta_j^{(2)} = \epsilon_A^{(2)} - \epsilon_B^{(2)}$.
  - **Compute the statistic**
    $s_j^2 = (\delta_j^{(1)} - \overline{\delta}_j)^2 + (\delta_j^{(2)} - \overline{\delta}_j)^2$.
- **Finally, compute**

$$t = \frac{\delta_1^{(1)}}{\sqrt{\frac{1}{5}\sum_{j=1}^{5} s_j^2}}$$

- **t has 5 degrees of freedom.**

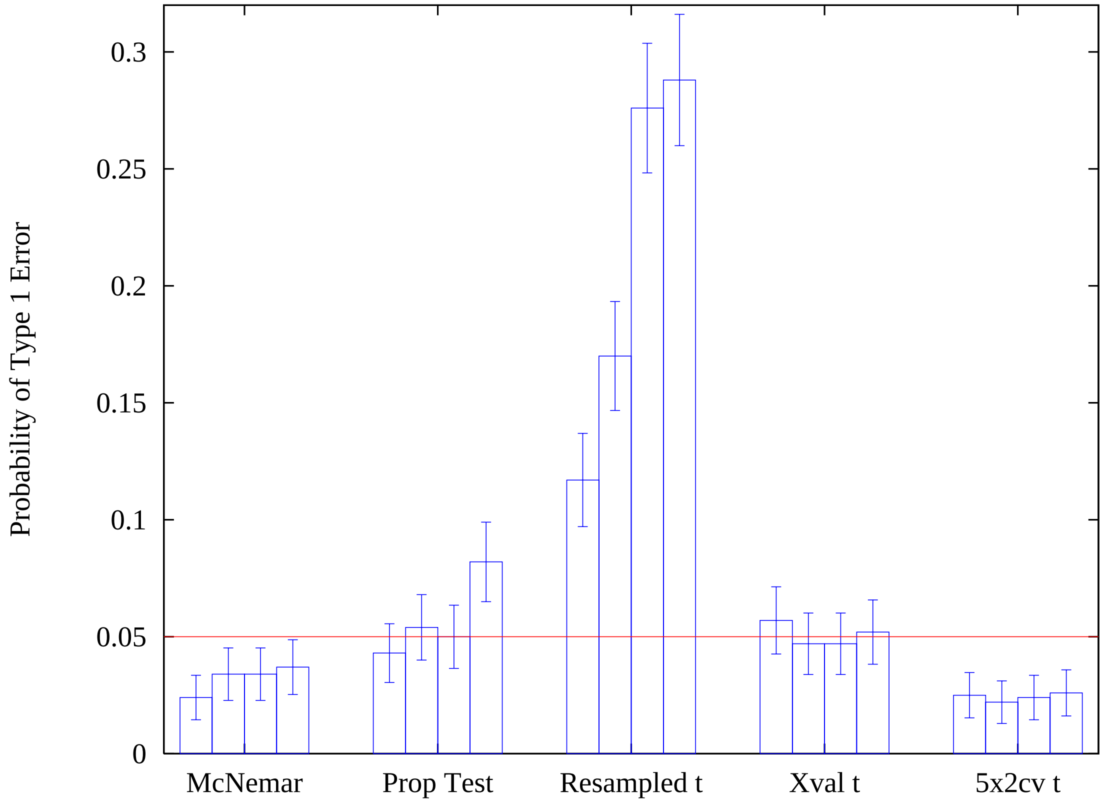
A Method to Avoid: The resampled t test

Protocol:

- **Repeat $n = 30$ times:**
  - **Subdivide S into 2 sets: $S^{(i)}_{train}$ and $S^{(i)}_{test}$.**
  - **Run the algorithms**
    $C^{(i)}_A = A(S^{(i)}_{train})$
    $C^{(i)}_B = B(S^{(i)}_{train})$
  - **Apply the classifiers to the test set, $S^{(i)}_{test}$, and measure the difference in error rates $\delta^{(i)} = \epsilon^{(i)}_A - \epsilon^{(i)}_B$.**
- **Compute statistic**

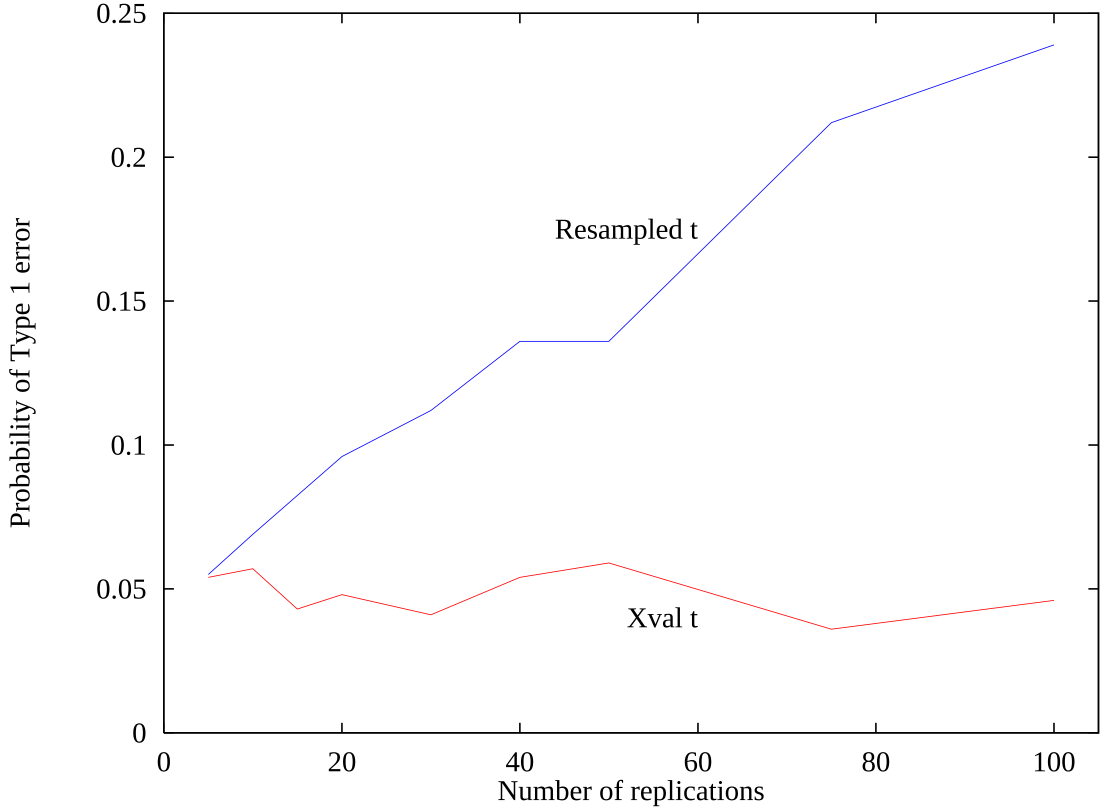
$$t = \frac{\overline{\delta}}{\sqrt{\frac{\sum_{i=1}^{n}\left(\delta^{(i)} - \overline{\delta}\right)^2}{n(n-1)}}}$$

Measurement of Type I Error



- **Resampled t test and difference-of-proportions test have unacceptable Type 1 errors.**

Achieving Arbitrary Levels of Significance



Regression Analysis of CPU Comparison

In our analysis of benchmark data across two CPU's, we assumed that there was a fixed, additive difference between the CPU's and we computed $\overline{x}_{CPU1} - \overline{x}_{CPU2}$. But usually, we assume that a computer program contains a certain amount of work (let's call it x_i) and a CPU has a certain speed. Hence, the following two linear models are more reasonable:

$$y_i^{CPU1} = k^{CPU1} x_i + \epsilon_1$$

$$y_i^{CPU2} = k^{CPU2} x_i + \epsilon_2$$

Where k^{CPU1} is the speed of CPU1 and k^{CPU2} is the speed of CPU2. ϵ_1 and ϵ_2 are random noise terms.

To compare two CPU's, it is convenient to take ratios, as follows:

$$\frac{y_i^{CPU2} - \epsilon_1}{y_i^{CPU1} - \epsilon_2} = \frac{k^{CPU2}}{k^{CPU1}}$$

Regrouping, we get

$$y_i^{CPU2} = \frac{k^{CPU2}}{k^{CPU1}} y_2 + \epsilon$$

where ϵ is a combined error term.

Regression Analysis

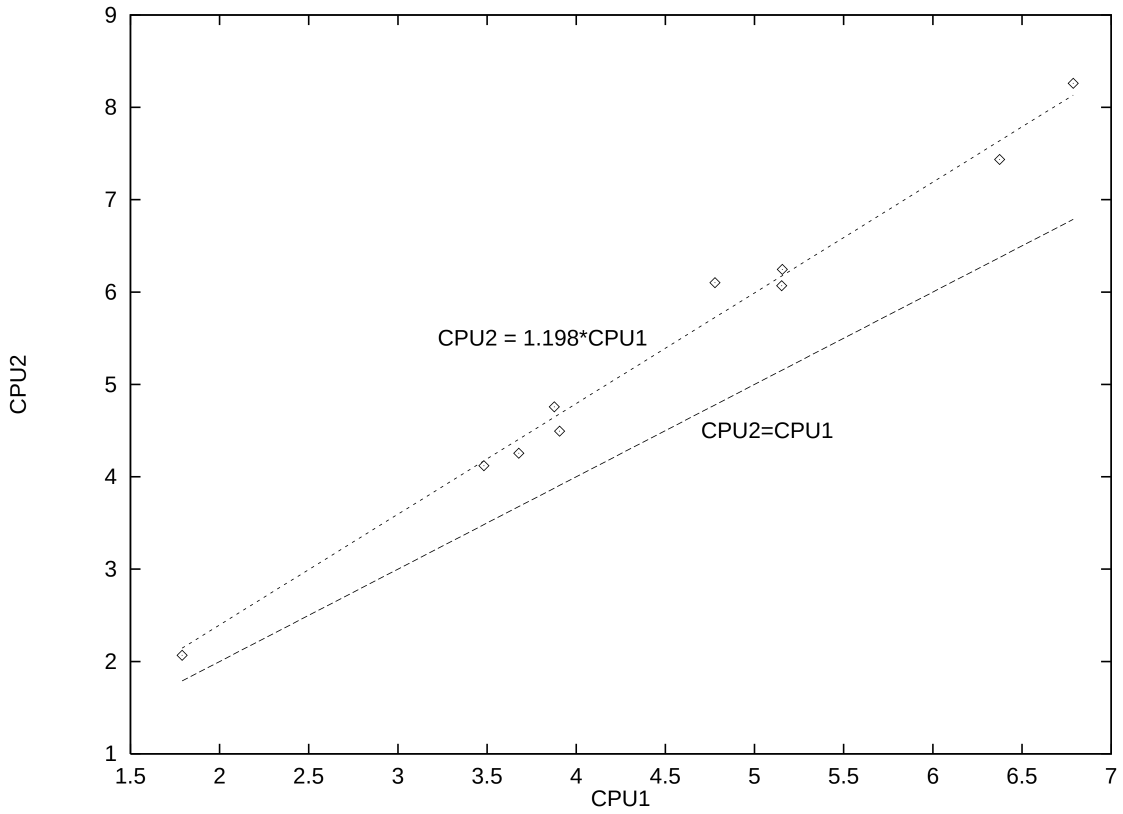
If we assume ϵ is normally distributed (a dubious assumption), we can determine the ratio $\frac{k^{CPU2}}{k^{CPU1}}$, by *linear regression*.

Program	CPU1	CPU2
P1	3.482514	4.119505
P2	3.677492	4.254761
P3	3.877525	4.756673
P4	6.787100	8.260877
P5	1.789549	2.067093
P6	5.156133	6.246014
P7	4.777698	6.101515
P8	3.906618	4.494181
P9	6.374434	7.434952
P10	5.152357	6.068562

```
> lm(cpu2 ~ cpu1 - 1)
Coefficients:
     cpu1
 1.198235

Degrees of freedom: 10 total; 9 residual
Residual standard error: 0.1808158
```

Results of Linear Regression



$$y_i^{CPU2} = 1.198 y_i^{CPU1}$$

Confidence Intervals on Regression Coefficients
Bootstrap Analysis

Draw 1000 samples of the (y_i^{CPU1}, y_i^{CPU2}) pairs.

Perform a linear regression on each sample.

Collect the coefficients, sort, and choose the 26th and 975th elements.

This takes 2.5 minutes in Splus and produces the interval (1.175634,1.223873).

Confidence Intervals on Regression Coefficients
Distributional Analysis

We will refer to the estimated value of the coeficient as b.

Let s be the residual standard error printed by Splus.
Then the standard error of the coefficient is $s_b = s/\sqrt{\Sigma_{i=1}^{n}(y_i^{CPU1})^2}$. A confidence interval for the coefficient is

$$b - t_{0.975}(n-1)s_b \leq b \leq b + t_{0.975}(n-1)s_b$$

In this example, this gives (1.17077,1.225699), which is slightly wider than the bootstrap interval.

Testing Techniques Using Human Subjects

Suppose we want to test two new user interfaces for an information retrieval system. We want to design a study where human subjects will use each of the new interfaces (plus the existing interface) to solve a series of six test problems while we measure the amount of time required to solve each problem (the "score").

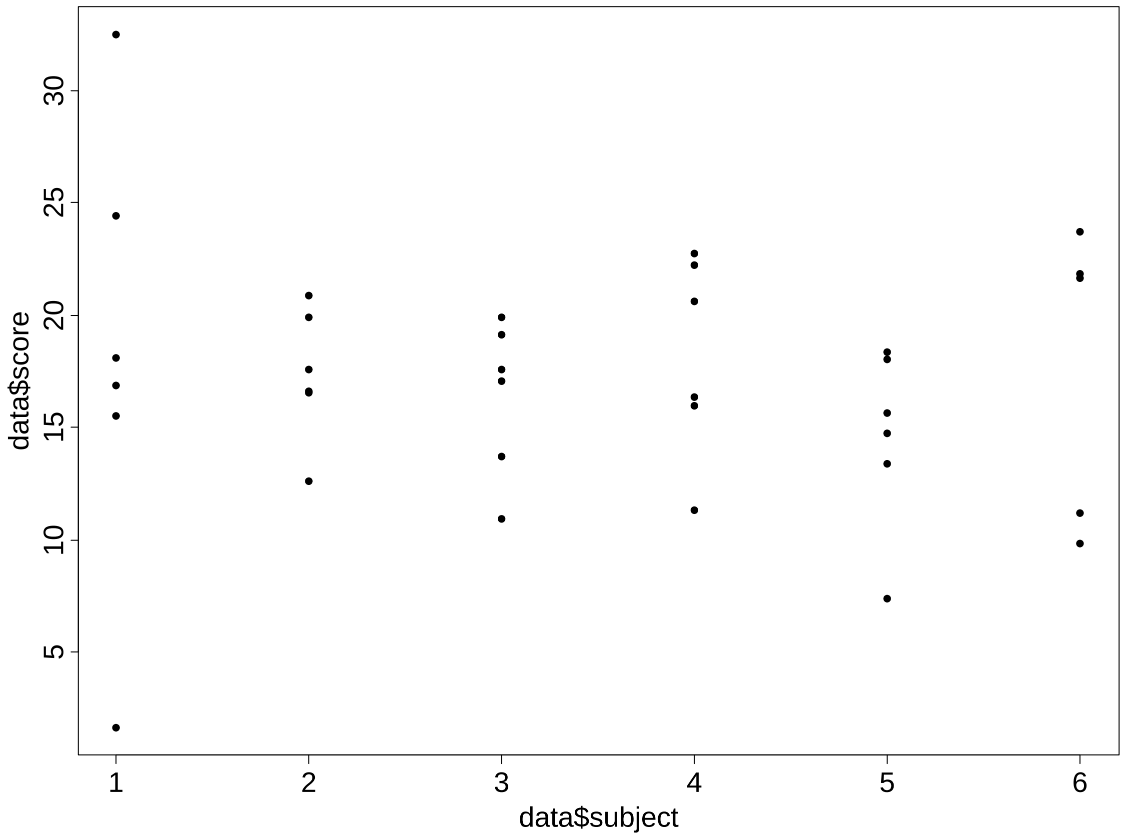
We want to be able to separate out any effects due to individual subjects. We also want to control for "learning" effects during the experiments (i.e., the subjects may become more comfortable with the physical set, the experimenter, etc.). Experiment Design:

Subject	Trial	Interface
1,4	1,2	existing
1,4	3,4	new-A
1,4	5,6	new-B
2,5	1,2	new-A
2,5	3,4	new-B
2,5	5,6	existing
3,6	1,2	new-B
3,6	3,4	existing
3,6	5,6	new-A

Data

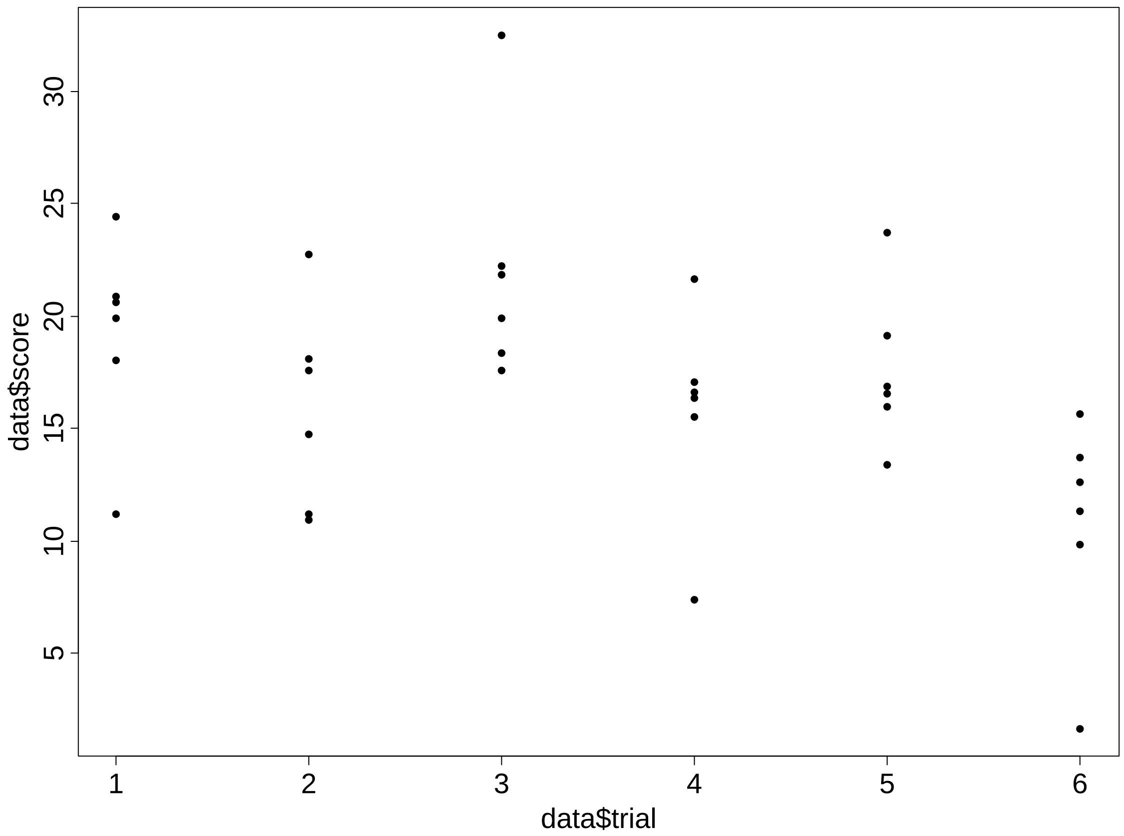
	subject	trial	interface	score		subject	trial	interface	score
1	1	1	1	24.415793	19	4	1	1	20.600881
2	1	2	1	18.120118	20	4	2	1	22.771879
3	1	3	2	32.474585	21	4	3	2	22.248769
4	1	4	2	15.529653	22	4	4	2	16.378728
5	1	5	3	16.871876	23	4	5	3	15.962623
6	1	6	3	1.642948	24	4	6	3	11.334707
7	2	1	2	20.874529	25	5	1	2	18.056006
8	2	2	2	17.577347	26	5	2	2	14.756638
9	2	3	3	19.882630	27	5	3	3	18.339784
10	2	4	3	16.636160	28	5	4	3	7.382027
11	2	5	1	16.588235	29	5	5	1	13.418451
12	2	6	1	12.654211	30	5	6	1	15.631696
13	3	1	3	19.918697	31	6	1	3	11.185494
14	3	2	3	10.958781	32	6	2	3	11.211880
15	3	3	1	17.559887	33	6	3	1	21.849035
16	3	4	1	17.073742	34	6	4	1	21.650350
17	3	5	2	19.116569	35	6	5	2	23.729027
18	3	6	2	13.731467	36	6	6	2	9.874535

Visualizing the Data



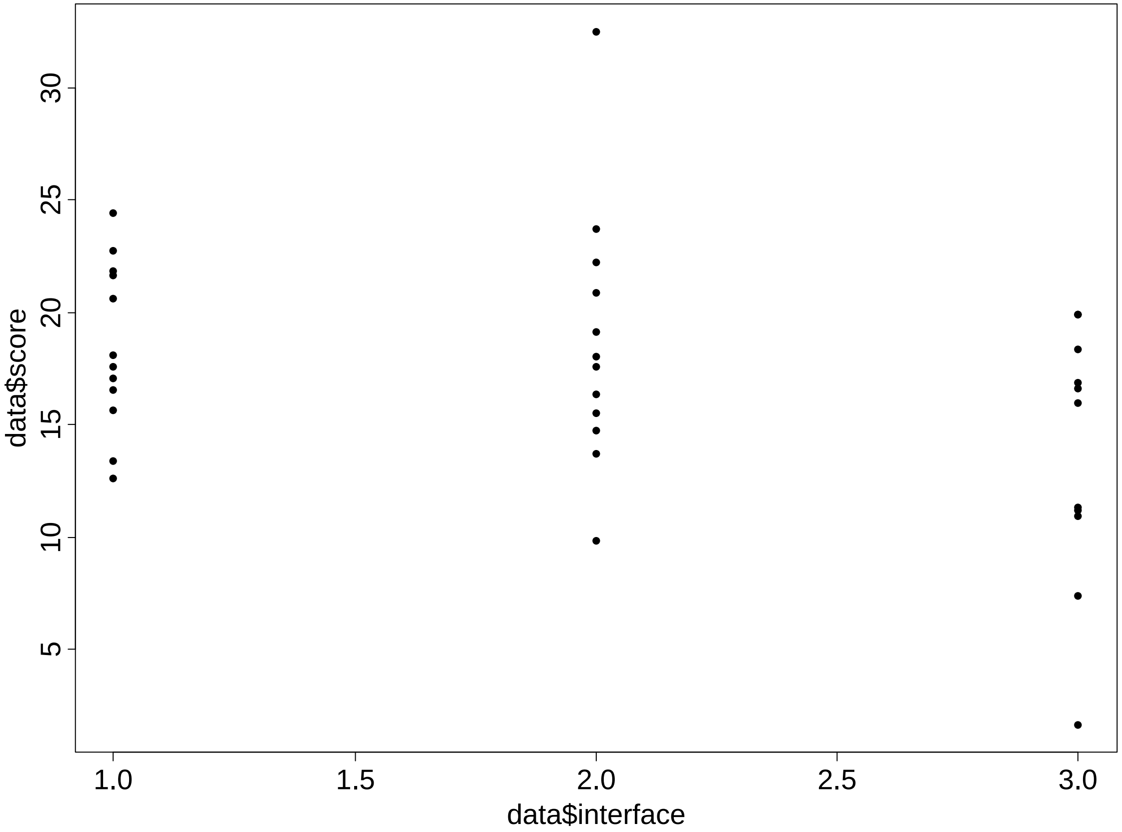
Plot of score versus subject.

Visualization (2)



Plot of score versus trial. We can see some “learning” effect.

Visualization (3)



Plot of score versus which interface was used.

Analysis of Variance (ANOVA)

We adopt the following model:

$$score = \mu + subject_i + trial_j + interface_k + \epsilon_{ijk}$$

where μ is the overall average performance on the tasks.
$subject_i$ is the change due to the ith subject.
$trial_j$ is the change due to the jth trial.
$interface_k$ is the change due to using the kth interface.
and ϵ_{ijk} is random noise.

The key idea of ANOVA is that variation in the observed score may be caused by variation within subjects and trials and noise as well as between the interfaces (which is what we are really trying to test).

Under the null hypothesis, variation within subjects, trials, and interfaces would be the same as the variation due to noise (because it would *all* be noise). So, if the variation between interfaces is much larger than the variation due to noise, we have a significant effect.

Analysis of Variance

$$\boxed{\begin{array}{l} score_{ijk} - \overline{score} \\ \text{of squares} \\ \text{(about mean)} \end{array}} = \boxed{\begin{array}{l} \overline{score_i} - \overline{score} \\ \text{sum of squares} \\ \text{from subjects} \end{array}} + \boxed{\begin{array}{l} \overline{score_j} - \overline{score} \\ \text{sum of squares} \\ \text{from trials} \end{array}} + \boxed{\begin{array}{l} \overline{score_k} - \overline{score} \\ \text{sum of squares} \\ \text{from interfaces} \end{array}} + \boxed{\begin{array}{l} score_{ijk} - sc\hat{o}re \\ \text{sum of squares} \\ \text{from } \epsilon \end{array}}$$

	Df	Sum of Sq	Mean Sq	F Value	Pr(F)
subject	1	17.9058	17.9058	0.777969	0.3843408
trial	1	158.1699	158.1699	6.872160	0.0132874
interface	1	155.0756	155.0756	6.737719	0.0141348
Residuals	32	736.5131	23.0160		

$$F = \frac{\text{Mean Sq}_i}{\text{Residual Mean Square}}$$

From this we see that the effects due to trial and interface are significant (at the 0.98 level), while the effect due to different subjects is insignificant.

ANOVA and Linear Regression

For every ANOVA, there is an equivalent (but usually complex) linear regression. It can be summarized by the coefficients:

```
(Intercept)   subject     trial interface
    17.7141 -0.412954 -1.227347 -2.541945
```

These aren't actually coefficients that multiply the values of the various variables. But their sign and magnitude tells us the direction of the interaction. There is a learning effect due to trials, because the score decreases with trials. There is an even larger improvement due to the interface.

True Model

Main effect: 18

Effect due to subject: $(0, 1, 0, 2, -1, -2)$

Effect due to trial: $(3, 2, 1, -1, -2, -3)$

Effect due to interface: $(2, 1, -3)$

Gaussian noise with mean 0 and standard deviation 4.

Analysis of Frequency Data using χ^2

Recall the user interface experiment.

	Usability	Heuristic	Walkthrough	Design Rules	Total
via technique:	30	105	30	13	178
expected by H_0:	44.5	44.5	44.5	44.5	178
$\dfrac{(\text{observed} - \text{expected})^2}{\text{expected}}$	4.72	82.25	4.72	22.30	144.00

We can test an hypothesis about frequencies by computing the set of frequencies expected under the hypothesis. Then, we can compute

$$\sum_i \frac{(\text{observed}_i - \text{expected}_i)^2}{\text{expected}_i}$$

This is distributed approximately as χ^2 with $n-1$ degrees of freedom. In this case, the observed value is highly significant!

We can test other hypotheses too, as long as the expected frequency is 5 or more in each cell.

Experimental Techniques from Psychology

- **Protocol Analysis**
  - Action protocols. Keystrokes, operations, eye-motion.
  - Verbal protocols. Thinking aloud.
- **Surveys and Questionnaires**
- **Experimental Manipulations**

Action Protocols

- **Sequence of actions, often with associated times.**
- **Keystrokes, mousing, application-level actions can be measured by software**
- **Eyetracking: Where was the subject looking?**

Verbal Protocols

- **Ask subjects to "think out loud"**

  This often requires prompting: "Please keep talking"

  Or use of some reminder, ideally automatic.

- **Do not ask subjects to explain their actions**

  Explanations produce rationalizations and reconstructions.

  These are unreliable.

- **Verbal protocols are incomplete**

  Do not capture subconscious or perceptual processing.

  Do not capture all conscious processing either.

  Absence from protocol does not imply absence from subject's mind.

- **Protocol interference?**

  The requirement to think aloud slows subjects by a factor of 2.

Protocol Analysis

- **Protocols are time-consuming to analyze**
- **Useful for identifying user goals and subgoals, conceptual categories, conceptual misunderstandings**
- **Typical analytical strategy:**

  Break problem-solving into episodes.

  Identify goals and actions in each episode.

  Try to track episode interruptions and shifts (hard!)

Protocol analysis can be very subjective.

Use multiple analysts.

Develop an agreed-upon set of criteria and stick to them.

Typical Results

- **Time Allocation:** "Subjects spent 20% of their time formulating queries, 30% waiting, 40% analyzing the results of queries, and 10% trying to figure out how to use the interface"
- **Problem Solving Strategies:** "Subjects lack a global plan, they repeatedly identify single bugs and fix them." or "Subjects systematically processed the specifications".

Uses of Protocol Analysis

- **Exploratory Research**

  Study how people currently solve a task:

  what are the information processing subtasks they must perform?

  What information sources do they use?

  Where would automation provide the biggest impact?

- **Design Evaluation**

  Build a system prototype

  Test human subjects on that prototype

- **Adjunct to Quantitative Measures**

  Measure speed and accuracy

  Use protocols to understand the causes of errors and misconceptions.

Surveys and Questionnaires

- **Advantages:**
  - **Cheap to perform.**
  - **Can use many subjects.**
  - **Easier to analyze than protocols.**
- **Disadvantages:**
  - **Non-responses may bias the results.**
  - **Hard to measure what you really want to measure.**
  - **Easily manipulated.**

Questionnaire Design

- **Ask questions in an unbiased manner**
- **Ask multiple questions related to the same topic**
- **Include questions on interfering or interacting factors**
- **Most questions should be multiple choice**

  Rating scales should have 5–7 levels at most.

  Rating scales should all be laid out the same way.

  All wording should be carefully checked for potential misunderstandings and biases.

  Pre-test the questionnaire (with verbal protocols).

  Consider randomizing the order of questions for different subjects.
- **Include some free-response questions and room for comments**

  Subjects may have other things they want to talk about (detect incompleteness)

  Subjects may want to complain about questions or explain their answers (detect problems with questions).

Questionnaire Design (2)

- **Make it easy for subjects to answer**

  Keep it as short as possible

  Provide pre-paid return envelopes

  Provide a reward for returning the questionnaire.

- **Choose a random sample—avoid self-selection**
- **Follow up with non-responding subjects**
- **Maintain confidentiality and reassure subjects of this.**
- **Offer every subject a free copy of the survey results.**

Questionnaire Analysis

- **Reliability and Validity**
- **Representativeness**
- **Use ANOVA or Contingency Table Analysis to Control for Interactions**
- **Use Care in Grouping Data**

Contingency Table Analysis

- **Null Hypothesis: The factors are independent**
- **Alternative Hypothesis: The factors are not independent**

Measured Data:

	Low Satisfaction	High Satisfaction	
Low Experience	22	12	34 (.466)
High Experience	10	29	39 (.534)
	32 (.438)	41 (.562)	73 (1.00)

Null Hypothesis:

	Low Satisfaction	High Satisfaction	
Low Experience	14.9	19.1	34
High Experience	17.1	21.9	39
	32	41	73

χ^2 test with 2 degrees of freedom:

$$\sum \frac{(\text{observed} - \text{expected})^2}{\text{expected}} = 11.27 > 5.99$$

$(p = 0.0036)$

Experimental Manipulations

- **Randomly divide subjects into groups**
- **Each group employs a different technique or method**
- **A dependent variable is measured**
- **ANOVA or χ^2 test is used to analyze the effect of the technique on the dependent variable**

The user-interface analysis study is an excellent example of this.

Computational Experiments

- **Questions to be asked:**

  Why does one algorithm do better than another?

  When does one algorithm do better than another?

- **Hold algorithms constant, vary problems**

  Problem attributes: size, amount of noise, degree of nonlinearity, amount of data, etc.

- **Hold problems constant, vary algorithms**

  NETtalk study: explored various modifications along a continuum from one algorithm to another.

Parallel Computing

- **Speedup**. Ratio of *best* serial running time to the running time on n processors.
- **Scaling**. Ratio of size of biggest problem that can be run on one machine to the size of the biggest problem that can be run on n machines.

Comparing Results Across Machines

- **CPU time can't be used**

  Machines have different speeds, memory architectures, user loads, etc.

- **Identify proxies for CPU time (and other resources)**

  Number of iterations

  Number of comparisons

  Number of transactions

  Number of rule firings

- **One proxy unit should always take the same amount of CPU time on a given machine**