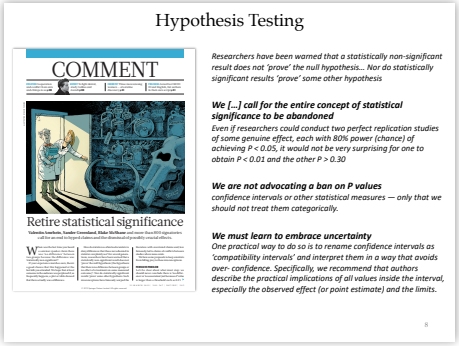
Business Analytics Using Computational Statistics

Wees 6-7
Multigroup Analysis

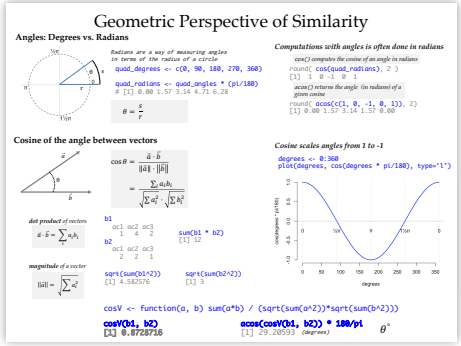
Week 9
Data Similarity

Week 10
Linear Regression

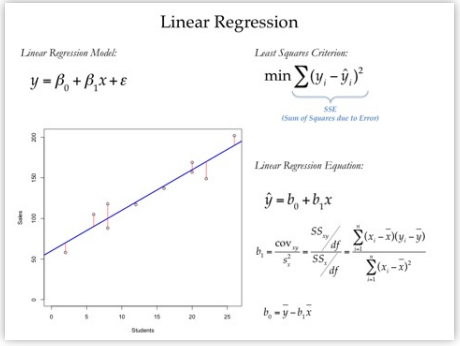
Recap of Testing and Power



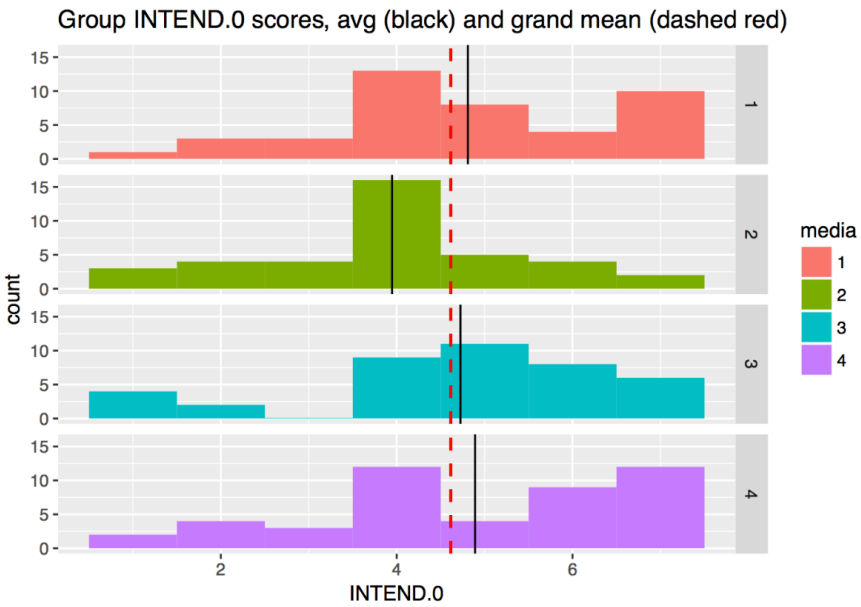
Similarity

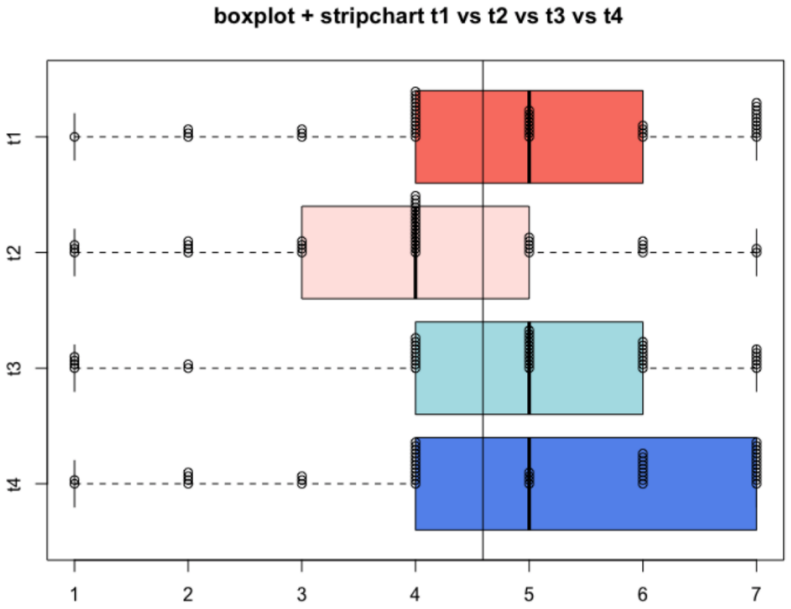


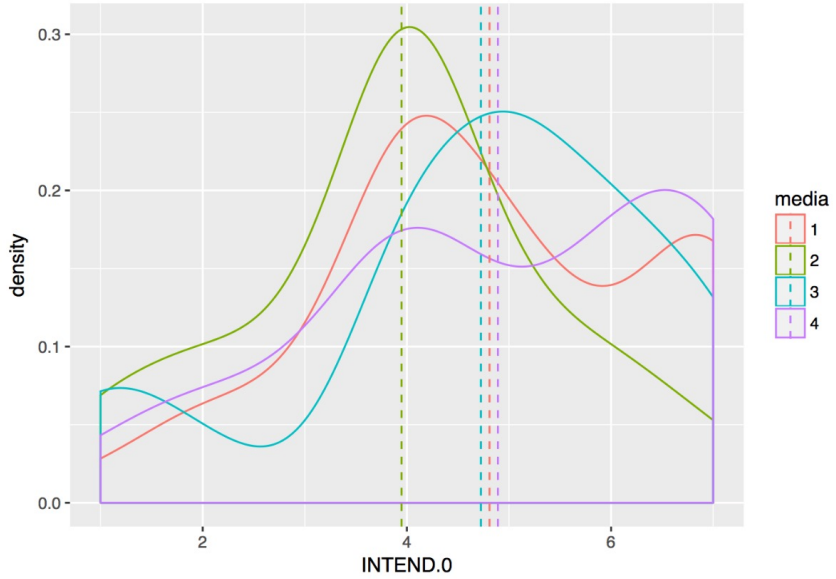
Linear Regression

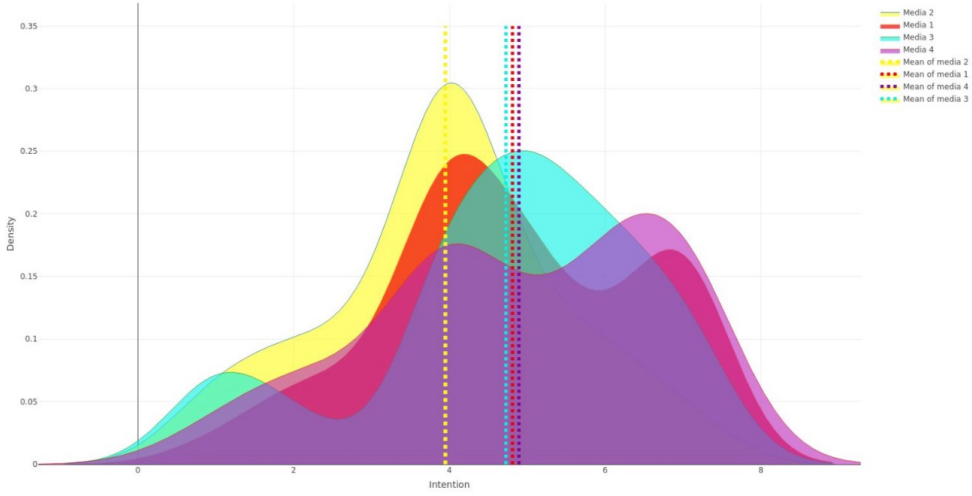


Sample Visualizations









One-Way ANOVA

Loading and Shaping Data

```
m1 <- read.csv("pls-media1.csv", header=TRUE)
m2 <- read.csv("pls-media2.csv", header=TRUE)
m3 <- read.csv("pls-media3.csv", header=TRUE)
m4 <- read.csv("pls-media4.csv", header=TRUE)

experiment <- rbind(m1, m2, m3, m4)
 media INTEND.0 INTEND.1 INTEND.2 ATT.0
1       1        3        5        4     3
2       1        5        5        4     6
43      2        4        4        4     5
44      2        6        6        5     6
81      3        1        2        3     4
82      3        4        4        5     4
121     4        3        5        4     3
122     4        4        4        4     4

exp_groups <- split(experiment$INTEND.0, experiment$media)
$`1`
 [1] 3 5 4 5 5 4 4 5 4 1 7 3 4 7 7 5 4 6 4 6 4 2 7 7 6 4 2 3 5 7 4 5 4 4 7 7 4 2 5 6 7 7
$`2`
 [1] 4 6 4 4 5 4 4 4 4 7 4 4 2 4 4 5 4 1 7 3 4 3 6 4 2 5 5 3 1 2 5 2 6 3 1 4 4 6
$`3`
 [1] 1 4 1 5 6 6 5 7 5 5 7 5 4 2 5 4 6 5 4 7 5 4 5 6 6 1 4 2 7 4 4 1 4 7 6 6 5 6 5 7
$`4`
 [1] 3 4 4 2 7 7 5 7 5 6 4 7 6 4 4 1 2 1 6 7 2 7 4 6 5 5 6 4 3 2 6 4 6 6 4 7 6 7 3 4 7 4 7 7 7 4
```

Descriptive Statistics

```
media_mean <- sapply(split(experiment$INTEND.0, experiment$media), mean)
round(media_mean, 2)   # means:   4.81 3.95 4.72 4.89

media_length <- sapply(
  split(experiment$INTEND.0, experiment$media), length
)                      # lengths: 42   38   40   46

media_se = media_sd / sqrt(media_length)
                      # std errs: 0.25 0.24 0.27 0.26
```

Boxplot

```
boxplot(experiment$INTEND.0 ~ experiment$media,
        xlab="Media", ylab="Response",
        medcol="gray", medlty="dashed", medlwd=1)
```

draws line segments on plot using existing axis units

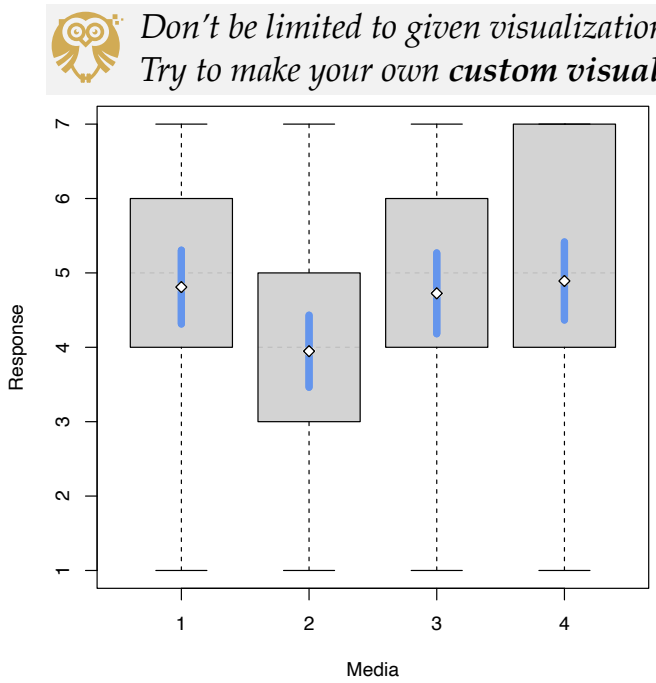
x0: start x *y0: start y*

```
segments( 1:4, media_mean - 1.96*media_se,
          1:4, media_mean + 1.96*media_se, …)
```

x1: end x *y1: end y*

draws points on plot

```
points(1:4, media_mean, pch=23, bg="white")
```



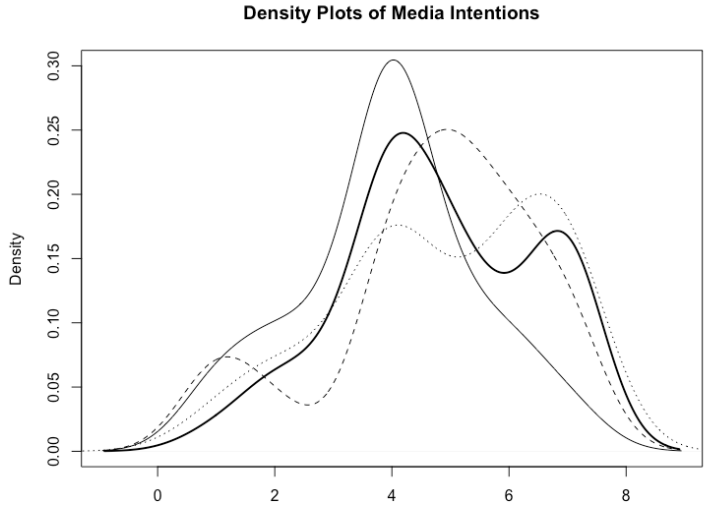
*Don't be limited to given visualization functions. Try to make your own **custom visualizations**.*

Standard errors and quantiles do not agree with each other.

Data may not be symmetrically distributed

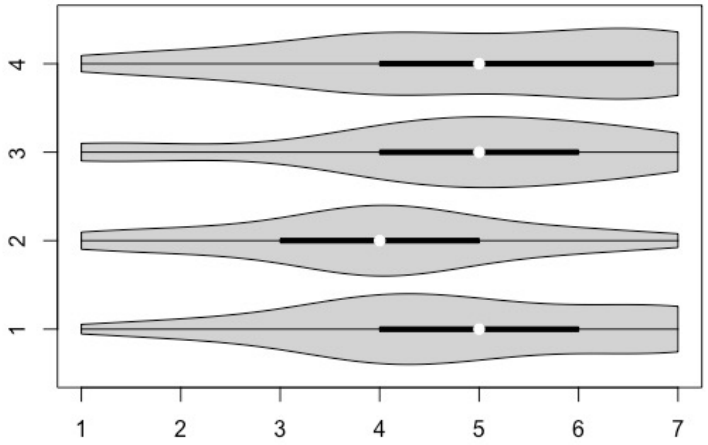
Density Plots

```
plot(density(exp_groups$`1`), …)
lines(density(exp_groups$`2`))
lines(density(exp_groups$`3`), lty="dashed")
lines(density(exp_groups$`4`), lty="dotted")
```



Violin Plots: *boxplot + rotated density plot*

```
library(vioplot)
vioplot(exp_groups$`4`, exp_groups$`3`,
        exp_groups$`2`, exp_groups$`1`,
        col="lightgray", horizontal = TRUE)
```



One-Way ANOVA by Code

```
all_media <- experiment$INTEND.0
media_grand_mean <- mean(all_media)
n_total <- length(all_media) # 166
k <- length(exp_groups)      # 4

sstr <- sum(media_length * ((media_mean - media_grand_mean)^2))
# 22.52285
df_mstr <- k - 1
# 3

sse <- sum((media_length - 1) * (media_sd^2))
# 464.8024
df_mse <- n_total - k
# 162

sstr + sse
# 487.3253

sst <- sum((all_media - media_grand_mean)^2)
# 487.3253

mstr <- sstr/df_mstr # 7.507617
mse <- sse/df_mse    # 2.869151

F <- mstr/mse        # 2.616669

pvalue <- pf(F, df_mstr, df_mse, lower.tail=FALSE)
# 0.05289015
```

factor(): *encodes vector of values as having distinct categorical levels*

One-Way ANOVA using *aov()*

```
ftest_aov <- aov(experiment$INTEND.0 ~ factor(experiment$media))
summary(ftest_aov)
#                          Df Sum Sq Mean Sq F-value Pr(>F)
# factor(experiment$media)   3   22.5   7.508   2.617 0.0529 .
# Residuals                162  464.8   2.869
```

☹️

Not significant at 95% confidence
(does this match your visual intuition?)

Tukey Post-hoc Pairwise Tests

```
TukeyHSD(ftest_aov)
# 95% family-wise confidence level
# diff         lwr       upr     p adj
# 2-1 -0.86215539 -1.84660332 0.1222925 0.1085727
# 3-1 -0.08452381 -1.05596494 0.8869173 0.9959223
# 4-1  0.08178054 -0.85664966 1.0202107 0.9959032
# 3-2  0.77763158 -0.21843807 1.7737012 0.1825044
# 4-2  0.94393593 -0.01996662 1.9078385 0.0573229
# 4-3  0.16630435 -0.78431033 1.1169190 0.9687417
```

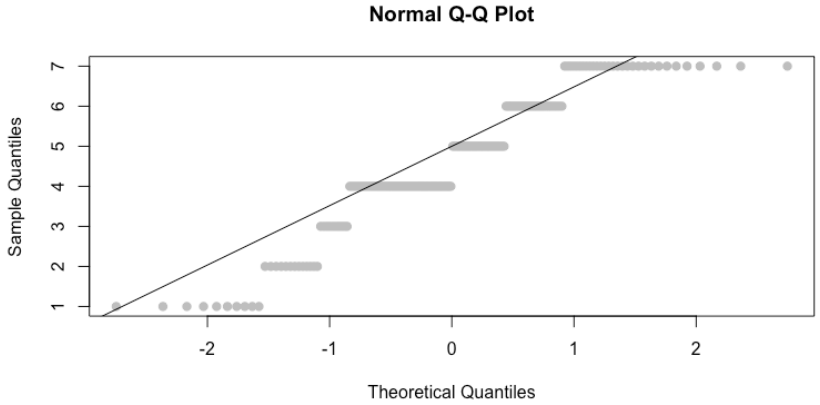
☹

Pairwise tests not significant *at 95%* ***family-wise*** *confidence*

Assumptions of ANOVA

Normality

```
qqnorm(experiment$INTEND.0, pch=19, col="gray")
qqline(experiment$INTEND.0)
```



Equal Variances

```
f_test <- oneway.test(experiment$INTEND.0 ~ factor(experiment$media), var.equal = FALSE)
  # F = 2.9533, num df = 3.000, denom df = 89.608, p-value = 0.03681

qf(df1=4-1, df2=nrow(experiment)-4, p=c(0.95, 0.99))
  # [1] 2.660406 3.904807
```

95% and 99% cutoffs (one-tailed)

var.equal: *this is a domain knowledge decision based on my understanding of how groups should vary within themselves*

F-test assuming unequal variances detects significant difference between groups

Kruskal Wallis

```
all_ranks <- rank(experiment$INTEND.0)
  [1]  28.5  97.5  58.5  97.5  97.5  58.5  58.5  97.5  58.5   5.5 151.5  28.5  58.5 151.5 151.5  97.5 […]

group_ranks <- split(all_ranks, experiment$media)
$`1`
 [1]  28.5  97.5  58.5  97.5  97.5  58.5  58.5  97.5  58.5   5.5 151.5  28.5  58.5 151.5 151.5  97.5 […]
$`2`
 [1]  58.5 124.0  58.5  58.5  97.5  58.5  58.5  58.5  58.5 151.5  58.5  58.5  17.0  58.5  58.5  97.5 […]
$`3`
 [1]   5.5  58.5   5.5  97.5 124.0 124.0  97.5 151.5  97.5  97.5 151.5  97.5  58.5  17.0  97.5  58.5 […]
$`4`
 [1]  28.5  58.5  58.5  17.0 151.5 151.5  97.5 151.5  97.5 124.0  58.5 151.5 124.0  58.5  58.5   5.5 […]

group_ranksums <- sapply(group_ranks, sum)
       1        2        3        4
3693.5 2421.0 3556.0 4190.5

group_n <- sapply(group_ranks, length)
 1  2  3  4
42 38 40 46

group_rankmeans <- sapply(group_ranks, mean)
       1        2        3        4
87.94048 63.71053 88.90000 91.09783

N <- nrow(experiment)
groups <- data.frame(ranksum = group_ranksums,
                     n = group_n,
                     rankmeans = group_rankmeans)
ngroups <- nrow(groups)
  ranksum  n rankmeans
1  3693.5 42  87.94048
2  2421.0 38  63.71053
3  3556.0 40  88.90000
4  4190.5 46  91.09783

rank_msq <- function(strategy) {
  strategy['ranksum']^2 / strategy['n']
}

H <- 12/(N*(N+1))*sum(apply(groups, 1, rank_msq)) - 3*(N+1)
[1] 8.45466

kw_p <- 1 - pchisq(H, df=ngroups-1)
[1] 0.03749292
```

Group Rank Sums

Group Sizes

Group Rank Means

Data Frame of Group Information

Function for Mean Squared Group Ranks

$$H = \frac{12}{N(N+1)} \sum_{i=1}^{k} \frac{R_i^2}{n_i} - 3(N+1)$$

Kruskal Wallis test finds significant difference
between values of the four groups

Dunn Post-hoc Pairwise Tests

```
dunnTest(experiment$INTEND.0 ~ factor(experiment$media))

# Dunn (1964) Kruskal-Wallis multiple comparison
# p-values adjusted with the Holm method.
#
#   Comparison           Z     P.unadj      P.adj
# 1      1 - 2  2.30087819 0.021398517 0.08559407
# 2      1 - 3 -0.09233644 0.926430736 0.92643074
# 3      2 - 3 -2.36408588 0.018074622 0.09037311
# 4      1 - 4 -0.31452459 0.753122646 1.00000000
# 5      2 - 4 -2.65613380 0.007904225 0.04742535
# 6      3 - 4 -0.21613379 0.828883460 1.00000000
```

Dunn test detects significant difference between group 2 and group 4

Computational Take-aways

***Vectorized Coding** and **Functional Iteration** makes our code more readable and recognizable*

Vectorized code expresses math directly

```
sstr <- sum(media_length * ((media_mean - media_grand_mean)^2))
mstr <- sstr/df_mstr
```

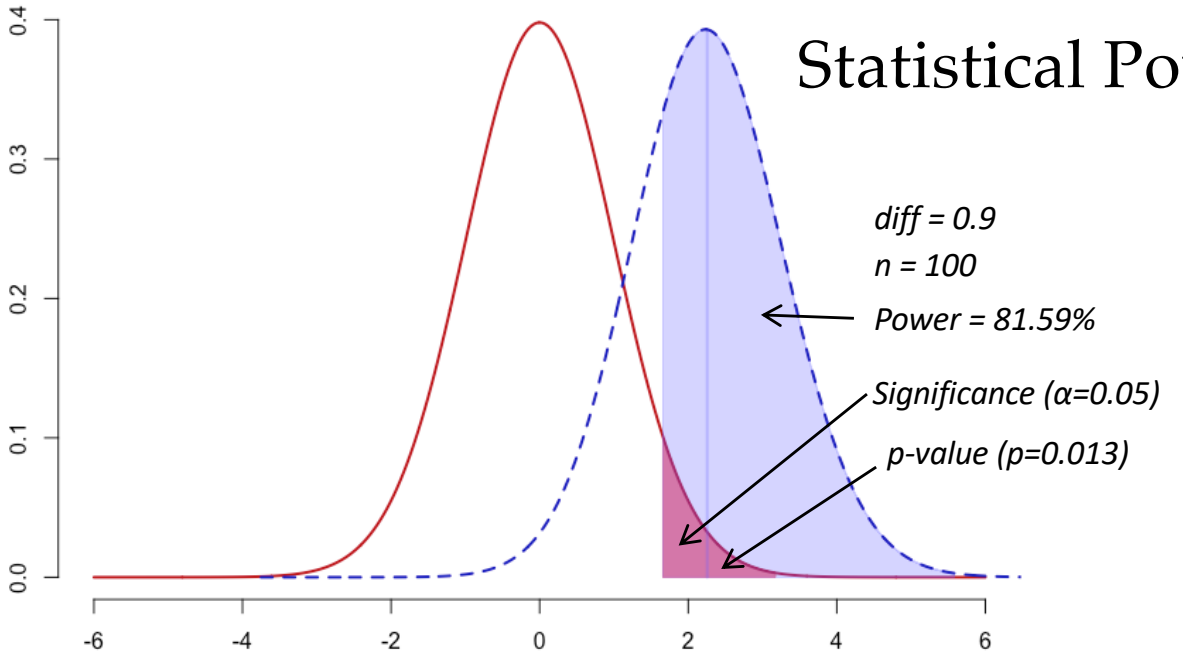
$$MSTR = \frac{SSTR}{df_{MSTR}} = \frac{\sum_{j=1}^{k} n_j(\bar{x}_j - \bar{\bar{x}})^2}{k-1}$$

Functional iteration allows us to express math iterations easily

```
rank_sq <- function(strategy) {
  strategy['ranksum']^2 / strategy['n']
}

H <- 12/(N*(N+1))*sum(apply(groups, 1, rank_sq)) - 3*(N+1)
```

$$H = \frac{12}{N(N+1)} \sum_{i=1}^{k} \frac{R_i^2}{n_i} - 3(N+1)$$

Statistical Power as Scientific Resolution



Good enough for identifying the shape?

Good enough for identifying the object?

Good enough to see if important features are useful and valid?

Good enough for science?

Hypothesis Testing

COMMENT

EVOLUTION Cooperation and conflict from ants and chimps to us p.308

HISTORY To fight denial, study Galileo and Arendt p.309

CHEMISTRY Three more unsung women — of astatine discovery p.311

PUBLISHING As well as ORCID ID and English, list authors in their own script p.311

ILLUSTRATION BY DAVID PARKINS

Retire statistical significance

Valentin Amrhein, **Sander Greenland**, **Blake McShane** and more than 800 signatories call for an end to hyped claims and the dismissal of possibly crucial effects.

When was the last time you heard a seminar speaker claim there was 'no difference' between two groups because the difference was 'statistically non-significant'?

If your experience matches ours, there's a good chance that this happened at the last talk you attended. We hope that at least someone in the audience was perplexed if, as frequently happens, a plot or table showed that there actually was a difference.

How do statistics so often lead scientists to deny differences that those not educated in statistics can plainly see? For several generations, researchers have been warned that a statistically non-significant result does not 'prove' the null hypothesis (the hypothesis that there is no difference between groups or no effect of a treatment on some measured outcome)[1]. Nor do statistically significant results 'prove' some other hypothesis. Such misconceptions have famously warped the literature with overstated claims and, less famously, led to claims of conflicts between studies where none exists.

We have some proposals to keep scientists from falling prey to these misconceptions.

PERVASIVE PROBLEM

Let's be clear about what must stop: we should never conclude there is 'no difference' or 'no association' just because a *P* value is larger than a threshold such as 0.05 ▶

https://www.nature.com/articles/d41586-019-00857-9

Researchers have been warned that a statistically non-significant result does not 'prove' the null hypothesis… Nor do statistically significant results 'prove' some other hypothesis

We […] call for the entire concept of statistical significance to be abandoned

Even if researchers could conduct two perfect replication studies of some genuine effect, each with 80% power (chance) of achieving $p < 0.05$, it would not be very surprising for one to obtain $p < 0.01$ and the other $p > 0.30$

***We are not advocating a ban on P values**, confidence intervals or other statistical measures — only that we should not treat them categorically.*

We must learn to embrace uncertainty

One practical way to do so is to rename confidence intervals as 'compatibility intervals' and interpret them in a way that avoids overconfidence. Specifically, we recommend that authors describe the practical implications of all values inside the interval, especially the observed effect (or point estimate) and the limits.

What's in a Vector?

From Swirl Tutorial 1 quiz responses by BACS students:

> *"I think it's because R is all about doing mathematical calculations and maneuvers, so it uses the word 'vector', which, in mathematical fields such as linear algebra, means a one-dimensional matrix."*

> *"vector feels like a series of value with direction. so c(1,2,3) and c(3,2,1) are different."*

> *"I have no idea :("*

Vectors help us describe displacements in space:

quantity with a magnitude and a direction

Column vector

```
v <- c(4, 2)
```

$$\vec{v} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

Row vector

```
t(v)
```

`t()` ***transpose*** *of vector or matrix (flips columns and rows)*

$$\vec{v}^T = < 4, 2 >$$

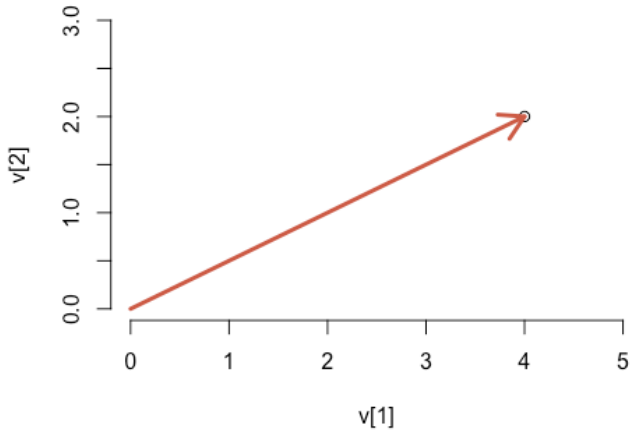
R follows some principles of vector arithmetic:

```
3*v
[1] 12  6
```

- Multiplication by scalar

```
v + v
[1] 8 4
```

- Addition of vectors




```
plot(t(v))
arrows(0, 0, v[1], v[2], length=0.2)
```

`arrows()` *adds arrows to an existing plot*
- `x0, y0, x1, y1:` *start/end coordinates*
- `length:` *size of arrow head*

How does thinking about data geometrically help us?

PicCollage Recommendation System

Image and photo collage app for iOS/Android

100+ million downloads worldwide
10-15 million active users every month

Sells **sticker bundles** ($$$)

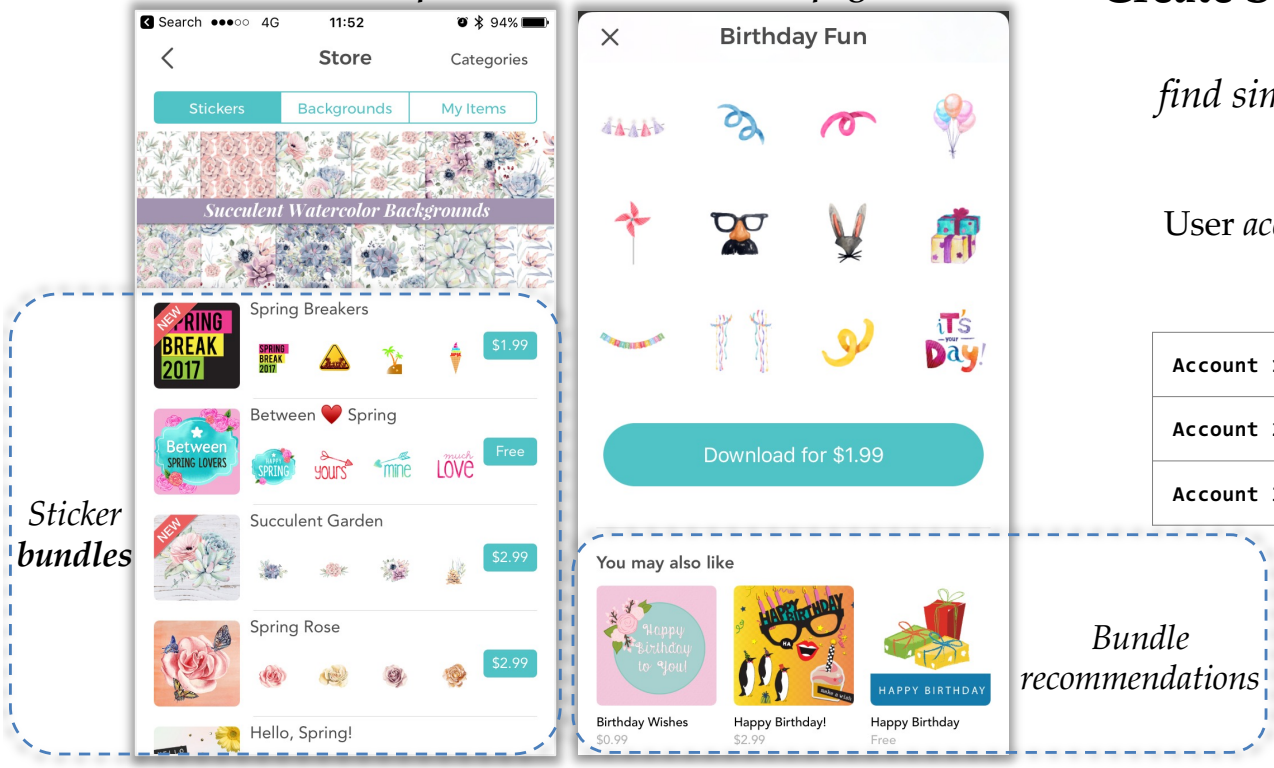
Made by **Cardinal Blue**, in Taiwan

Sticker bundles can be recommended by designers based on their expertise.

But how might we automate a recommendation system that can recommend similar sticker bundles to a chosen bundle?

Sticker shop

Bundle page



Create Sticker Bundle Recommendations

For any given bundle,
find similar bundles that might interest users

Data

User *accounts* and total *bundle usage* per account

	Bundle 1	Bundle 2	Bundle 3	Bundle 4
Account 1	1	2	4	3
Account 2	4	2	4	0
Account 3	2	1	2	0

How can we check if bundles are "similar"?

Vectors and Matrices

1. Data as Vectors

```
b1 <- c(ac1 = 1,  ac2 = 4,  ac3 = 2)
b2 <- c(ac1 = 2,  ac2 = 2,  ac3 = 1)
b3 <- c(ac1 = 4,  ac2 = 4,  ac3 = 2)
b4 <- c(ac1 = 3,  ac2 = 0,  ac3 = 0)

ac_bundles <- cbind(b1, b2, b3, b4)

class(ac_bundles)
[1] "matrix" "array"
```

```
ac_bundles <- matrix(
  c(1, 2, 4, 3,
    4, 2, 4, 0,
    2, 1, 2, 0),
  nrow=3, ncol=4, byrow = TRUE,
  dimnames=list(c("ac1", "ac2", "ac3"),
                c("b1", "b2", "b3", "b4")))
```

2. Matrix: vectors as columns/rows

*Accounts in 4D **bundle-space***

```
ac_bundles
     b1 b2 b3 b4
ac1   1  2  4  3
ac2   4  2  4  0
ac3   2  1  2  0
```

***Columns** and **Rows** are both vectors*

*Columns are interpreted as **dimensions***
*Rows are interpreted as **cases/observations***

*Extracting **row vectors** and **column vectors***

```
ac_bundles[1, ]
b1 b2 b3 b4
 1  2  4  3
```

*Each **account** is a dimension of bundles*
(it tells us one account's preference of bundles)

```
ac_bundles[, 1]
ac1 ac2 ac3
  1   4   2
```

*Each **bundle** is a dimension of accounts*
(it tells us how accounts differ on one bundle)

***Bundles** in 3D account-space*

```
t(ac_bundles)
    ac1 ac2 ac3
b1    1   4   2
b2    2   2   1
b3    4   4   2
b4    3   0   0
```

`t()` *transpose of vector or matrix*
(flips columns and rows)

Vectors in Data

1. Data as Vectors

```
b1 <- c(ac1 = 1,  ac2 = 4, ac3 = 2)
b2 <- c(ac1 = 2,  ac2 = 2, ac3 = 1)
b3 <- c(ac1 = 4,  ac2 = 4, ac3 = 2)
b4 <- c(ac1 = 3,  ac2 = 0, ac3 = 0)

ac_bundles <- cbind(b1, b2, b3, b4)
```

2. Matrix: vectors as columns/rows

Accounts in 4D bundle-space

```
ac_bundles
```

	b1	b2	b3	b4
ac1	1	2	4	3
ac2	4	2	4	0
ac3	2	1	2	0

***Columns** and **Rows** are both vectors*

*Columns are interpreted as **dimensions***
*Rows are interpreted as **cases/observations***

Bundles in 3D account-space

```
t(ac_bundles)
```

	ac1	ac2	ac3
b1	1	4	2
b2	2	2	1
b3	4	4	2
b4	3	0	0

`t()` *transpose of vector or matrix (flips columns and rows)*

```
ac_bundles[1, ]
```

b1	b2	b3	b4
1	2	4	3

*Each **account** is a dimension of bundles (it tells us one account's preference of bundles)*

```
ac_bundles[, 1]
```

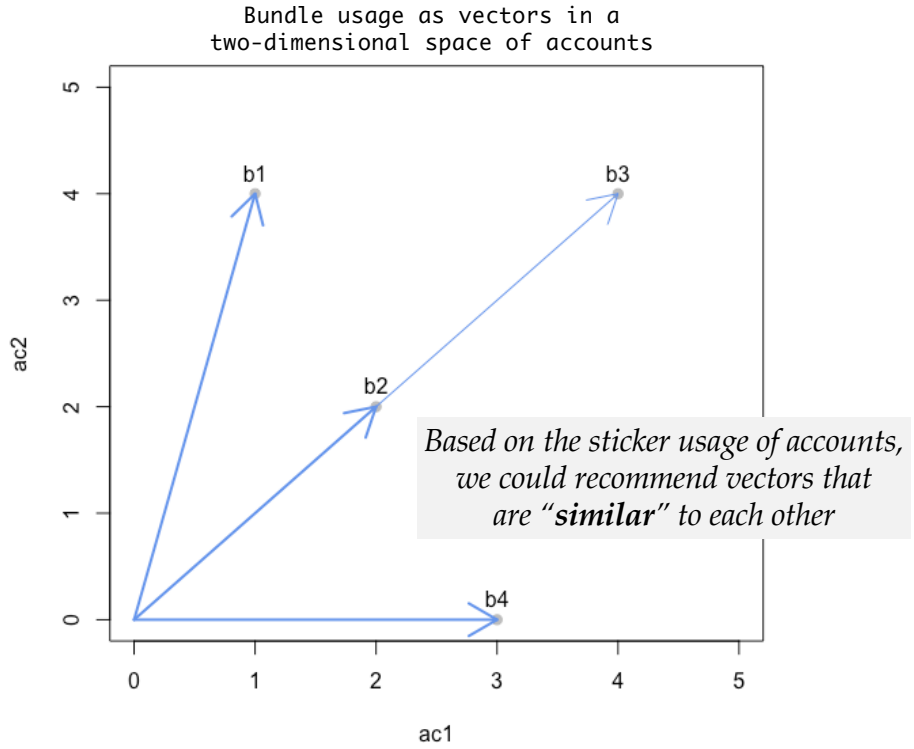
ac1	ac2	ac3
1	4	2

*Each **bundle** is a dimension of accounts (it tells us how accounts differ on one bundle)*

3. Visualizing Similarity

visualizing bundles in 2D account-space

```
plot(t(ac_bundles), xlim=c(0, 5), ylim=c(0, 5))
arrows(0, 0, ac_bundles[1, ], ac_bundles[2, ])
text(ac_bundles[1, ], ac_bundles[2, ]+0.3, colnames(ac_bundles))
```



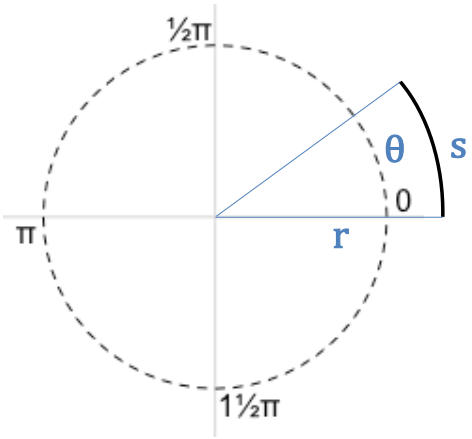
*Based on the sticker usage of accounts, we could recommend vectors that are "**similar**" to each other*

*Geometry gives us a **model** of the world, across domains*

a simplified representation of a problem that is helpful and understandable

Angles: Degrees vs. Radians

Angles: Degrees vs. Radians



$$\theta = \frac{s}{r}$$

Radians *are a way of measuring angles in terms of the* ***circumference*** *of a unit circle*

```
quad_degrees <- c(0, 90, 180, 270, 360)
# [1]   0  90 180 270 360

quad_radians <- quad_degrees * (pi/180)
# [1] 0.00 1.57 3.14 4.71 6.28
```

Computations with angles is often done in radians

Cosine *for angles in radians*

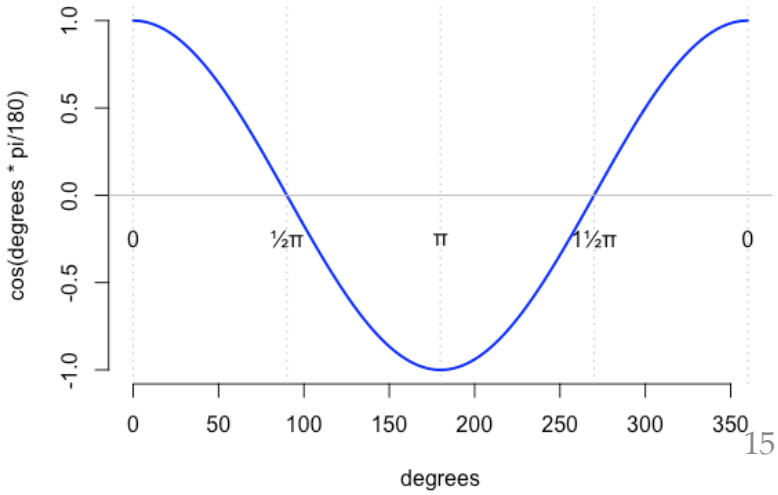
```
round( cos(quad_radians), 2 )
[1]  1  0 -1  0  1
```

Arc-cosine *returns the angle in radians*

```
round( acos(c(1, 0, -1, 0, 1)), 2)
[1] 0.00 1.57 3.14 1.57 0.00
```

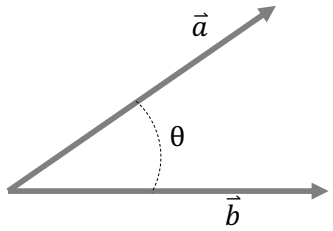
Cosine ***scales*** *(normalizes) angles from 1 to -1*

```
degrees <- 0:360
plot(degrees, cos(degrees * pi/180), type='l')
```



Cosine: Geometric Measure of Similarity

Cosine as a measure of the angle between vectors



$$\cos\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\cdot\|\vec{b}\|} = \frac{\sum_i a_i b_i}{\sqrt{\sum a_i^2}\cdot\sqrt{\sum b_i^2}}$$

Consider two vectors

```
      b1                b2
ac1 ac2 ac3       ac1 ac2 ac3
  1   4   2         2   2   1
```

***Dot product** of vectors*

$$\vec{a}\cdot\vec{b} = \sum_i a_i b_i$$

```
sum(b1 * b2)
[1] 12
```

***Magnitude** of a vector*

$$\|\vec{a}\| = \sqrt{\sum a_i^2}$$

```
sqrt(sum(b1^2))
[1] 4.582576
```

```
sqrt(sum(b2^2))
[1] 3
```

***Cosine** of vectors*

$$\cos\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\cdot\|\vec{b}\|}$$

```
cosV <- function(a, b) sum(a*b) / (sqrt(sum(a^2))*sqrt(sum(b^2)))

cosV(b1, b2)
[1] 0.8728716
```

```
acos(cosV(b1, b2)) * 180/pi
[1] 29.20593  (degrees)
```

Cosine Similarity

Cosine Similarity

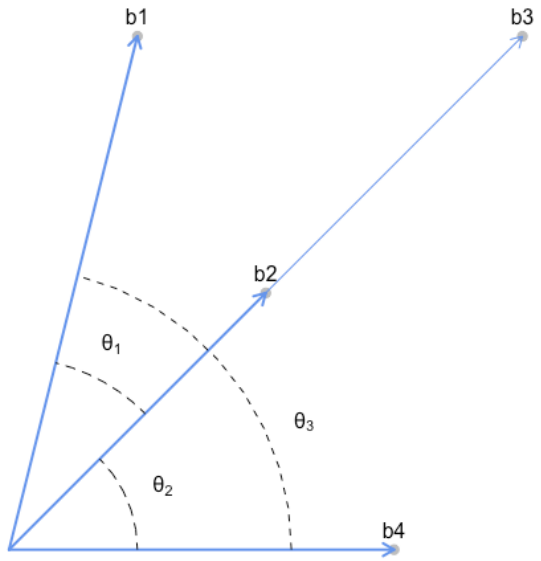
```
install.packages("lsa")

library(lsa)
cosine(ac_bundles)
```

```
     b1   b2   b3   b4
b1 1.00 0.87 0.87 0.22
b2 0.87 1.00 1.00 0.67
b3 0.87 1.00 1.00 0.67
b4 0.22 0.67 0.67 1.00
```

θ°

```
acos(cosine(ac_bundles)) * 180/pi
```

```
      b1    b2    b3    b4
b1  0.00 29.21 29.21 77.40
b2 29.21  0.00  0.00 48.19
b3 29.21  0.00  0.00 48.19
b4 77.40 48.19 48.19  0.00
```



Sorted Names Matrix

```
   [,1] [,2] [,3] [,4]
b1 "b1" "b2" "b3" "b4"
b2 "b2" "b3" "b1" "b4"
b3 "b3" "b2" "b1" "b4"
b4 "b4" "b2" "b3" "b1"
```

Recommendation Matrix

	1st	2nd	3rd
Bundle 1	B2	B3	B4
Bundle 2	B3	B1	B4
Bundle 3	B2	B1	B4
Bundle 4	B2	B3	B1

Now, given a bundle to compare to, we can recommend *similar* bundles

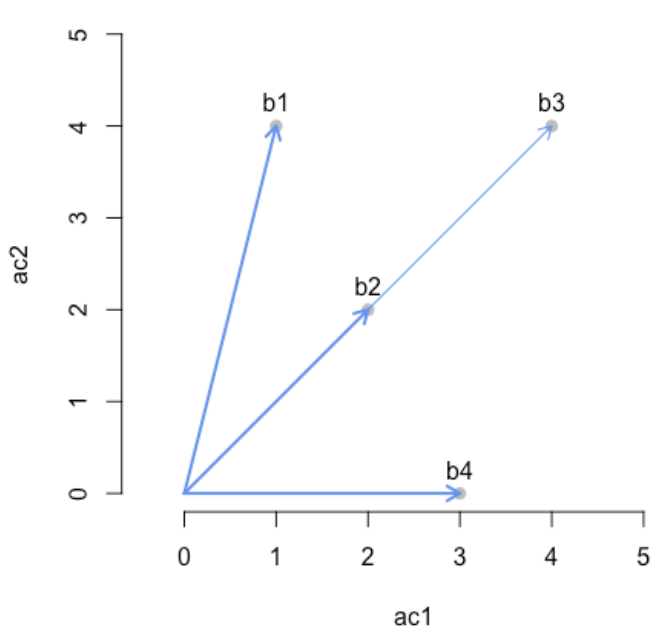
This is a form of
item-item collaborative filtering

Data Similarity

Geometric
Cosine

$$\cos\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\cdot\|\vec{b}\|} = \frac{\sum_i a_i b_i}{\sqrt{\sum a_i^2}\cdot\sqrt{\sum b_i^2}}$$

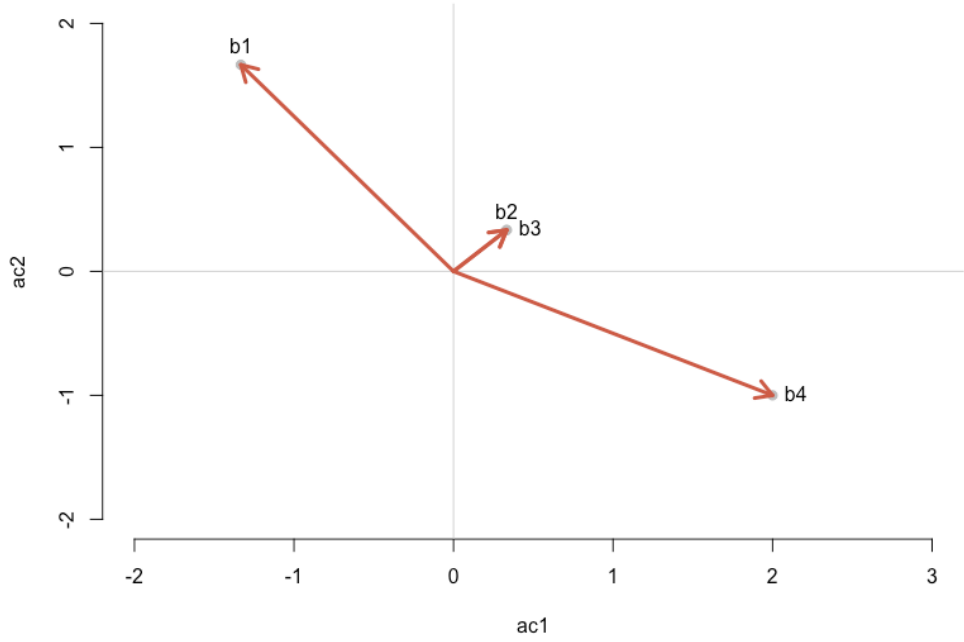
	b1	b2	b3	b4
ac1	1	2	4	3
ac2	4	2	4	0
ac3	2	1	2	0



Statistical
Correlation

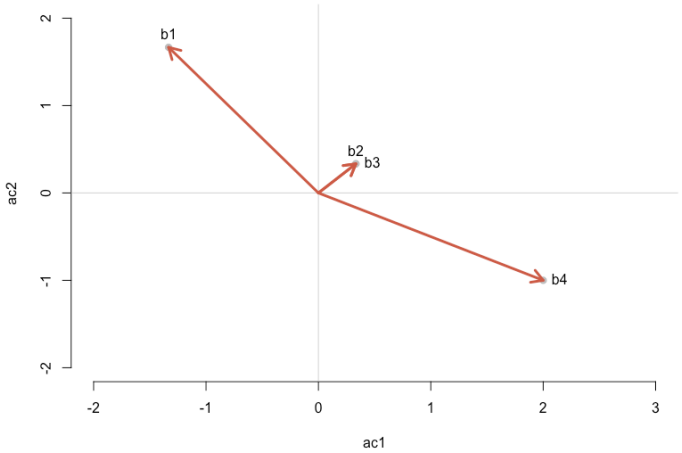
$$cor(a,b) = \frac{\sum_i (a_i-\bar{a})(b_i-\bar{b})}{\sqrt{\sum (a_i-\bar{a})^2}\cdot\sqrt{\sum (b_i-\bar{b})^2}}$$

	b1	b2	b3	b4
ac1	-1.33	0.33	0.33	2
ac2	1.67	0.33	0.33	-1
ac3	-0.33	-0.67	-0.67	-1



Could we use **Correlation** for similarity?

$$cor(a,b) = \frac{\sum_i (a_i - \bar{a})(b_i - \bar{b})}{\sqrt{\sum (a_i - \bar{a})^2} \cdot \sqrt{\sum (b_i - \bar{b})^2}}$$



Correlations puts the average usage at the origin –

cor(…) computes correlation of columns

```
cor(ac_bundles)
```

```
       b1   b2   b3    b4
b1   1.00 0.19 0.19 -0.76
b2   0.19 1.00 1.00  0.50
b3   0.19 1.00 1.00  0.50
b4  -0.76 0.50 0.50  1.00
```

Mean centering bundle *b* asks the question:
"How many times did each account use b, relative to the average account"

```
bundle_means <- apply(ac_bundles, 2, mean)
   b1   b2   b3   b4
 2.33 1.67 2.67 1.00
```

Means of each bundle (columns)

```
bundle_means_matrix <- t(replicate(nrow(ac_bundles), bundle_means))
        b1   b2   b3 b4
 [1,] 2.33 1.67 3.33  1
 [2,] 2.33 1.67 3.33  1
 [3,] 2.33 1.67 3.33  1
```

Repeated means of each column vector

```
ac_bundles_mc_b <- ac_bundles - bundle_means_matrix
        b1    b2    b3   b4
ac1 -1.33  0.33  0.33    2
ac2  1.67  0.33  0.33   -1
ac3 -0.33 -0.67 -0.67   -1
```

Column-mean-centered data matrix

```
    b1 b2 b3 b4
ac1  1  2  4  3
ac2  4  2  4  0
ac3  2  1  2  0
```

original data matrix

***Correlation**: cosine of mean-centered vectors (centered to each item)*

```
cor_sim <- cosine(ac_bundles_mc_b)
       b1    b2    b3    b4
b1   1.00  0.19  0.19 -0.76
b2   0.19  1.00  1.00  0.50
b3   0.19  1.00  1.00  0.50
b4  -0.76  0.50  0.50  1.00
```

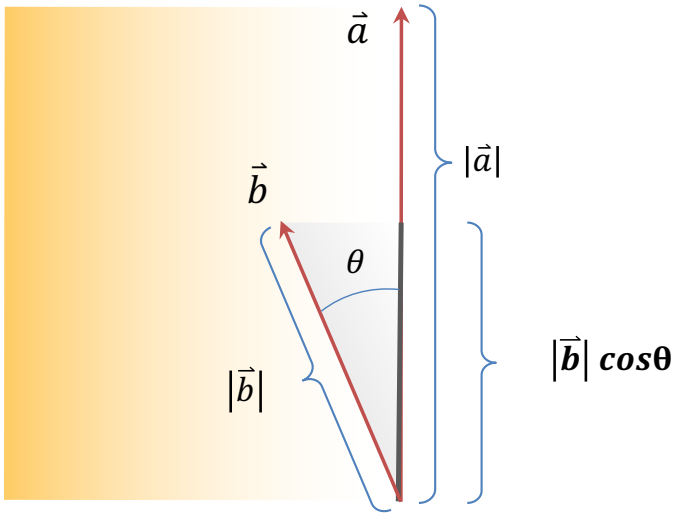
	1st	2nd	3rd
Bundle 1	B3	B2	B4
Bundle 2	B3	B1	B4
Bundle 3	B2	B1	B4
Bundle 4	B2	B3	B1

The Geometry of Similarity

The Dot Product $\vec{a} \cdot \vec{b} = |\vec{a}||\vec{b}|cos\theta$

Recall:

$$\cos\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\cdot\|\vec{b}\|}$$



Projection of a vector onto another

geometrically *or* ***statistically***

$|\vec{a}||\vec{b}|cos\theta$

The product of length of $\vec{a}$ *in the direction of* $\vec{b}$

$\sum(a_i - \bar{a})(b_i - \bar{b})$

The product of deviations of a and b from their means

captures agreement in all their component directions

captures agreement in dispersion between columns

Correlation: statistical view of similarity

		x_i	y_i
Dispersion	*Variability (Sum of Squares)*	$SS_x = \sum_{i=1}^{n}(x_i - \bar{x})^2 = \sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})$	$SS_y = \sum_{i=1}^{n}(y_i - \bar{y})(y_i - \bar{y})$
	Variance (Mean Squares)	$s_x^2 = \frac{SS_x}{df} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})}{n-1}$	$s_y^2 = \frac{SS_y}{df} = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(y_i - \bar{y})}{n-1}$

Relationship

Covariance

Units: *(units of x)(units of y)*

$$\text{cov}_{xy} = \frac{SS_{xy}}{df} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{n-1}$$

Correlation

Units: *unit free*

$-1 < r_{xy} < 1$

$$r_{xy} = \underbrace{\frac{\text{cov}_{xy}}{s_x s_y}}_{\text{fully standardized covariance}} = \frac{SS_{xy}}{\sqrt{SS_x SS_y}} = \frac{\sum(x - \bar{x})(y - \bar{y})}{\sqrt{(x - \bar{x})^2 (y - \bar{y})^2}}$$

= *normalized agreement in dispersion*

$-1 < r_{xy} < 1$

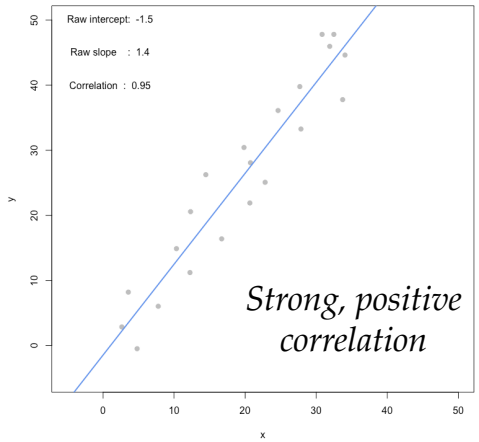
dimensions

cases

	x	y
1	2.65	2.84
2	10.34	14.88
3	20.75	28.08
4	27.69	39.78
5	31.90	45.97
…	…	…
21	30.83	47.81

```
cor(pts)
  [1] 0.95
```

Correlation gives us a normalized similarity, adjusting within dimensions



Cosine vs. Correlation

Zero value columns

*Cosine **cannot compare** zero columns*

```
cosine(c(0,0), c(3,2))

      [,1]
[1,]  NaN
```

*Correlation **cannot compare** zero columns*

```
cor(c(0,0), c(3,2))

[1] NA
Warning message: the standard deviation is zero
```

Identically rated items (constant columns)

```
scores <- rbind(c(1,5,3,2), c(2,4,3,9), c(4,7,3,2))
     [,1] [,2] [,3] [,4]
[1,]    1    5    3    2
[2,]    2    4    3    9
[3,]    4    7    3    2
```

*Cosine **can use** identically rated items*

```
cosine(scores)

          [,1] [,2] [,3] [,4]
[1,] 1.00 0.94 0.88 0.65
[2,] 0.94 1.00 0.97 0.67
[3,] 0.88 0.97 1.00 0.80
[4,] 0.65 0.67 0.80 1.00
```

*Correlation **cannot use** identically rated items*

```
cor(scores)

      [,1]  [,2] [,3]  [,4]
[1,]  1.00  0.79   NA -0.19
[2,]  0.79  1.00   NA -0.76
[3,]    NA    NA    1    NA
[4,] -0.19 -0.76   NA  1.00

Warning message:
 the standard deviation is zero
```

Cosine vs. Correlation

Shift Invariance

*Cosine **varies with shift** in data!*

```
cosine(b1, b2)

       [,1]
  [1,] 0.87

cosine(b1 + 2, b2)
       [,1]
  [1,] 0.94
```

*Correlation is **invariant to shift** in data*

```
cor(b1, b2)

  [1] 0.19

cor(b1 + 2, b2)

  [1] 0.19
```

Scaling Invariance

*Cosine is **invariant to scaling** in data!*

```
cosine(b1, b2)

       [,1]
  [1,] 0.87

cosine(b1 * 2, b2)

       [,1]
  [1,] 0.87
```

*Correlation is **invariant to scaling** in data*

```
cor(b1, b2)

  [1] 0.19

cor(b1 * 2, b2)

  [1] 0.19
```

Peek into Regression

```
library(compstatslib)

pts <- interactive_regression()
# Click on the plot to create data points;
# hit [esc] to stop
```

```
pts
          x         y
1   0.687664  1.865604
2  47.820210 45.294952
3  39.808399 45.840547
4  19.959318 37.220148
.   .    .    .    .
17 11.153543  8.849217
18 30.497375 38.093100
19 19.742782 22.707326
20 33.889764 34.055698
```

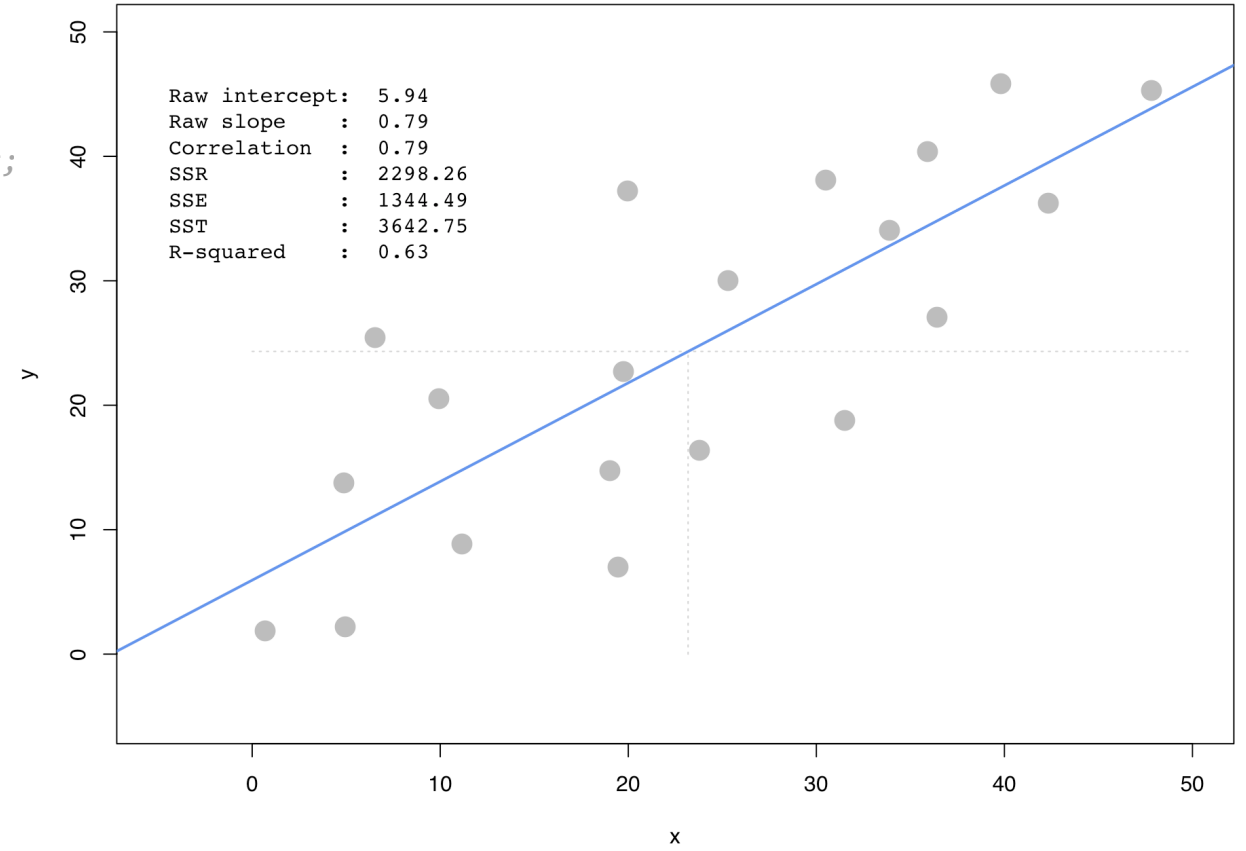
```
var(pts)
          x         y
x 192.3419 152.5317
y 152.5317 191.7239
```

Covariance Matrix
Units: students × dollars

```
cor(pts)
          x         y
x 1.000000 0.794301
y 0.794301 1.000000
```

Correlation Matrix
Units: unit free

$$-1 < r_{xy} < 1$$




```
pts_regr <- lm(pts$y ~ pts$x)
summary(pts_regr)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    5.941      3.836   1.549    0.139
pts$x          0.793      0.143   5.547 2.89e-05 ***
```