# Advanced Permutation Techniques

Mark M. Fredrickson (mfredric@umich.edu)

Computational Methods in Statistics and Data Science (Stats 406)

## Conditioning and Permuting

We noted last time for **the bootstrap**:

- Goal: **estimation**
- **Condition on the sample values**
- Draw **bootstrap replications** with replacement

For **permutation tests**:

- Goal: **hypothesis testing**
- Select a **test statistic** that is **permutation invariant** in its arguments
- **Condition on feature of the sample** and **permute what remains**
- **Two sample problems** are particularly convenient: **condition on the data** and **permute the labels**.

# Models and Parameters

## Incorporating Models

The most basic permutation tests (for two samples),

$$H_0 : F = G$$

But what if we wanted to test a particular relationship between $F$ and $G$ such that

$$H_0 : F(x) = G(h(x; \theta)), \forall x \in (-\infty, \infty)$$

where $h(x; \theta)$ is a function that may depend on a **parameter** $\theta$.

If $H_0$ is true, then $X$ and $h(Y)$ have the **same distribution** and we can **permute the labels** for $X$ and $h(Y)$.

## Shift models

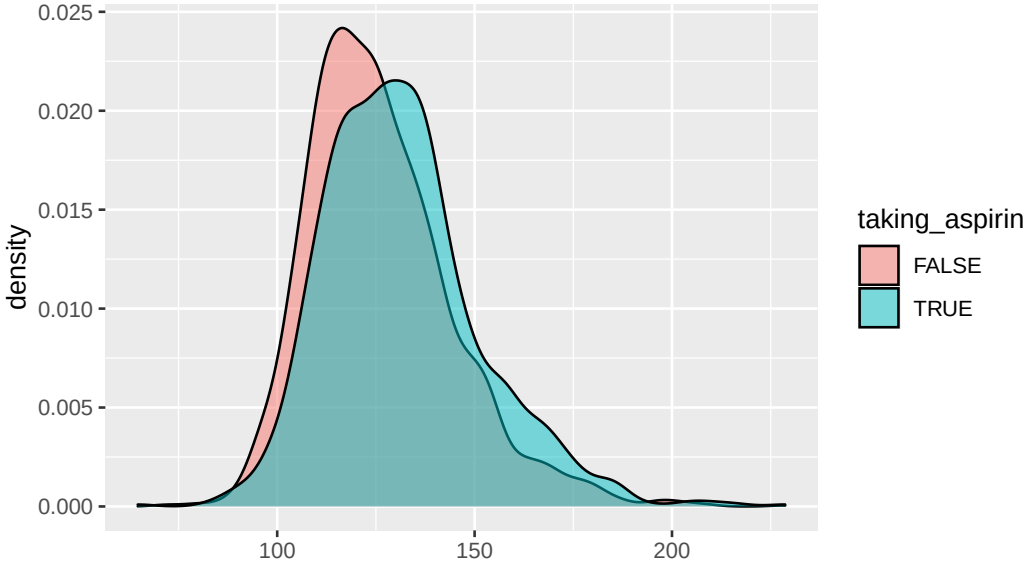
Perhaps the most common $h(x;\theta)$ functions is a **shift model**:

$$h(x;\theta) = x - \theta$$

These tend to make sense when the outcome is measured on a **interval scale** (as compared to a **ratio scale**).

We'll work with the **systolic pressure readings** now, and return to the sys/dia ratio later.

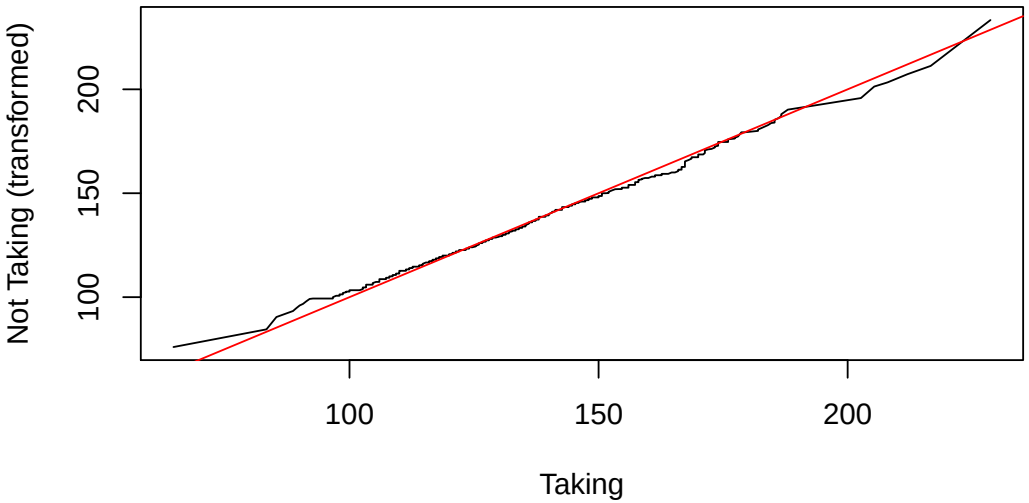
# Systolic BP measurements



Let's create a confidence interval for $h(x;\theta) = x - \theta$ using a KS test.

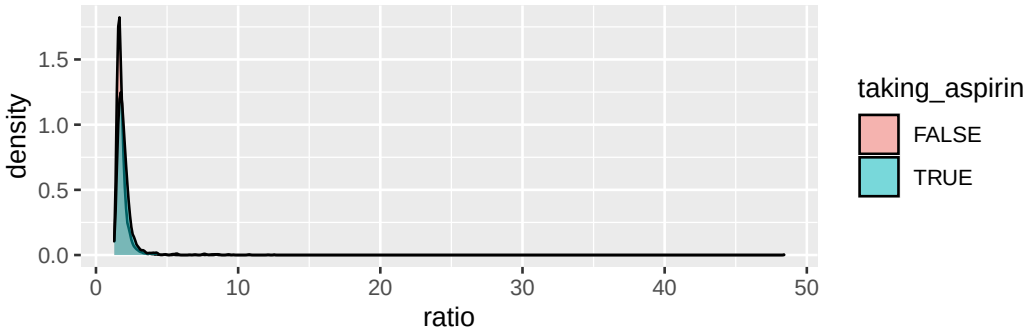
```
> thetas <- seq(-8, 0, length.out = 100)

> ps <- sapply(thetas, function(theta) {
+   with(nhanes, ## creates variables sys_mean, taking_apsirin
+     ks.test(x = sys_mean[taking_aspirin],
+             y = sys_mean[!taking_aspirin] - theta)$p.value)
+ })

> (ci90 <- range(thetas[ps >= 0.1])) ## 90% CI

[1] -5.979798 -4.686869

> (thetahat <- thetas[which.max(ps)])

[1] -5.333333
```

## Aligning the ECDFs at $\hat{\theta}$



# Ratio of Systolic to Diastolic

The **shift model** worked fairly well when looking the systolic BP. Will it work with the **ratio of systolic to diastolic**?



## Some models can be rejected everywhere

Now trying with the ratios of systolic to diastolic:

```
> thetas <- seq(-1, 1, length.out = 100)
> ps <- map_dbl(thetas, function(theta) {
+     with(nhanes,
+          ks.test(x = ratio[taking_aspirin],
+                  y = ratio[!taking_aspirin] - theta)$p.value)
+ })
> any(ps >= 0.01)

[1] FALSE
```

## Another model

**Shift models** are common, but we have lots of flexibility to craft others.

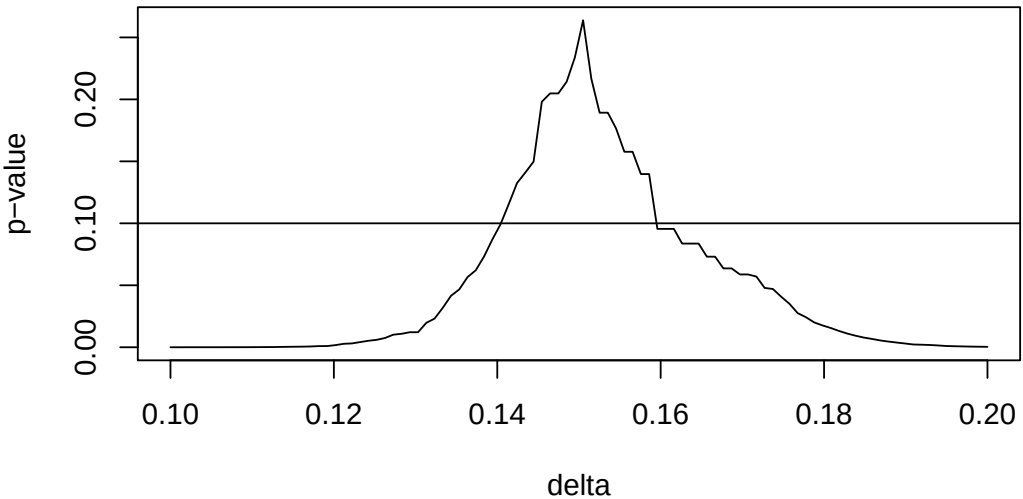
Since the ratio must be a positive number (and generally $> 1$), it makes sense to think about an **exponential parameter**.

Also, perhaps these distributions only really differ in the **upper tail**, so limit investigation there.

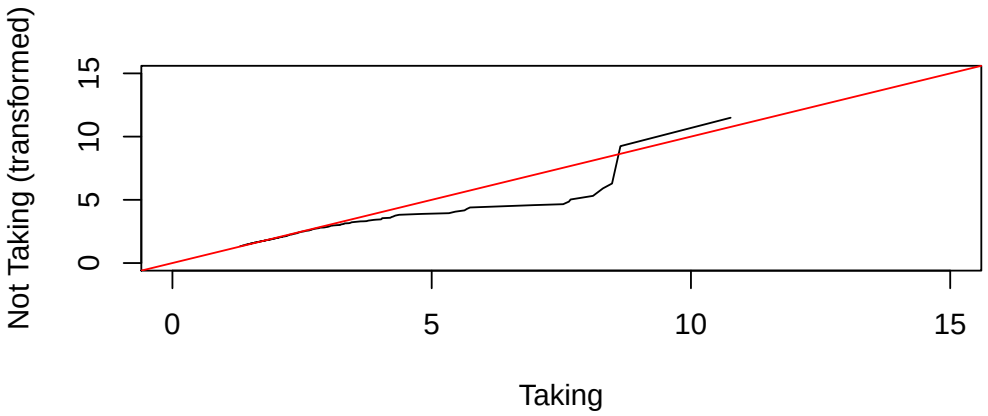
$$h(x) = \begin{cases} x : x \leq 1.2 \\ x^{1+\delta} : x > 1.2 \end{cases}$$

## Implementing in R

```
> deltas <- seq(0.1, 0.2,length.out = 100)
> h <- function(x, delta) { x^(1 + delta * (x > 1.2) )}
> ps <- map_dbl(deltas, function(delta) {
+ with(nhanes, ## creates variables sys_mean, taking_apsirin
+      ks.test(x = ratio[taking_aspirin],
+              y = h(ratio[!taking_aspirin], delta))$p.value)
+ })
> any(ps >= 0.01)

[1] TRUE
```


p−value
0.00
0.10
0.20
0.10
0.12
0.14
0.16
0.18
0.20
delta

## Aligning ECDFs at $\hat{\delta}$



## Two Sample Permutation Tests Summary

- Two samples (sizes $n$, $m$) grouped into $(W_i, Z_i)$
- When $H_0 : F = G$, shuffle the treatment labels to get the distribution of $T(W, Z)$.
- Picking a good $T$ is key. Some options:
  - Mean difference (perm. $t$)
  - Maximum difference of the ECDFs (KS)
  - Sum of ranks in one group (WMW)
- When using ranks (KS, WMW) we can tabulate $T$'s distribution without seeing the data ("distribution free")
- For $H_0 : F(x) = G(h(x))$, transform $h(Y)$ and shuffle the labels. Many options for $h$.

# Discrete Data

## Permutation Tests for Discrete Data

In our notation, $Z_i = I(i \leq n)$ is a **discrete random variable**.

If $W$ is also discrete with levels $1, 2, \ldots$, we can **cross classify** with $Z$ into a table. For simplicity, assume $W_i \in \{0, 1\}$.

| | $Z = 0$ | $Z = 1$ | |
|---|---|---|---|
| $W = 0$ | $A_{00}$ | $A_{01}$ | $\sum I(W_i == 0)$ |
| $W = 1$ | $A_{10}$ | $A_{11}$ | $\sum I(W_i == 1)$ |
| | $m$ | $n$ | $n + m$ |

where

$$A_{ab} = \sum_{i=1}^{n} I(W_i = a) I(Z_i = b)$$

## Fixed and Random Elements

In general, we take $n$ and $m$ to be **fixed values**.

Under the **permutation hypothesis** (i.e. $F = G$), we can relabel all observations, but we always have the same totals (**row totals** from last slide)

$$\sum_{i=1}^{n} I(W_i = 0), \quad \sum_{i=1}^{n} I(W_i = 1)$$

The entries $A_{ab}$, however, **will change** but be constrained by **fixed row and column totals**.

Hypothesis test: is the observed table "extreme" if $F = G$?

# Creating a discrete variable

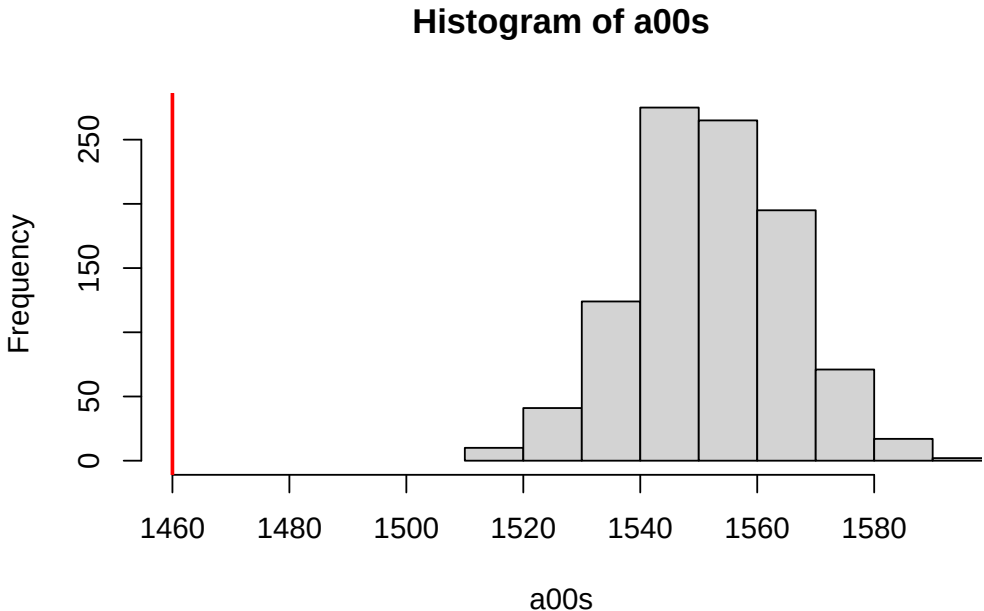
A **systolic** blood pressure of **less than 120** is considered **healthy**.

```
> nhanes$healthy <- nhanes$sys_mean <= 120
> library(xtable)
> print(xtable(table(nhanes$healthy, nhanes$taking_aspirin)))
```

| | FALSE | TRUE |
|---|---|---|
| FALSE | 1460 | 703 |
| TRUE | 1094 | 310 |

## Implementing in R

```
> observed_a00 <- with(nhanes, table(healthy, taking_aspirin))[1,1]
> a00s <- replicate(1000, {
+     newz <- sample(nhanes$taking_aspirin) # permutes labels
+     table(nhanes$healthy, newz)[1, 1]
+ })
> 2 * min(mean(a00s >= observed_a00), mean(a00s <= observed_a00))

[1] 0
```


Histogram of a00s
Frequency
0
50
150
250
1460
1480
1500
1520
1540
1560
1580
a00s

# Fisher's Exact Test

This procedure is known as **Fisher's exact test**. It is a **distribution free** test (additionally, the PDF is **hypergeometric**, but we can always permute to get the right answer).

It generalizes to more than 2 categories or more than 2 samples.

```
> fisher.test(nhanes$healthy, nhanes$taking_aspirin)$p.value

[1] 1.207e-11
```

## Two sample for ordered data

While Fisher's test easily generalizes for **unordered data** with more than two categories, how can we use **ordering of categories** in our data?

Example: recall the study of students located in San Antonio and Phoenix. Teachers were asked about each student if students had "good attention span, completes chores or homework." and could answer one of:

- Not true
- Somewhat true
- Certainly true

## Responses by city

| | (1) Not true | (2) Somewhat true | (3) Certainly true |
|---|---|---|---|
| Phoenix, AZ | 135 | 290 | 426 |
| San Antonio, TX | 233 | 492 | 545 |

Our strategy will be to replace "Not true", "Somewhat true", and "Certainly true" with numerical scores, and then compare across the cities (Cochran-Armitage test).

## Filling in numeric values

One easy method would be to fill the values 1,2,3 (or 0, 1, 2) instead of the categories, but **is "certainly true" twice as much as "not true"**?

Suppose there a **latent variable** $Y$ such that if

- $Y \leq \theta_1$, the student does not complete homework;
- $\theta_1 < Y \leq \theta_2$, the student completes homework sometimes;
- $\theta_2 < Y$, the student always completes homework.

If we could observe $Y$, we could use it's numerical score, or better yet, **its rank**.

# Midranks

We don't get to observe $Y$ or the $\theta$ parameters, but it must be the case that **all students in the first column have lower $Y$ than all students in the second column**, and likewise for the second and third column.

Additionally, our best guess for all students in the same category, will be the **average rank** within that category. Then for each category we have **midrank** score of

$$\{\#\text{ in lower categories}\} + \frac{\{\#\text{ in this category}\}}{2}$$

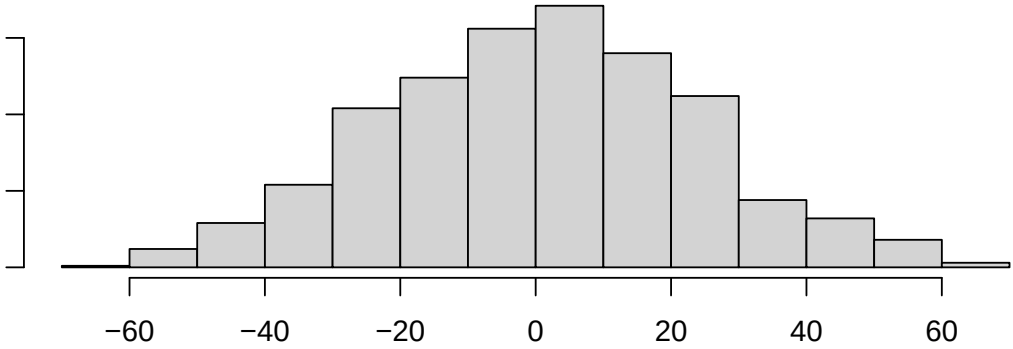
```
> col_totals <- colSums(hwtab)
> (midranks <-  col_totals / 2  + cumsum(c(0, col_totals))[1:3])
```

```
      (1) Not true  (2) Somewhat true (3) Certainly true
               184                759               1636
```

## Difference of means on midranks

Now that we have a numerical score to use for each observation, we one of our existing permutation tests, such as the **difference of means** test.

```
> w <- midranks[gamoran$B4Y]
> (obs_stat <- mean(w[gamoran$PH.AZ], na.rm = TRUE)
+     - mean(w[!gamoran$PH.AZ], na.rm = TRUE))

[1] -76.91

> null_distribution <- replicate(1000, {
+     newZ <- sample(gamoran$PH.AZ)
+     mean(w[newZ], na.rm = TRUE) - mean(w[!newZ], na.rm = TRUE)
+ })
```




```
> 2 * min(mean(obs_stat <= null_distribution),
+         mean(obs_stat >= null_distribution))

[1] 0
```

## Models for discrete data

Note that after replacing the category labels with a numeric score $W$ (e.g. the midrank, quantiles from a Normal distribution), we are testing

$$H_0 : F_0(x) = F_1(x) \text{ vs. } F_0(x) \neq F_1(x)$$

where $F_0$ is the distribution for $W$ when $Z = 0$ and $F_1$ is the distribution when $Z = 1$.

Previously, saw that we could test **models based on functions $h$** such that

$$H_0 : F_0(x) = F_1(h(x)) \text{ vs } H_1 : F_0(x) \neq F_1(h(x))$$

For discrete data, we come up with $h$ functions that operate on cells of the table. (e.g., move 10 "somewhat true" students from San Antonio to the "Not true" column).

# Independence Tests

# Independence Tests

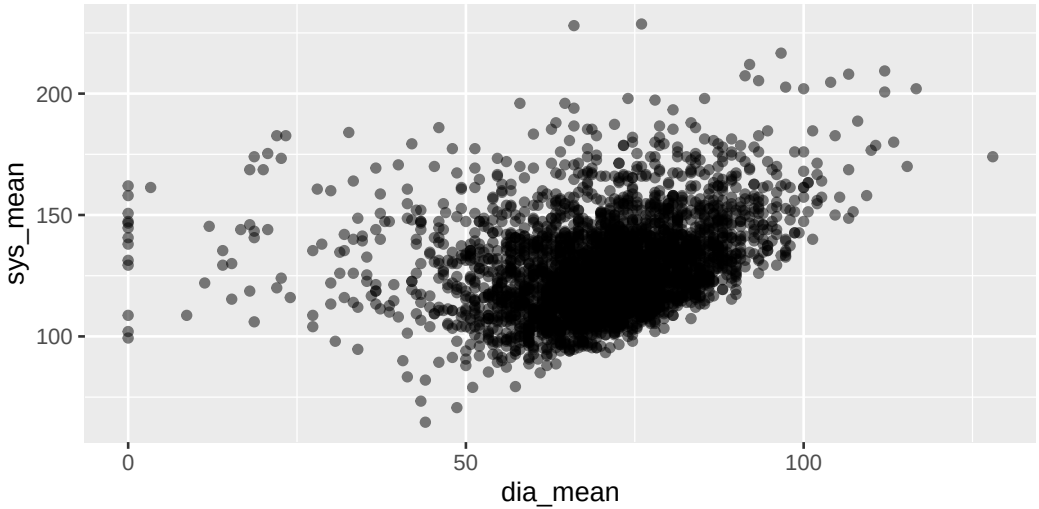
You may have encountered Fisher's exact test as an example of an **independence test**.

Suppose that instead of **two samples** we had a **single sample of pairs** $(W_i, Z_i), i = 1, \ldots, n$, IID.

Under the hypothesis that $F_{WZ} = F_W F_Z$ (independent), we could arbitrarily permute all the $Z$ values and any statistic $T(W, Z)$ would have the same distribution.

We can test this hypothesis with a **test statistic for pairs**.

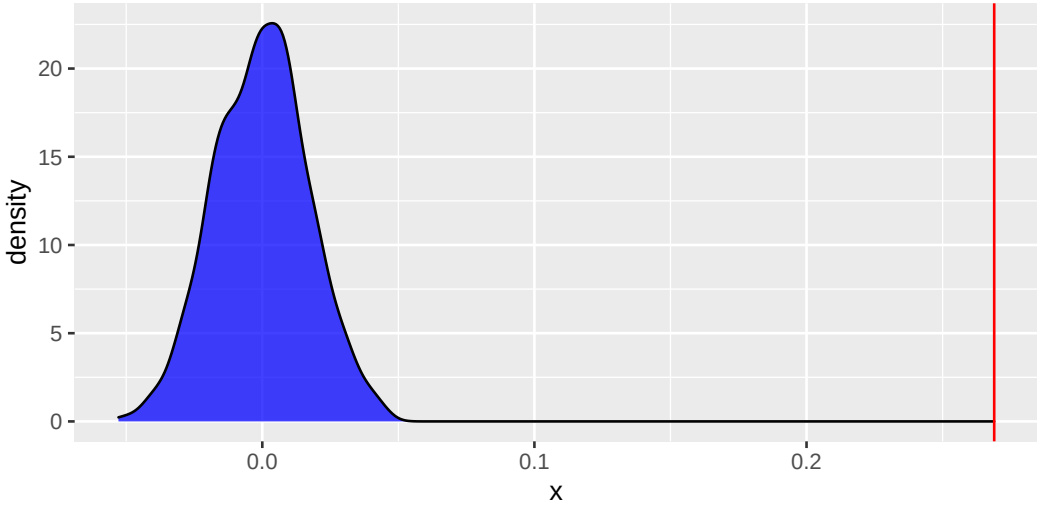
# Systolic and Diastolic BP



## Test independence of systolic and diastolic BP

**Sample correlation** is a way to summarize the **linear relationship** between two variables.

```
> observed_cor <- with(nhanes, cor(sys_mean, dia_mean))
> cors <- replicate(1000, {
+   shuffled_dia <- sample(nhanes$dia_mean)
+   cor(nhanes$sys_mean, shuffled_dia)
+ })
> 2 * min(mean(cors >= observed_cor), mean(cors <= observed_cor))

[1] 0
```


density
20
15
10
5
0
0.0
0.1
0.2
x

## More on independence tests

Many other options for test statistics:

- Generalizations of correlation to non-linear dependence (see Rizzo, ch 8)
- Distribution free methods based on ranks
- Chi-squared and $G^2$ type statistics for categorical data (see Agresti's *Categorical Data Analysis* book)

Note: `cor.test` in R does not implement a permutation test – it uses an assumption that the variables are jointly Normal.

## Connections to randomized trials

One kind of variable that has a special form is when $Z$ is **randomly assigned** by a researcher from a known process.

For example, the students in Phoenix and San Antonio where assigned to either participate in a social capital building program (**treatment**) or not (**control**).

From a causal perspective, we want to ask "what could have been different if students had been assigned to a different condition?" One answer is that we would see the same response for all subjects:

$$[Y_i \mid Z_i = 1] = [Y_i \mid Z_i = 0]$$

(note: this is stronger than just saying the treatment and control subjects have the same distribution)

We call this the **"sharp null hypothesis of no effect"**.

# Testing SNHNE

Suppose the sharp null hypothesis is true.

For a given test statistic that compares the treatment and control groups, we want to ask, "Is the observed value extreme if the sharp null is true?"

According to the sharp null, we would have seen exactly the same response for subjects, regardless of their treatment assignments, so we **shuffle treatment assignments** to get the distribution of the test statistic under the null.

We assign a fixed pool of people to fixed sized treatment and control groups, this is exactly a **permutation test**.

For other kinds of randomization mechanisms, we need to **respect the distribution of** $Z$.

## Clustered assignment

So far, we have been treating the students in the Gamoran et al. study as if they are samples from Phoenix and San Antonio.

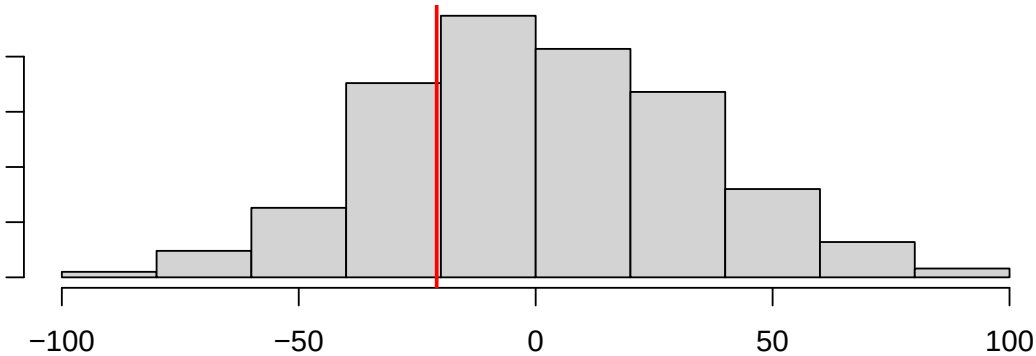
More importantly, they are **clustered** in schools and **schools were assigned** to either participate in a social capital building program or not.

```
> school_assignments <- group_by(gamoran, Y1SCHOOLID) %>%
+     summarize(z = first(z))
> table(school_assignments$z)

FALSE  TRUE
   26    26
```

## Permuting by school

We've already computed midranks for all subjects for whether the student completed homework. Let's get the null distribution based on the clustered assignment:

```
> (w_clus <- mean(w[gamoran$z], na.rm = TRUE) - mean(w[!gamoran$z], na.rm =

[1] -20.89

> null_distribution_cluster <- replicate(1000, {
+     newZ <- sample(school_assignments$z)
+     names(newZ) <- school_assignments$Y1SCHOOLID
+     studentZ <- newZ[as.character(gamoran$Y1SCHOOLID)]
+     mean(w[studentZ], na.rm = TRUE) - mean(w[!studentZ], na.rm = TRUE)
+ })
```




```
> 2 * min(mean(w_clus <= null_distribution_cluster),
+         mean(w_clus >= null_distribution_cluster))

[1] 0.512
```

## Re-randomizing the wrong way

How much would we hurt ourselves if we thought that **students had been individually assigned** to the social capital program?

```
> null_distribution_wrong <- replicate(1000, {
+     newZ <- sample(gamoran$z)
+     mean(w[newZ], na.rm = TRUE) - mean(w[!newZ], na.rm = TRUE)
+ })
> 2 * min(mean(w_clus <= null_distribution_wrong),
+         mean(w_clus >= null_distribution_wrong))
```

```
[1] 0.414
```

## Independence and Randomization Tests Summary

- Setting: IID pairs of $(W, Z)$ or randomly assigned $Z$ and outcome $W$.
- Hypothesis test: $H_0 : F(w, z) = F(w)F(z)$ or sharp null of no effect
- Procedure:
  - IID Pairs: shuffle either $W$ and $Z$ and compute test statistic
  - Random assignment: follow assignment procedure for $Z$ and compute test statistic
- The usual technique of using a model function $h$ also applies such that $H_0 : h(W)$ and $Z$ are independent
- Be careful to reflect the true randomization procedure.

# Multivariate Tests

Suppose now that we have **two samples** of **random vectors** (with $p$ components):

$$X_i = (X_{i1}, X_{i2}, \ldots, X_{ip})', \quad i = 1, \ldots, n$$
$$Y_j = (Y_{j1}, Y_{j2}, \ldots, Y_{jp})', \quad j = 1, \ldots, m$$

Again, we want to test same distributions:

$$H_0 : F(x_1, x_2, \ldots, x_p) = G(y_1, y_2, \ldots, y_p), \quad \forall x, y \in \mathbb{R}^p$$

## What kind of test statistic?

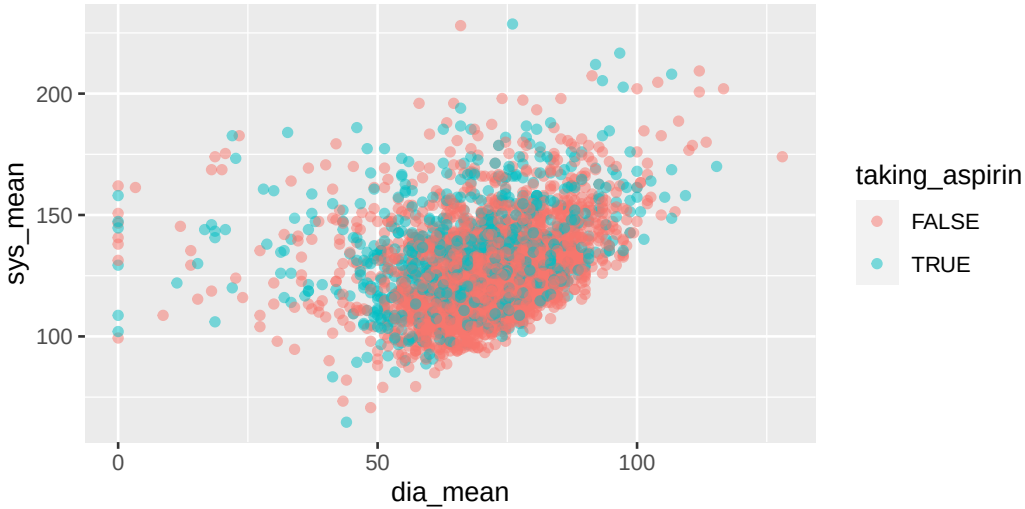
When $p$ is large (particularly when $p > n$), it can be difficult to find a way to compare the two samples.

Option 1: **combine within** each $X$ and $Y$ to create summary score. (e.g., replace $X$ with the mean of all the components).

We've already seen this when we took a **ratio** of systolic and diastolic mean.

Option 2: Think about observations as **points in space**.

# Joint Distribution of Systolic and Diastolic



## Nearest Neighbor Test

Let's use the "joint sample" notation:

$$(Z_i, W_{i1}, W_{i2}, \ldots, W_{ip})$$

Idea: For each point (in both samples), find the **point that is closet** in distance:

$$N(i) = \text{argmin}_j \sqrt{(X_{i1} - X_{j1})^2 + (X_{i2} - X_{j2})^2 + \cdots + (X_{ip} - X_{jp})^2}$$

Write the group label of the neighbor as $Z_{N(i)}$.

Our test statistic counts the number of neighbors **in the same sample**:

$$T(Z, W) = \sum_{i=1}^{n} I(Z_i = Z_{N(i)})$$

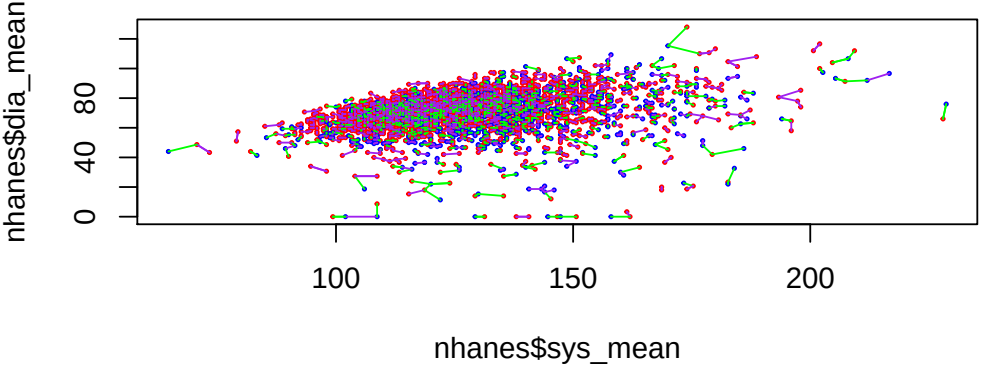
## Computing Distances and Neighbors

```
> sys_dia_dist <- as.matrix(dist(nhanes[, c("sys_mean", "dia_mean")]))
> sys_dia_dist[1:5, 1:5]

      1     2     3     4     5
1  0.00 46.43 30.40 27.73 45.38
2 46.43  0.00 25.73 32.28 20.67
3 30.40 25.73  0.00  6.60 15.33
4 27.73 32.28  6.60  0.00 20.54
5 45.38 20.67 15.33 20.54  0.00
```

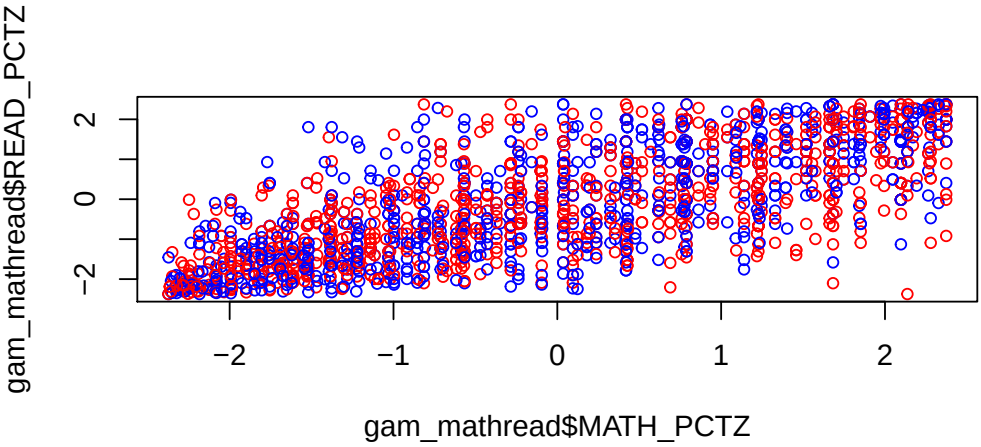
```
> diag(sys_dia_dist) <- Inf # can't be closest to ourselves
> nearest_neighbors <- apply(sys_dia_dist, 1, which.min)
> nearest_neighbors[1:5]

   1    2    3    4    5
 849 1974  424  969   72
```

```
> teststat <- function(z, nn) {
+     sum(z == z[nn])
+ }
> observed_t <- teststat(nhanes$taking_aspirin, nearest_neighbors)
> ts <- replicate(1000, {
+     z <- sample(nhanes$taking_aspirin)
+     teststat(z, nearest_neighbors)
+ })
> 2 * min(mean(observed_t <= ts), mean(observed_t >= ts))

[1] 0.068
```

# Visualizing Statistic



# Nearest neighbor for math and reading outcomes



```
> math_read_dist <- as.matrix(
+     dist(gam_mathread[, c("READ_PCTZ", "MATH_PCTZ")]))
> diag(math_read_dist) <- Inf # can't be closest to ourselves
> mrnn <- apply(math_read_dist, 1, which.min) # row wise apply
> mrnn_obs_t <- teststat(gam_mathread$z, mrnn)
```

```
> nn_cluster <- replicate(1000, {
+     newZ <- sample(school_assignments$z)
+     names(newZ) <- school_assignments$Y1SCHOOLID
+     studentZ <- newZ[as.character(gam_mathread$Y1SCHOOLID)]
+     teststat(studentZ, mrnn)
+ })
> 2 * min(mean(mrnn_obs_t <= nn_cluster),
+         mean(mrnn_obs_t >= nn_cluster))

[1] 0.78
```

# Advanced Permutation Techniques Conclusion

We saw several more advanced **permutation methods**

- Using **models** to make conditioning possible.
- Permuting **tables** or other **discrete data**, replacing discrete values with numbers
- Testing **independence** between two variables by permuting one
- Connection between **permutation and randomization**
- **Multivariate tests** using distances or graphs

Common aspect is that we can use Monte Carlo to draw from the possible permutations/randomizations to evaluate a statistic under the null hypothesis. Many of these methods can be combined.