DATA2002

What can we do when ANOVA assumptions fail?

Garth Tarr

THE UNIVERSITY OF SYDNEY

What happens when assumptions fail?

Kruskal-Wallis test

When ANOVA assumptions might be violated

Assumptions underlying ANOVA (and related methods)

We have learnt much about the comparison of several samples:

- The F-test (with p-value computed using the F-distribution)
- **contrasts** (and associated methods, all based on the t-distribution)
- **multiple comparisons**:
  - Bonferroni
  - Tukey
  - Scheffé

However, underlying all of these are the assumptions that

- each sample is from a **normal population**;
- all **population variances are equal**.
  - so all populations are identical up to possible location shifts

What do we do if these assumptions are not reasonable?

Possible violations

- There are various ways the assumptions might be violated:
  - the normality might be ok, but **equal variances** might not be;
  - the normality might **not** be ok, but the "identical up to location shifts" assumption might be ok.
- There are a few tools we can appeal to:
  - simulation
  - resampling (together with *conditioning*).

Relaxing the equal variance assumption

Wolf River data

- The data in the file `wolfriver.csv` contains 30 measurements (10 at 3 depths: Bottom = 1; Middepth = 2; Surface = 3) on each of two chemicals, `Aldrin` and `HCB`.
    - these were taken downstream from an abandoned dump site near the Wolf River in Tennessee (Jaffe, Parker, and Wilson, 1982).
- It was believed the concentration might not be constant across different depths.
- We shall be *mainly* interested in each chemical separately, although considering them jointly is also a bit interesting...

```
library(tidyverse)
wolf = read_csv("https://raw.githubusercontent.com/DATA2002/data/master/wolfriver.csv")
glimpse(wolf)
```

```
## Rows: 30
## Columns: 3
## $ Aldrin <dbl> 3.08, 3.58, 3.81, 4.31, 4.35, 4.40, 3.67, 5…
## $ HCB    <dbl> 3.74, 4.61, 4.00, 4.67, 4.87, 5.12, 4.52, 5…
## $ Depth  <dbl> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 2, 2, 2, 2, 2…
```

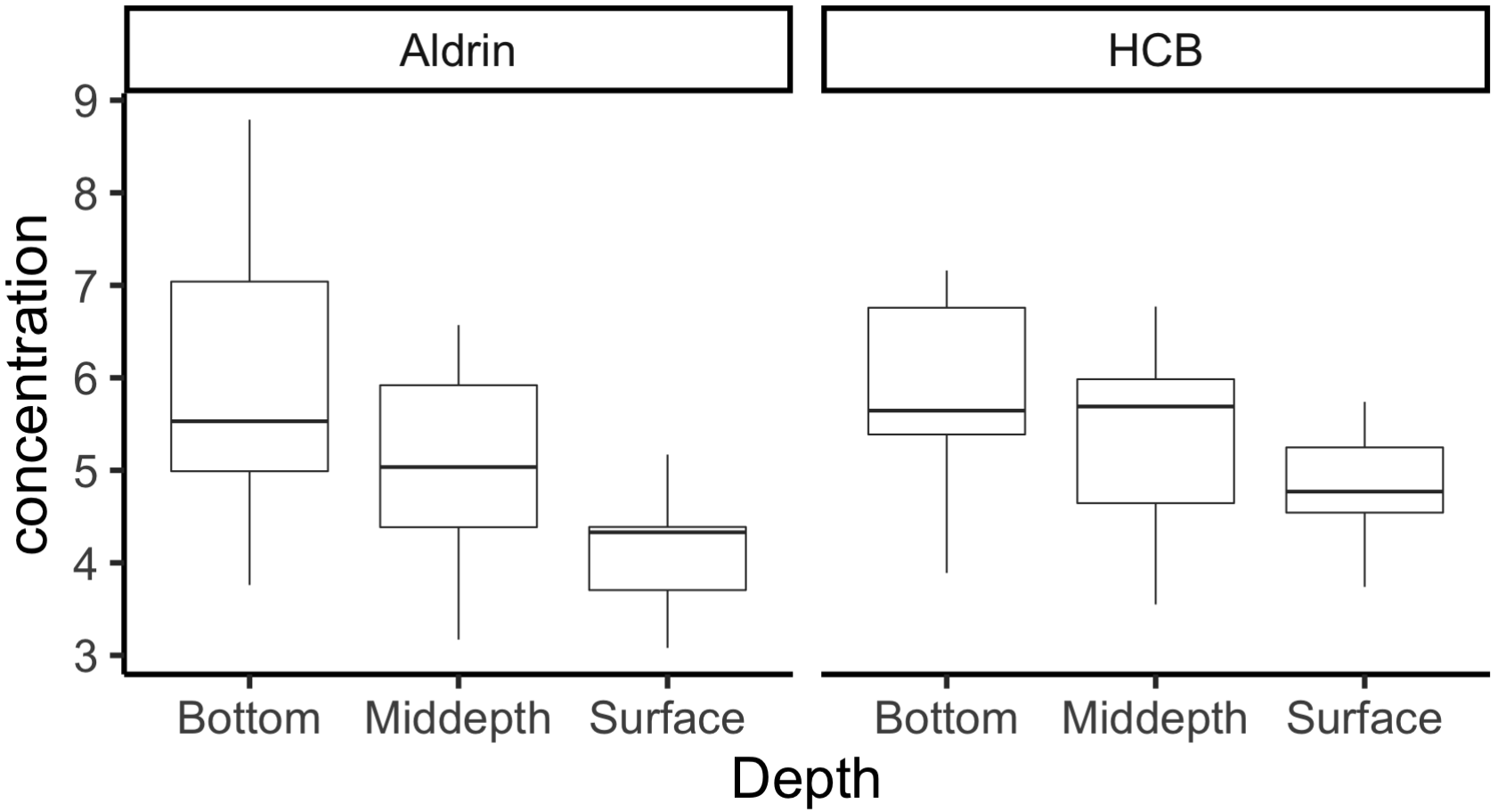
```
wolf = wolf %>% mutate(
  Depth = case_when(
    Depth == 1 ~ "Bottom",
    Depth == 2 ~ "Middepth",
    Depth == 3 ~ "Surface"
  )
)
```

```
wolf %>% count(Depth)
```

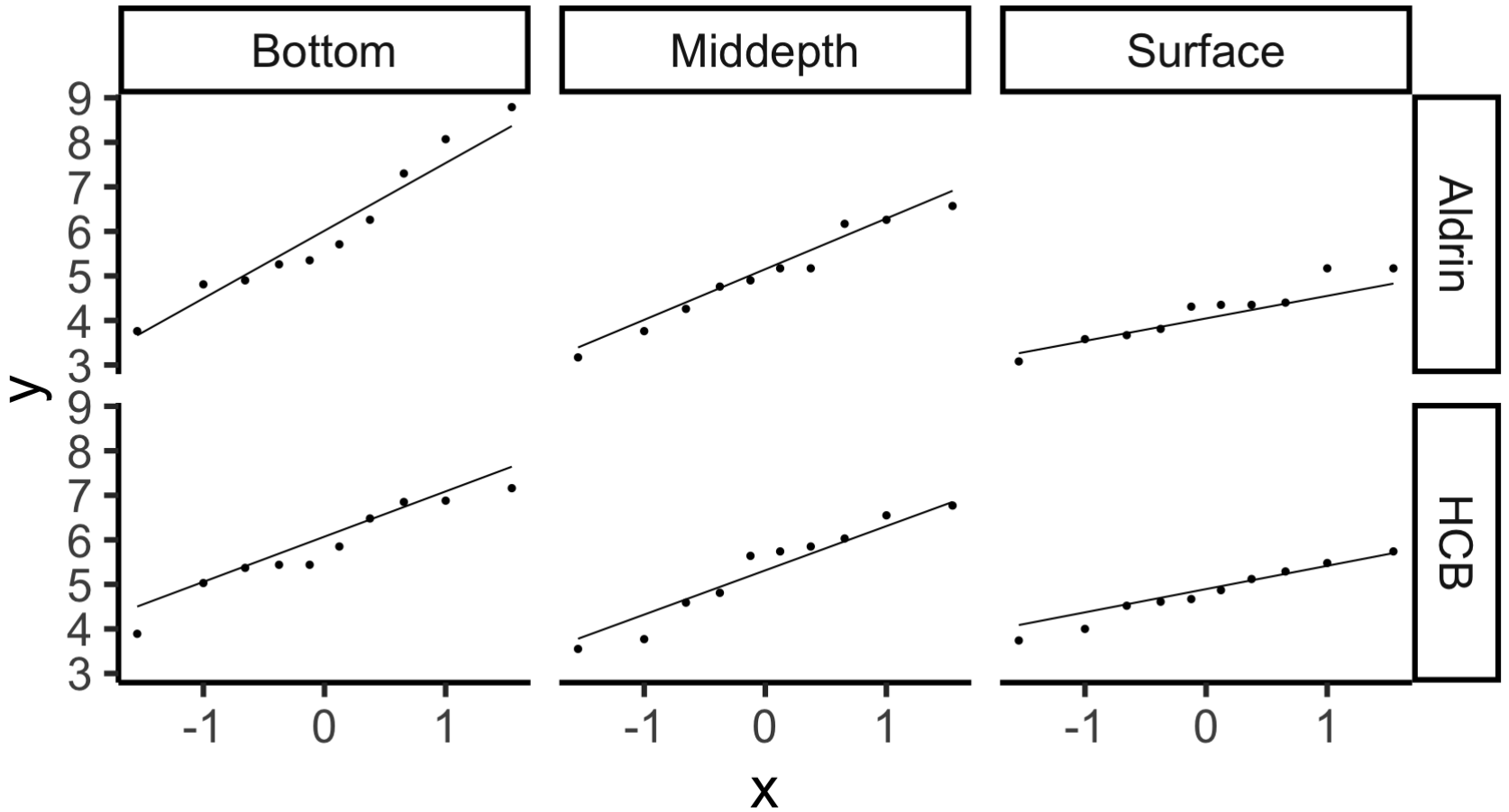
```
## # A tibble: 3 × 2
##   Depth         n
##   <chr>     <int>
## 1 Bottom       10
## 2 Middepth     10
## 3 Surface      10
```

```
wolf_long = wolf %>%
  gather(key = "chemical",
         value = "concentration", -Depth)
```

```
ggplot(wolf_long, aes(x = Depth, y = concentration)) +
  geom_boxplot() + facet_wrap(~chemical)
```



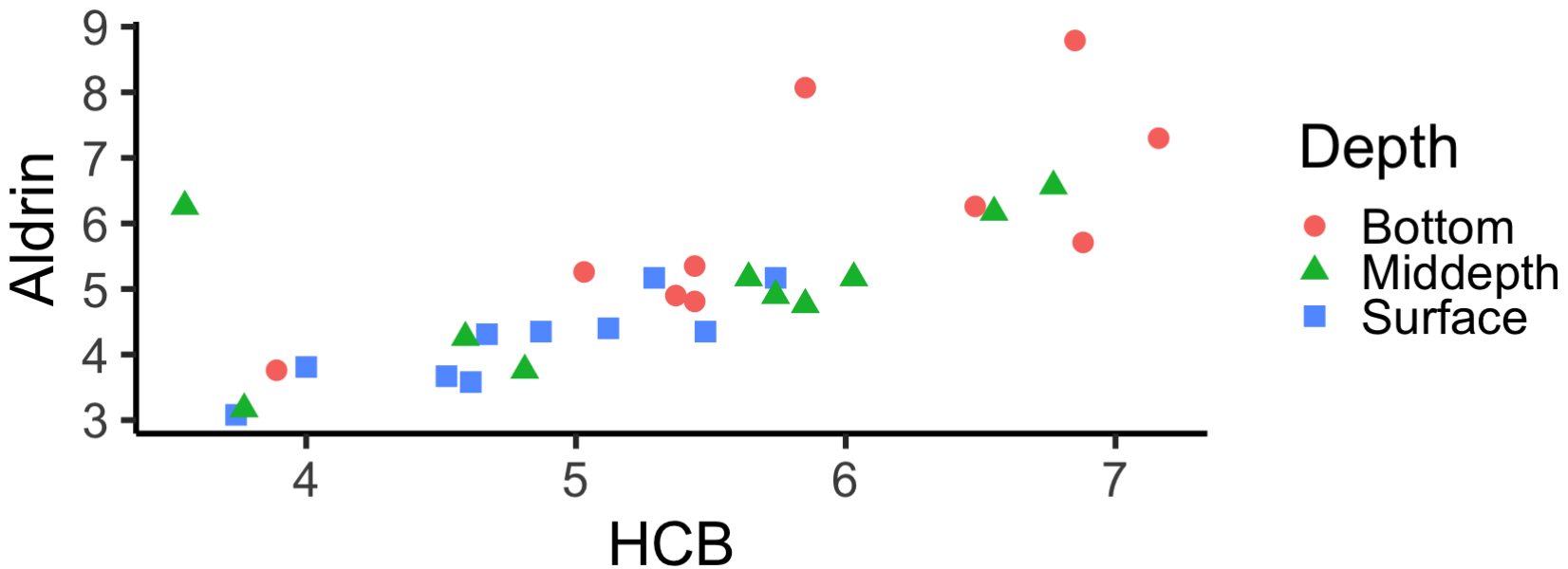
```
ggplot(wolf_long, aes(sample = concentration)) +
  geom_qq() + geom_qq_line() + facet_grid(chemical ~ Depth)
```



Assumptions?

- *Both sets* of boxplots suggest a different spread in each group
  - normality is probably ok (points close to line in QQ plots)
- A *scatterplot* (tracking both chemicals together) reveals something interesting:

```
ggplot(wolf) + aes(x = HCB, y = Aldrin, shape = Depth, colour = Depth) +
  geom_point(size = 5)
```



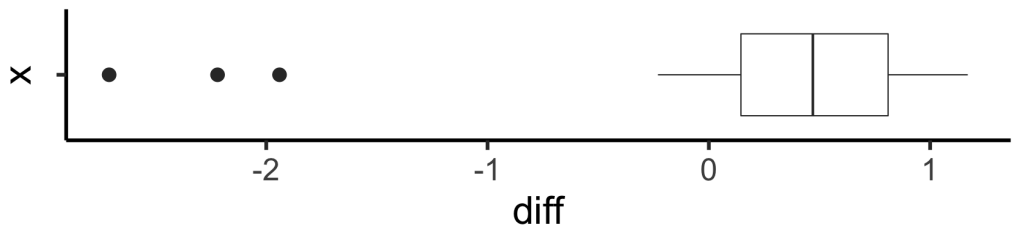
Outliers ?!

- There are 3 possible "outliers" (actually *bivariate* outliers, really):

```
wolf %>% arrange(HCB-Aldrin)
```

```
## # A tibble: 30 × 3
##    Aldrin   HCB Depth
##     <dbl> <dbl> <chr>
##  1   6.26  3.55 Middepth
##  2   8.07  5.85 Bottom
##  3   8.79  6.85 Bottom
##  4   5.26  5.03 Bottom
##  5   7.3   7.16 Bottom
##  6   5.35  5.44 Bottom
##  7   5.17  5.29 Surface
##  8   3.76  3.89 Bottom
##  9   3.81  4    Surface
## 10   6.57  6.77 Middepth
## # … with 20 more rows
```

```
wolf %>% mutate(
  diff = HCB - Aldrin
) %>%
  ggplot() + aes(x = "", y = diff) +
  geom_boxplot(outlier.size = 5) +
  coord_flip()
```



Relaxing the "common variance" assumption: all pairwise comparisons

- We could consider assuming normality, but dropping the "common variance" assumption.
- A simple way to do so is to consider all pairwise **Welch tests**, and apply a Bonferroni correction.
- Recall **Welch tests** only assume that each sample is normal, with possibly different variances σ_X^2 and σ_Y^2 and different means, and all random variables are independent.

$$T = \frac{\bar{X} - \bar{Y}}{\sqrt{\frac{S_X^2}{m} + \frac{S_Y^2}{n}}},$$

  which is **approximately** $t_{d^*(m,n,\sigma_X,\sigma_Y)}$ under H_0, for a known function $d^*(\dots)$.
- The **Welch tests** is the what R defaults to in the `t.test()` function.

Welch test pairwise comparisons (unadjusted)

Middepth vs Surface

```
t.test(wolf$Aldrin[wolf$Depth=="Middepth"],wolf$Aldrin[wolf$Depth=="Surface"])$p.value
```

```
## [1] 0.06053252
```

Middepth vs Bottom

```
t.test(wolf$Aldrin[wolf$Depth=="Middepth"],wolf$Aldrin[wolf$Depth=="Bottom"])$p.value
```

```
## [1] 0.119901
```

Surface vs Bottom

```
t.test(wolf$Aldrin[wolf$Depth=="Surface"],wolf$Aldrin[wolf$Depth=="Bottom"])$p.value
```

```
## [1] 0.005471484
```

- Since we are doing 3 pairwise comparisons, we mutliply the "unadjusted" p-values by 3 to get "adjusted-for-multiplicity" p-values.
- The smallest of these can be used as a test that all (population) means are equal:

```
t.test(wolf$Aldrin[wolf$Depth=="Surface"], wolf$Aldrin[wolf$Depth=="Bottom"])$p.value
```

```
## [1] 0.005471484
```

```
3 * t.test(wolf$Aldrin[wolf$Depth=="Surface"], wolf$Aldrin[wolf$Depth=="Bottom"])$p.value
```

```
## [1] 0.01641445
```

- This is a perfectly valid p-value for testing the "global" or "overall" hypothesis that all means are equal (*assuming all 3 populations are normal, but with possibly different variances*).

```
pairwise.t.test(wolf$Aldrin, wolf$Depth, p.adjust.method = "none", pool.sd = FALSE)
```

```
##
##     Pairwise comparisons using t tests with non-pooled SD
##
## data:  wolf$Aldrin and wolf$Depth
##
##          Bottom Middepth
## Middepth 0.1199 -
## Surface  0.0055 0.0605
##
## P value adjustment method: none
```

```
pairwise.t.test(wolf$Aldrin, wolf$Depth, p.adjust.method = "bonferroni", pool.sd = FALSE)
```

```
##
##     Pairwise comparisons using t tests with non-pooled SD
##
## data:  wolf$Aldrin and wolf$Depth
##
##          Bottom Middepth
## Middepth 0.360  -
## Surface  0.016  0.182
##
## P value adjustment method: bonferroni
```

Simultaneous confidence intervals

- To obtain a set of 3 simultaneous Bonferroni-style 95% confidence intervals, we compute 3 individual $(1-(0.05/3))\times 100 = 98.\dot{3}\%$ intervals

Middepth vs Surface

```
t.test(wolf$Aldrin[wolf$Depth=="Middepth"],
       wolf$Aldrin[wolf$Depth=="Surface"],
       conf.level = 1-(0.05/3))$conf.int
```

```
## [1] -0.2721228  1.9321228
## attr(,"conf.level")
## [1] 0.9833333
```

Middepth vs Bottom

```
t.test(wolf$Aldrin[wolf$Depth=="Middepth"],
       wolf$Aldrin[wolf$Depth=="Bottom"],
       conf.level = 1-(0.05/3))$conf.int
```

```
## [1] -2.6317072  0.6277072
```

Surface vs Bottom

```
t.test(wolf$Aldrin[wolf$Depth=="Surface"],
       wolf$Aldrin[wolf$Depth=="Bottom"],
       conf.level = 1-(0.05/3))$conf.int
```

```
## [1] -3.3395315 -0.3244685
## attr(,"conf.level")
## [1] 0.9833333
```

- Of course, this does not include 0 because the *adjusted* p-value <0.05!

Relaxing the normality assumption

A p-value for the "global" hypothesis under weaker assumptions

- Under the formal ANOVA assumptions:
  - each population is normal
  - variances are the same

  the null hypothesis reduces to

All observations come from the same normal distribution

A weaker set of assumptions *at least under the null hypothesis* is that

All observations come from the same distribution

The powerful tool of *conditioning*

- A common tool in testing is to *condition on an "ancillary" statistic*:
  - "ancillary statistic" just means a statistic that does not tell us anything useful.
- A familiar example is the **sign test**:
  - we usually *condition* on the number N of non-zeroes (i.e. we *ignore the ties*)
- Then, p-values are in fact *conditional* probabilities, e.g. for a one-sided sign test based on the number S of positive signs, the p-value is

$$P(S \geq s \mid N = n) = P(B(n, 0.5) \geq s)$$

Conditioning on the combined sample

- If we combine all the groups into one combined sample (i.e. throw away the labels) then the remaining "data" tells us **nothing** about differences between groups i.e. what we are interested in.
- In this sense, the *combined sample* is an "ancillary statistic".
- Once we condition on the *combined sample*, the only remaining "randomness" is the *allocation of observations to groups*.
- Under the null hypothesis of "no differences between groups" **all possible allocations are equally likely**.

Enumerating all possible allocations: exact p-values

- We can (in principle) compute an *exact conditional* p-value for **any "sensible" statistic** under this particular null hypothesis.
- There are actually

$$\frac{N!}{n_1!n_2!\ldots n_g!}$$

  different possible allocations of the N total observations into groups of size $n_1, n_2, \ldots, n_g$.
- We can (in principle) compute the value of the statistic under *each possible allocation*.
- Since each such value is *equally likely under the null hypothesis*, we can use this "sampling distribution" to compute a p-value.

- Suppose the statistic is T, the observed value is t_0 and larger values indicate more evidence against the null hypothesis.
- The *exact* conditional p-value is a *simple proportion*:

$$P(T \geq t_0 \mid \text{combined sample}) = \frac{\text{no. allocations with } T \geq t_0}{\text{total no. allocations}}\,.$$

- Unfortunately, unless the sample sizes are *very small*,
  - the total number of allocations is **MASSIVE**;
  - computing the value of the statistic over all possible allocations is not feasible.
- Fortunately, we can *estimate* this proportion by taking a sufficiently large random sample from the "population of all possible allocations":
  - this is a binomial/hypergeometric (depending on whether we sample with or without replacement) proportion estimation problem!

Permutation tests

- In R, if the data is represented as a data frame with
  - observations in one column and
  - groups indicated by a factor in another column

  then is it **easy** to obtain a "random" allocation:
  - simply randomly permute the observation vector, keeping the factor vector fixed.
- Do this a large number of times.
- The "observed proportion" of the times the statistic exceeds t_0 becomes an *estimate* of the "exact" p-value.
- This general procedure is known as a *permutation test*.

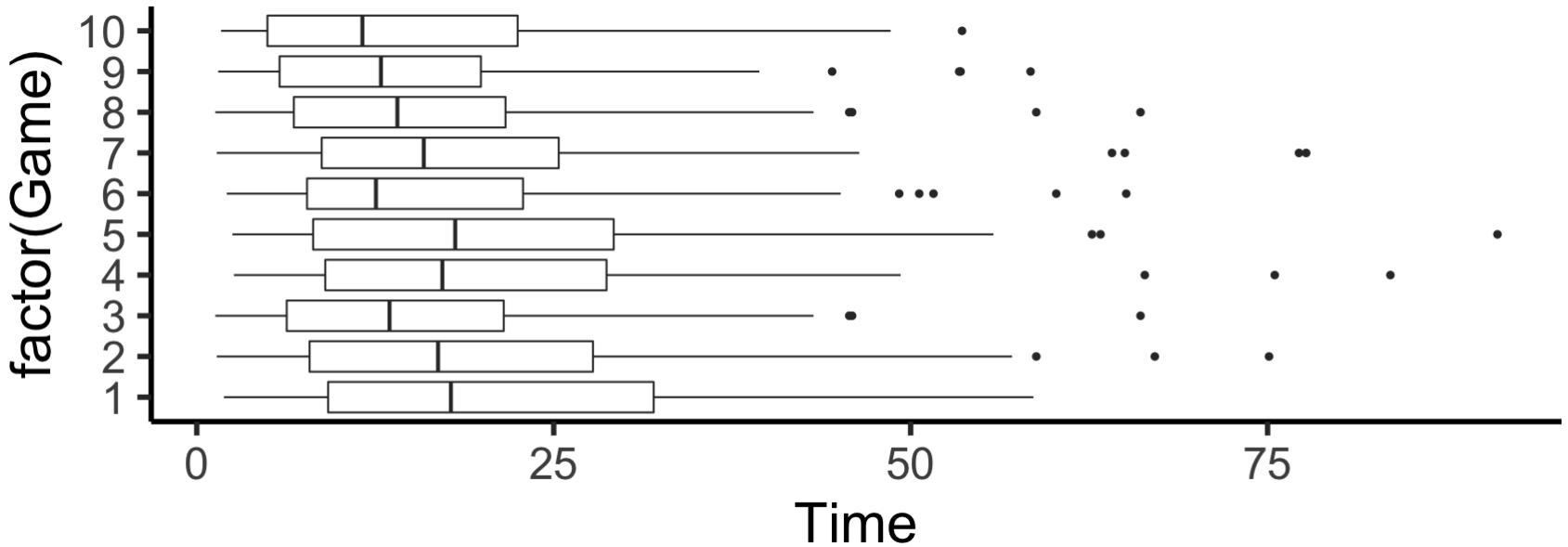
Rugby analysis

- In the early 1990's the rules for International Rugby were changed, apparently to make the game "more continuous", that is, for passages of "play" to be longer between stoppages.
  - The data is in the file `rugby.txt`.
- The lengths of time of passages of play were recorded in 10 games featuring the New Zealand national team (the "All Blacks"):
  - the first 5 games under the old rules
  - the last 5 games under the new rules
- Boxplots appear on the next slide.

Data source: http://www.statsci.org/data/oz/rugby.html

Rugby data

```
rugby = read_tsv("http://www.statsci.org/data/oz/rugby.txt")
ggplot(rugby) + aes(x = factor(Game), y = Time) +
  geom_boxplot() + coord_flip() +
  theme_classic(base_size = 30)
```



Looks skewed, not normal...

F-test?

- Let's try doing a permutation test using the F-statistic.
- It is convenient to use the R function `anova(aov(...))` or `broom::tidy(aov(...))` instead of `summary(aov(...))` since the ANOVA table is returned as numbers in a matrix:

```
rugby_anova = aov(Time ~ factor(Game), data = rugby)
anova(rugby_anova)
```

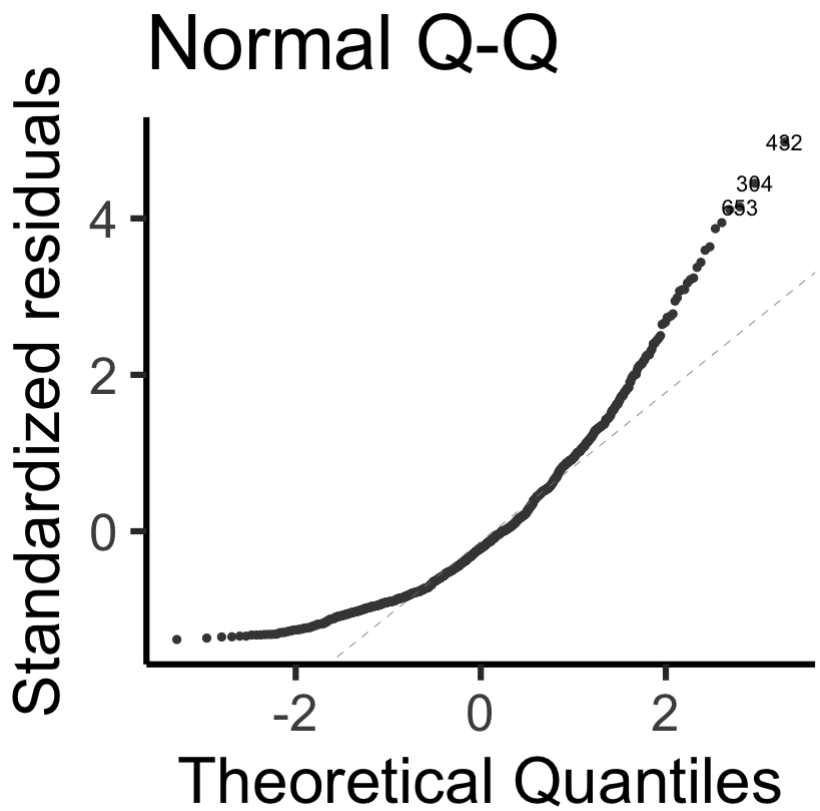
```
## Analysis of Variance Table
##
## Response: Time
##                Df Sum Sq Mean Sq F value    Pr(>F)
## factor(Game)    9   6904  767.08  3.8867 7.335e-05 ***
## Residuals     969 191241  197.36
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(rugby_anova)[1,4]
```

```
## [1] 3.886706
```

Check for normality

```
library(ggfortify)
autoplot(rugby_anova, which = 2)
```



The **broom** package

```
rugby_anova = aov(Time~factor(Game), data = rugby)
mod_sum = broom::tidy(rugby_anova)
mod_sum
```

```
## # A tibble: 2 × 6
##   term            df   sumsq meansq statistic    p.value
##   <chr>        <dbl>   <dbl>  <dbl>     <dbl>      <dbl>
## 1 factor(Game)     9   6904.   767.      3.89  0.0000733
## 2 Residuals      969 191241.   197.     NA    NA
```

```
mod_sum$statistic[1]
```

```
## [1] 3.886706
```

```
mod_sum %>% kable(format = "markdown", digits = c(0,0,0,1,3,4))
```

term	df	sumsq	meansq	statistic	p.value
factor(Game)	9	6904	767.1	3.887	1e-04
Residuals	969	191241	197.4	NA	NA

- The R function `sample()`, with only one vector argument, returns a *permutation* of that vector:

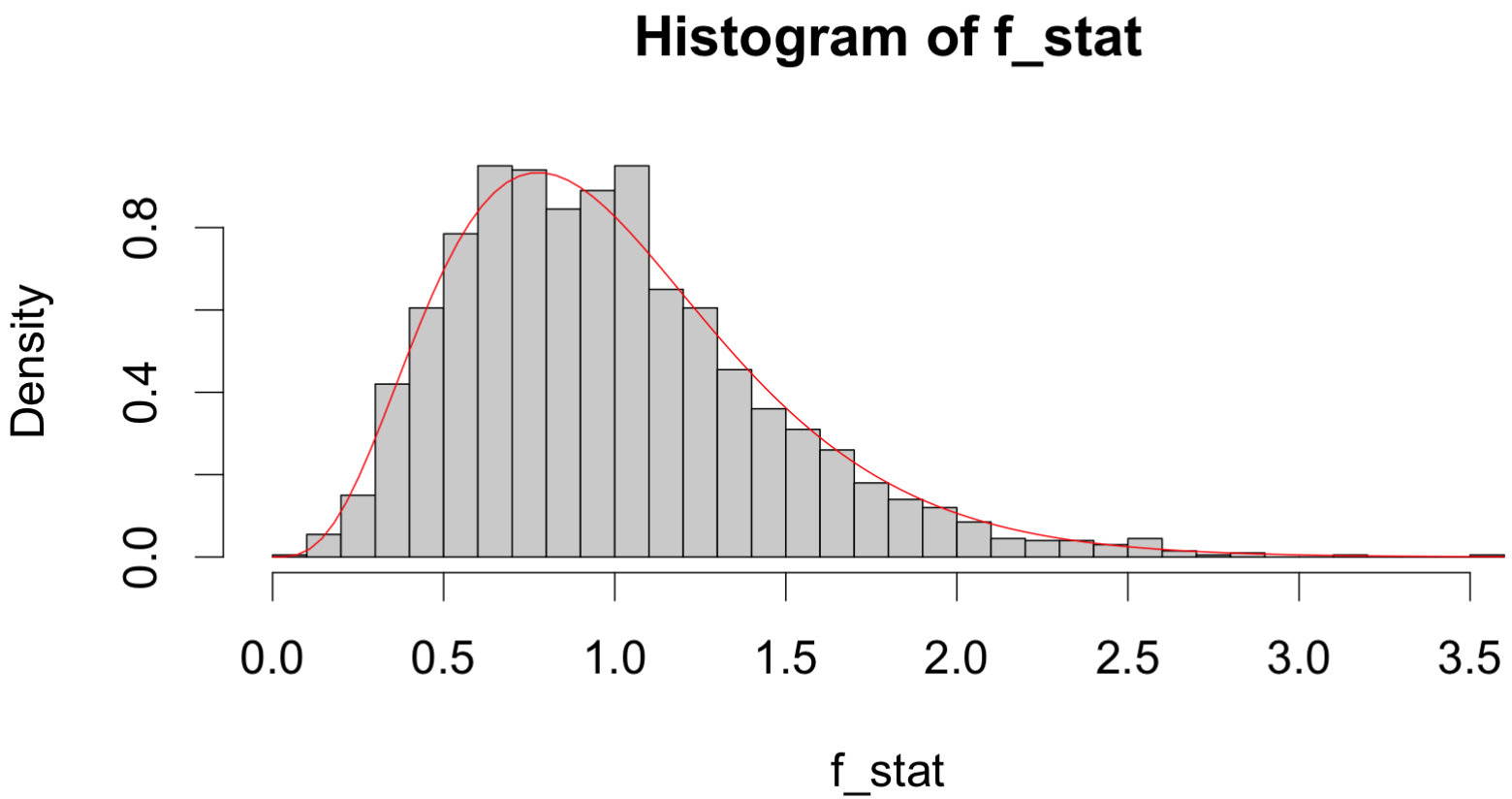
```
x = 1:5
sample(x)
```

```
## [1] 3 4 5 1 2
```

- The following loop takes a sample of size `B` from all possible permutations and computes the value of the F-statistic:

```
B = 2000
f_stat = vector(mode = "numeric", length = B)
for (i in 1:B){
  permuted_anova = aov(sample(rugby$Time)  ~ factor(rugby$Game))
  f_stat[i] = broom::tidy(permuted_anova)$statistic[1]
}
```

```
hist(f_stat, probability = TRUE, breaks = 40)
curve(df(x, 9, 969), add = TRUE, col = "red")
```



- The F-distribution density is drawn over the histogram and the fit looks pretty good.
- Our *estimate* of the *exact conditional p-value* is obtained as follows:

```
t_0 = broom::tidy(rugby_anova)$statistic[1]
t_0
```

```
## [1] 3.886706
```

```
mean(f_stat >= t_0)
```

```
## [1] 0
```

- So of the 2000 random permutations 0 gave an F-ratio bigger than (or equal to) 3.8867058.
- **We have avoided making any normality assumption here!**
- This says a lot about the *robustness* of the F-test...
    - the *conditional* distribution is very close to the corresponding F-distribution.

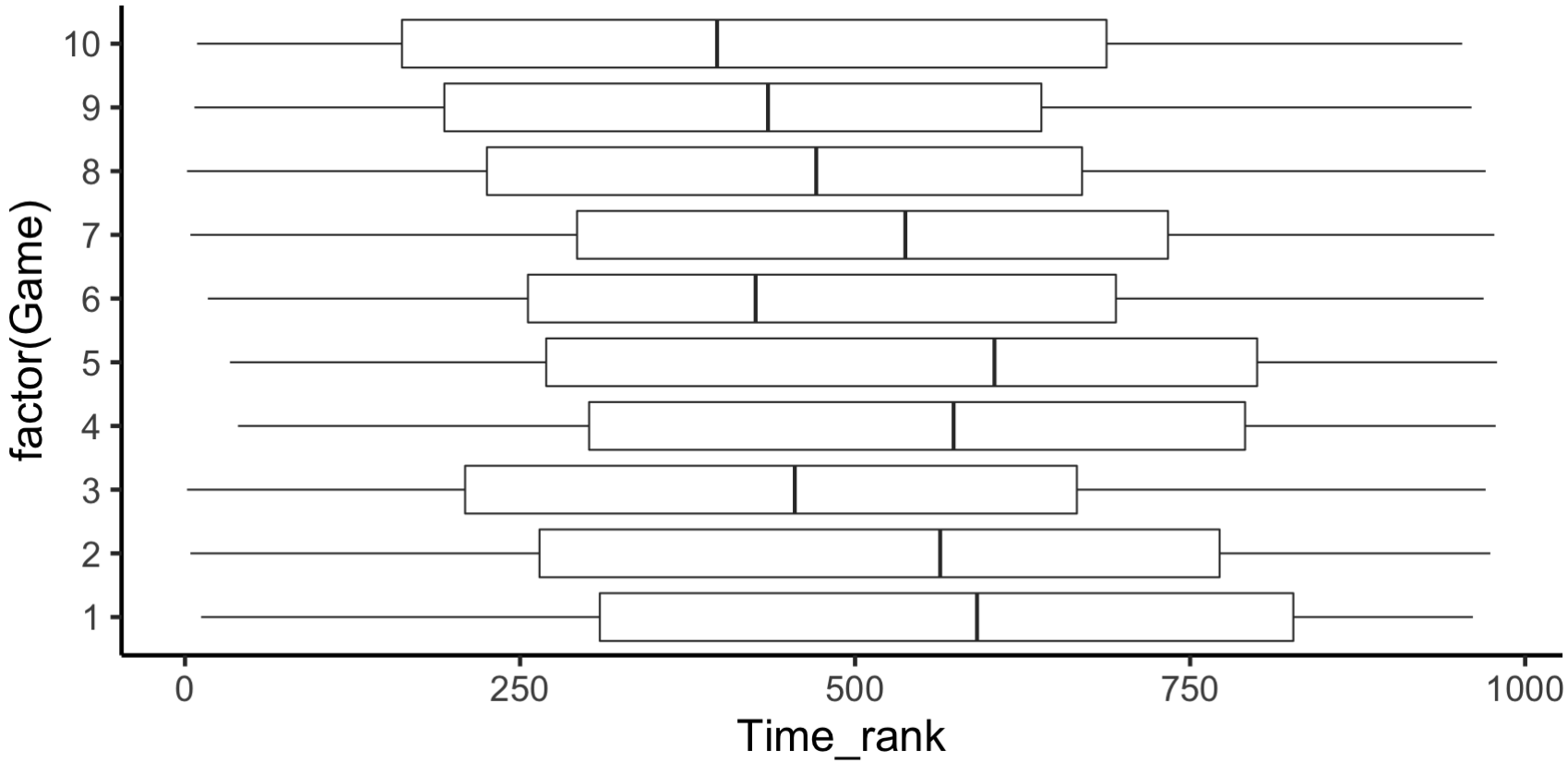
Using ranks

Kruskal-Wallis test

- This is performed by
  - replacing each observation by its "global" rank;
  - then computing the F-ratio as usual *on the ranks*.
- A p-value can be obtained
  - using a permutation test approach **or**
  - a "large-sample" χ^2 approximation can also be used

Rugby data (ranks)

```
rugby = rugby %>% ungroup() %>% mutate(Time_rank = rank(Time))
ggplot(rugby, aes(x = factor(Game), y = Time_rank)) +
  geom_boxplot() + coord_flip() + theme_classic(base_size = 24)
```



Rugby data (ANOVA on the ranks)

Perform the usual ANOVA *on the ranks*.

```
rugby_kw = aov(Time_rank ~ factor(Game), data = rugby)
broom::tidy(rugby_kw)
```

```
## # A tibble: 2 × 6
##   term            df     sumsq  meansq statistic   p.value
##   <chr>        <dbl>     <dbl>   <dbl>     <dbl>     <dbl>
## 1 factor(Game)     9  2452917. 272546.      3.49  0.000296
## 2 Residuals      969 75737977.  78161.     NA    NA
```

Kruskal-Wallis test statistic

- The traditional approach to the Kruskal-Wallis test uses a test statistic that is computed as a ratio (like the F-test)
  - the numerator is *exactly* the Treatment Sum of Squares of the ranks
  - the denominator is the *sample variance of all the ranks!*
    - this denominator is not random (it is the same regardless of the allocation).

$$T = \frac{\text{Treatment SS of the ranks}}{\text{Varance of all the ranks}}$$

- When H_0 is true, it has an *approximate* χ^2_{g-1} distribution.

```
rugby_rank_anova = aov(Time_rank ~ factor(Game), data = rugby)
rank_Treat_SS = broom::tidy(rugby_rank_anova)$sumsq[1]
rank_Treat_SS
```

```
## [1] 2452917
```

```
rank_Treat_SS/var(rugby$Time_rank)
```

```
## [1] 30.68072
```

The R function `kruskal.test()`

- This statistic is computed (and a resultant "approximate" p-value is obtained) via the R function `kruskal.test()`:

```
kruskal.test(Time ~ factor(Game), data = rugby)
```

```
##
##     Kruskal-Wallis rank sum test
##
## data:  Time by factor(Game)
## Kruskal-Wallis chi-squared = 30.681, df = 9, p-value
## = 0.0003358
```

Kruskal-Wallis procedure

1. **Hypotheses:** H_0: the response variable is distributed identically for all groups vs H_1: the response variable is systematically higher for at least one group
2. **Assumptions:** Observations are independent within each group and groups are independent of each other. The different groups follow the same distribution (differing only by the location parameter).
3. **Test statistic:** like ANOVA applied to the ranks (not examinable), see here for details. Under the null hypothesis the Kruskal-Wallis test statistic approximately follows a χ^2 distribution with $g-1$ degrees of freedom where g is the number of groups.
4. **p-value:** $P(T \geq t_0) = P(\chi^2_{g-1} \geq t_0)$.
5. **Decision:** If the p-value is less than α we reject the null hypothesis and conclude that the population mean of at least one group is significantly different to the others. If the p-value is larger than α we do not reject the null hypothesis and conclude that there is no significant difference between the population means.

Permuted ranks

- The permutation test approach is valid for any "sensible" statistic;
  - it only assumes the same distribution in each group under the null hypothesis.
- What of the "sensible statistic"?
  - If the data are truly normal, the F-statistic makes sense;
  - is it still "sensible" if the normality assumption is being relaxed?
  - could also do a permutation test using the Kruskal-Wallis statistic

Rugby data (KW permutation test)

```
set.seed(1)
B = 2000
Game = rugby$Game
Time = rugby$Time
kw_stat = vector("numeric", length = B)
for (i in 1:B){
  aov_rank = aov(rank(sample(Time)) ~ factor(Game))
  kw_stat[i] = broom::tidy(aov_rank)$statistic[1]
}
original_rank_mod = aov(rank(Time)~factor(Game))
t0 = broom::tidy(original_rank_mod)$statistic[1]
t0
```

```
## [1] 3.486988
```

```
mean(kw_stat >= t0)
```

```
## [1] 5e-04
```

References

Jaffe, P., F. Parker, and D. Wilson (1982). "Distribution of toxic substances in rivers". In: *Journal American Society of Civil Engineers, Environmental Engineering Division* 108, pp. 639-649.

Lenth, R. (2018). *emmeans: Estimated Marginal Means, aka Least-Squares Means*. R package version 1.2.3. URL: https://CRAN.R-project.org/package=emmeans.