IntroSlides with focus on statistics

Basic steps in workflow

1. *Define environment*
2. *Import*
3. *Transform*
4. *Explore (general/outlier/distribution) (go back to 3?)*
5. *Classify scale level / distribution (based on 3/4)*
6. *Describe*
7. *Test / Model (may include step 6)*
8. *Report*

Define environment

- *Activate packages to use: library() / pacman::p_load()*
- *ggplot theme: theme_set() / theme_update()*
- *flextable settings: set_flextable_defaults()*
- *knitr::opts_chunk$set()*

```r
if(!requireNamespace("pacman")){install.packages("pacman")}
pacman::p_load(conflicted,tidyverse,wrappedtools, readxl,car, flextable,
               ggbeeswarm, ggsignif, ggridges, patchwork, easystats)

conflicts_prefer(dplyr::filter,dplyr::select)
theme_set(theme_light(base_size = 20))
gdtools::register_gfont('Roboto')# Mono')
```

```
[1] TRUE
```

```r
set_flextable_defaults(
  theme_fun = theme_zebra, font.size = 18, font.family = 'Roboto',
  table.layout = 'autofit',
  padding.bottom = .2, padding.top = .2, padding.left = 2, padding.right = 2)

knitr::opts_chunk$set(message = FALSE, warning = FALSE, comment = NA, echo = TRUE)
```

Import

- *read_xlsx() / read_csv() / read_csv2()*
- options related to separators, number formats, ranges etc.
- *rename() / rename_with()*

```r
rawdata <- read_excel('Data/DOC-20230130-WA0000_.xlsx',
                      sheet = 1,col_names = TRUE)
```

Glimpse at data: Find the problems?

```r
head(rawdata,n = 15)  |> flextable() |>
  theme_zebra(even_body = 'aquamarine',odd_body = 'antiquewhite')
```

CODE OF CUP	CODE OF SAMPLE	WEIGHT OF EMPTY ALUMINUM(wt)	WEIGTH OF ALMINIUM CUP + SAMPLE (Wt + s)	WEIGTH OF ALUMINIUM CAP + SAMPLE AFTER DRYING (Wt-AL +s+d)	weight of sample before drying (Wts)	weight of sample after drying (Wts+d)	MOISTURE CONTENT (%)
69	D	4.1974	9.3865	4.7000	5.1891	4.6865	90.31431
	D	4.1964	9.2734	4.4670	5.0770	4.8064	94.67008
A	D	4.2108	9.2653	4.6670	5.0545	4.5983	90.97438
114	D	4.2134	9.3146	4.6345	5.1012	4.6801	91.74508
M1	D	4.1856	9.3147	4.6171	5.1291	4.6976	91.58722
a/17	D	4.2090	9.3204	4.5661	5.1114	4.7543	93.01366
8	D	4.1894	9.2661	4.5778	5.0767	4.6883	92.34936
33	D	4.1968	9.2880	4.6057	5.0912	4.6823	91.96849
M	D	4.1535	9.2872	4.6350	5.1337	4.6522	90.62080
E/18/1	D	4.2534	9.2476	4.7403	4.9942	4.5073	90.25069
24/A2	D	4.2066	8.3463	4.5849	4.1397	3.7614	90.86166
13	A	4.1554	9.2384	4.7402	5.0830	4.4982	88.49498
Xp	A	4.1893	9.2495	4.7381	5.0602	4.5114	89.15458
2p/029	A	4.0654	9.2173	4.6940	5.1519	4.5233	87.79868
15	A	4.0641	9.2032	4.8124	5.1391	4.3908	85.43908

Rename

```r
colnames(rawdata)
```

```
[1] "CODE OF CUP"
[2] "CODE OF SAMPLE"
[3] "WEIGHT OF EMPTY ALUMINUM(wt)"
[4] "WEIGTH OF ALMINIUM CUP + SAMPLE (Wt + s)"
[5] "WEIGTH OF ALUMINIUM CAP + SAMPLE  AFTER DRYING (Wt-AL +s+d)"
[6] "weight of sample before drying (Wts)"
[7] "weight of sample after drying (Wts+d)"
[8] "MOISTURE CONTENT (%)"
```

```r
rawdata <- rawdata |>
  rename(Region=`CODE OF SAMPLE`) |>
  rename_with(.fn = ~str_replace_all(.,
                                     c('AL.+UM'= 'Cup', 'C[UA]P' = 'Cup','\\(\\w+.*\\)
                                       'Cup Cup'='Cup','  '=' ')) |>
                str_to_title() |> str_trim())
cn()
```

```
[1] "Code Of Cup"                          "Region"
[3] "Weight Of Empty Cup"                  "Weigth Of Cup + Sample"
[5] "Weigth Of Cup + Sample After Drying" "Weight Of Sample Before Drying"
[7] "Weight Of Sample After Drying"        "Moisture Content (%)"
```

Transform

- *Change or create columns with mutate() / mutate(across())*
- *e.g. for log-transformation, creation of factors, text recoding*

```
rawdata <- rawdata |>
  mutate(
    `Weight Of Sample After Drying`=`Weigth Of Cup + Sample After Drying`-
      `Weight Of Empty Cup`,
    `Dry Content (%)`=`Weight Of Sample After Drying`*100/
      `Weight Of Sample Before Drying`,
    `Moisture Content (%)`=100-`Dry Content (%)`)
```

Code Of Cup	Region	Weight Of Empty Cup	Weigth Of Cup + Sample	Weigth Of Cup + Sample After Drying	Weight Of Sample Before Drying	Weight Of Sample After Drying	Moisture Content (%)	Dry Content (%)
69	D	4.1974	9.3865	4.7000	5.1891	0.5026	90.31431	9.685687
	D	4.1964	9.2734	4.4670	5.0770	0.2706	94.67008	5.329919
A	D	4.2108	9.2653	4.6670	5.0545	0.4562	90.97438	9.025621
114	D	4.2134	9.3146	4.6345	5.1012	0.4211	91.74508	8.254920
M1	D	4.1856	9.3147	4.6171	5.1291	0.4315	91.58722	8.412782

Explore / group variables

Explore (general/outlier/distribution)

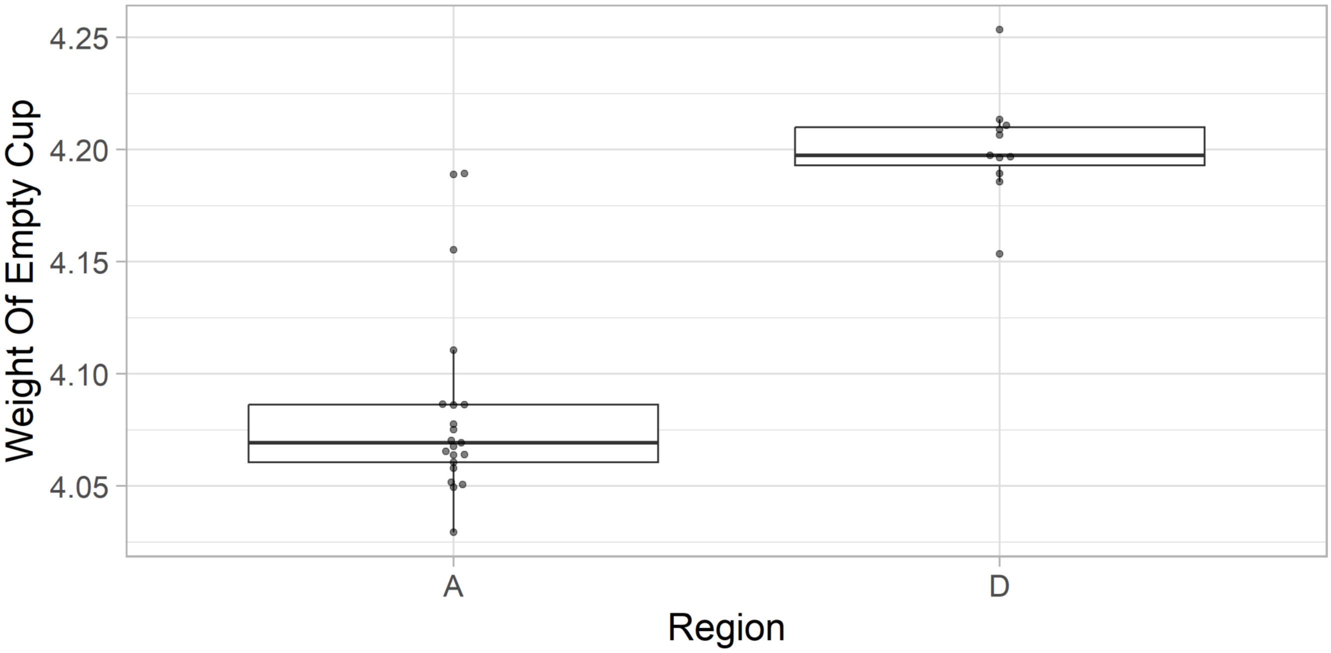
- *ggplot()+geom_boxplot() / geom_beeswarm() / geom_density()*
- *ks.test() / ksnormal() / shapiro.test()*

Classify scale level / distribution

- *gaussvars / ordvars / factvars, possibly more…*
- *Store variables accordingly, e.g. FindVars()*

Explore: Outlier

```
ggplot(data = rawdata,
       aes(x = `Region`,
           y = `Weight Of Empty Cup`))+
  geom_boxplot(outlier.alpha = 0) + #hide outliers, beeswarm will plot them
  geom_beeswarm(alpha=.5)
```

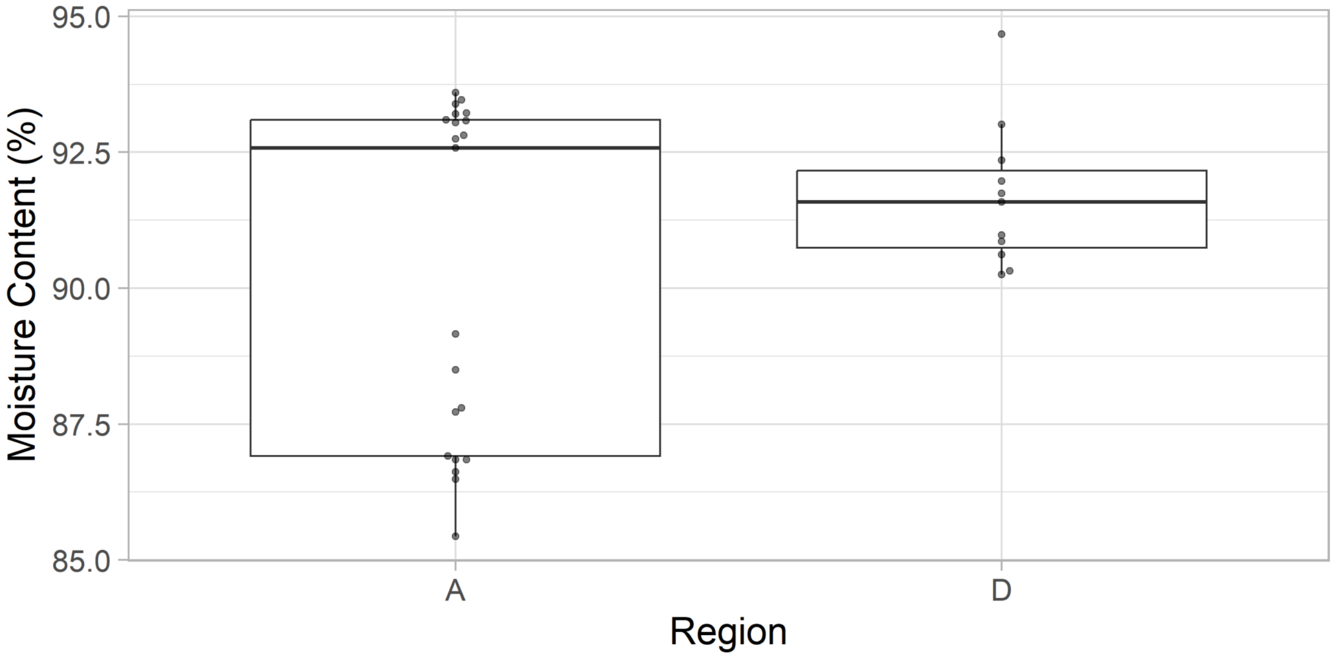


Handle outliers?

Removal is the worst strategy possible, correct errors, think about distributions, winsorize…

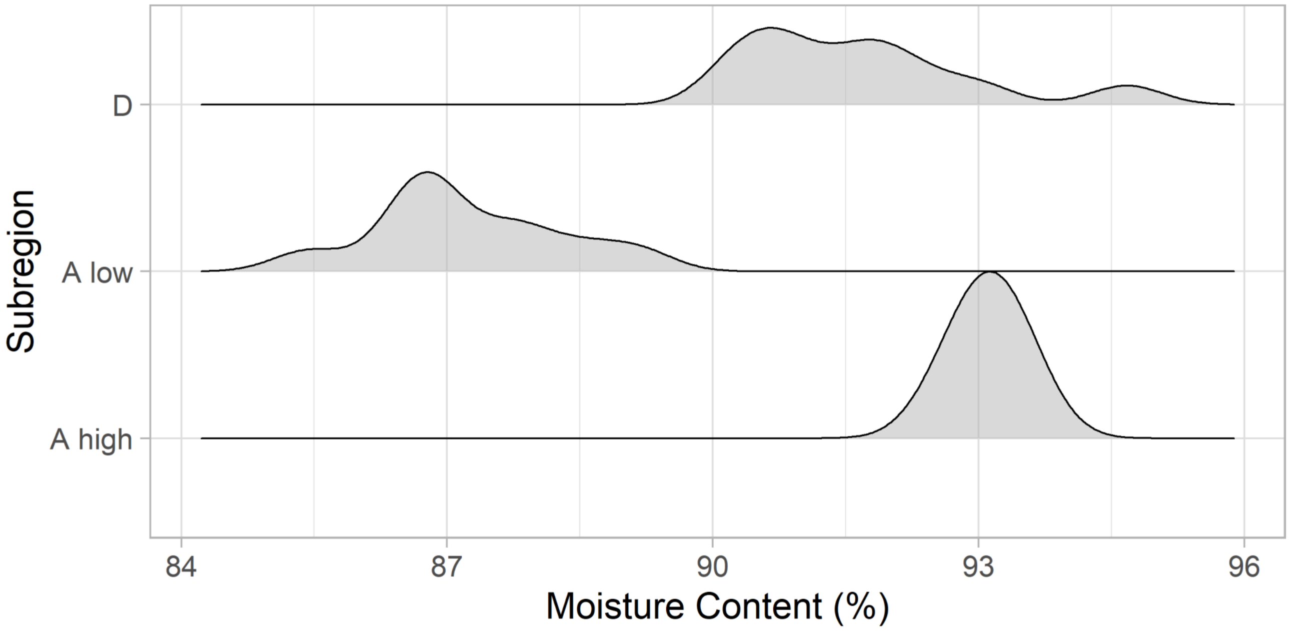
Explore: Unexpecteds

```
ggplot(data = rawdata,
       aes(x = `Region`,
           y = `Moisture Content (%)`))+
  geom_boxplot(outlier.alpha = 0) +
  geom_beeswarm(alpha=.5)
```



Transform Subregions?

```
rawdata <- mutate(rawdata, Subregion = case_when(
  `Region`=='A' & `Moisture Content (%)` > 90 ~ 'A high',
  `Region`=='A' & `Moisture Content (%)` <= 90 ~ 'A low',
  `Region`=='D' ~ 'D') |> factor())
ggplot(data = rawdata,aes(x = `Moisture Content (%)`,y=Subregion))+
  geom_density_ridges(alpha=.5, scale=1)
```



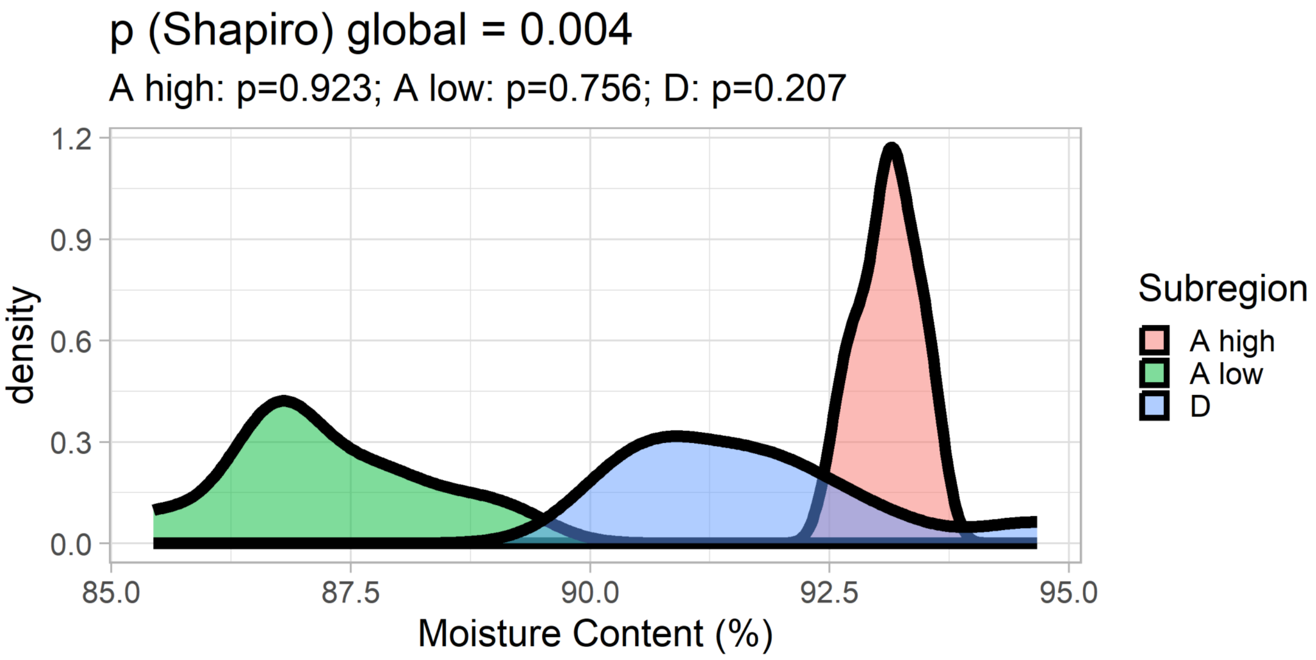
Explore: Normal distribution 1

- Gaussian Normal distribution is required for many statistical procedures
- Common tests are graphical exploration, Shapiro-Wilk-test and Kolmogorov-Smirnov-test

```r
p_normal <-
  shapiro.test(x = rawdata$`Moisture Content (%)`)
p_normal
```

```
    Shapiro-Wilk normality test

data:  rawdata$`Moisture Content (%)`
W = 0.89133, p-value = 0.003752
```

```r
ggplot(data = rawdata,aes(x = `Moisture Content (%)`,fill=`Subregion`))+
  geom_density(linewidth=3, alpha=.5)+
  labs(title = paste('p (Shapiro) global',
               formatP(pIn = p_normal$p.value, pretext = T)),
       subtitle = rawdata |> group_by(Subregion) |>
  summarize(pNormal=shapiro.test(`Moisture Content (%)`)$p.value |> formatP()) |>
  unite('p(Normal)',sep = ': p=') |> pull(1) |> paste(collapse = '; '))
```



Explore: Normal distribution 2

```
rawdata |>
  group_by(`Subregion`) |>
  summarize(across(.cols = where(is.numeric), .fns = ksnormal)) |>
  pivot_longer(cols = -1, names_to = 'Variable', values_to = 'pKS') |>
  pivot_wider(names_from = `Subregion`, values_from = pKS)
```

```
# A tibble: 7 × 4
  Variable                              `A high` `A low`      D
  <chr>                                    <dbl>   <dbl>  <dbl>
1 Weight Of Empty Cup                      0.989   0.444 0.697
2 Weigth Of Cup + Sample                   0.785   0.980 0.0187
3 Weigth Of Cup + Sample After Drying      0.900   0.710 0.969
4 Weight Of Sample Before Drying           0.196   0.999 0.0716
5 Weight Of Sample After Drying            0.976   0.555 1.00
6 Moisture Content (%)                     0.975   0.733 0.954
7 Dry Content (%)                          0.975   0.733 0.954
```

Explore: Group variables by type/distribution

Scale level determines what statistics are appropriate

Typical scale levels are

- nominal/categorical/factorial/qualitative: just different groups (species, eye color, genotype, treatment)
- ordered categories: few groups with inherent order (quality bad<medium<good, pain between 0 and 10)
- ordinal measures: many different values, natural order, no distribution assumption (satisfaction on a scale from 0 to 100)
- measures following a Normal distribution
- possibly measures from other known distributions (beta, log-normal, poisson…), often treated as ordinal

Make type decision obvious/reproducible

```r
gaussvars <- FindVars(varnames = c('Weight Of Sample','Content'),
                      allnames = cn(rawdata))
gaussvars
```

```
$index
[1] 6 7 8 9

$names
[1] "Weight Of Sample Before Drying" "Weight Of Sample After Drying"
[3] "Moisture Content (%)"           "Dry Content (%)"

$bticked
[1] "`Weight Of Sample Before Drying`" "`Weight Of Sample After Drying`"
[3] "`Moisture Content (%)`"           "`Dry Content (%)`"

$count
[1] 4
```

```r
ordvars <- FindVars(c('Cup'), exclude = 'Code')
ordvars$names
```

```
[1] "Weight Of Empty Cup"                "Weigth Of Cup + Sample"
[3] "Weigth Of Cup + Sample After Drying"
```

```r
factvars <- FindVars('egion',casesensitive = FALSE)
factvars$bticked
```

```
[1] "`Region`"    "`Subregion`"
```

Model

Describe

- *mean() / sd() / meansd()*
- *median() / quantile() / median_quart()*
- *table() / prop.table() / cat_desc_stats()*

Test

- *t.test() / lm()+[Aa]nova() / compare2numvars()*
- *wilcox.test()*
- *fisher.test() / glm(family=binomial)*

Model: Describe

Sample size n: per variable, if there are NAs	
Mean: central tendency, the expected *typical* value	$\frac{\sum x}{n}$
Variance: measure for variability/heterogeneity of data	
Standard deviation SD: the *typical* weighted deviation from the mean	

Standard error of the mean SEM: how reliable is the mean *estimate*, what would be the expected SD of means from repeated experiments?	
Median: Split between lower/upper 50% of data	
Quartiles: Split at 25%/50%/75% of data (more general: **Quantiles**, e.g.**Percentiles**), used in boxplot	various computational approaches

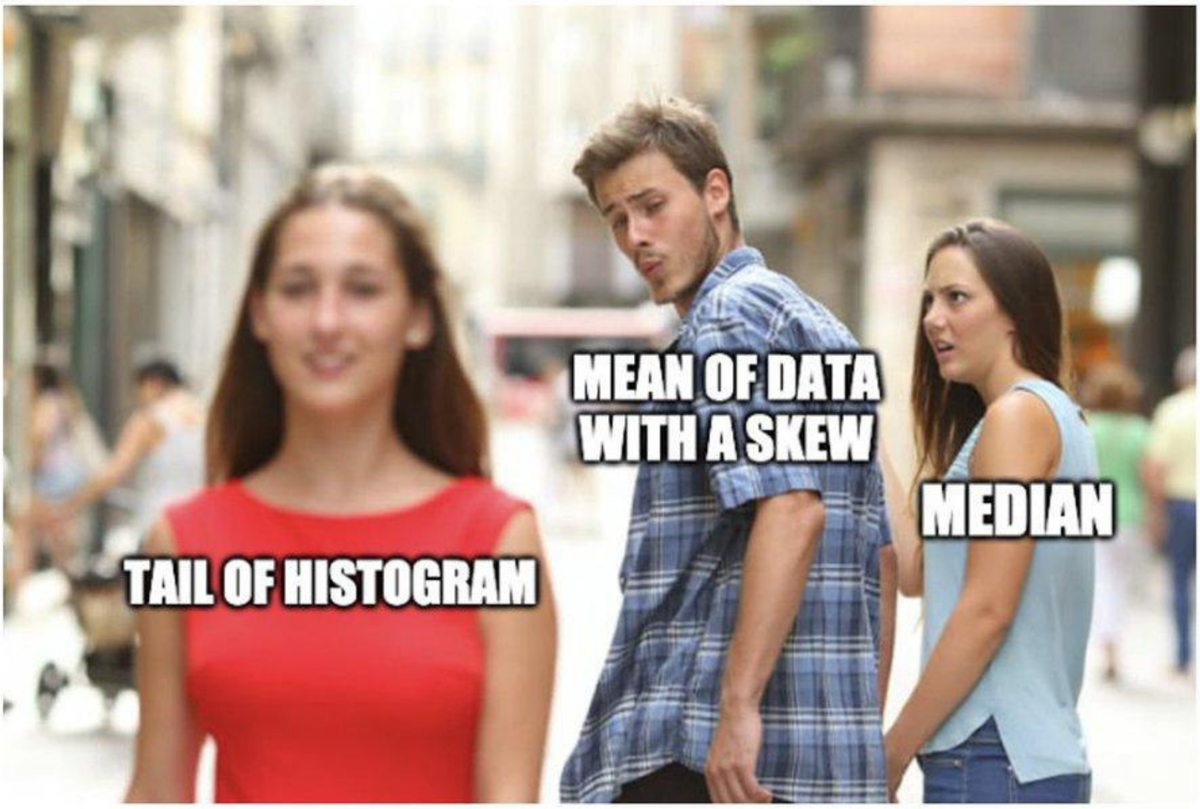
```r
desc_gauss <- rawdata |>
  summarize(across(.cols = gaussvars$names,
                   .fns = meansd))
desc_gauss
```

```
# A tibble: 1 × 4
  `Weight Of Sample Before Drying` Weight Of Sample Aft…¹ `Moisture Content (%)`
  <chr>                            <chr>                  <chr>
1 5.1 ± 0.2                        0.47 ± 0.14            91 ± 3
# ℹ abbreviated name: ¹`Weight Of Sample After Drying`
# ℹ 1 more variable: `Dry Content (%)` <chr>
```

```r
desc_ord <- rawdata |>
  summarize(across(ordvars$names,.fns=~median_quart(.x,roundDig = 3))) |>
  pivot_longer(everything(),
               names_to = 'Measure', values_to = 'Median[1Q/3Q]')
desc_ord
```

```
# A tibble: 3 × 2
  Measure                            `Median[1Q/3Q]`
  <chr>                              <chr>
1 Weight Of Empty Cup                4.09 (4.06/4.19)
2 Weigth Of Cup + Sample             9.25 (9.22/9.29)
3 Weigth Of Cup + Sample After Drying 4.61 (4.45/4.72)
```

Descriptive Stats should match distribution and data



Model: Test

Tests require hypotheses

IT SEEMS THERE IS A DIFFERENCE BETWEEN A "HYPOTHESIS" AND A "GUESSTIMATE."
DIST. BY CREATORS SPEEDBUMP.COM ©2012 COVERLY 1-17

Null hypothesis ?

- Working hypothesis: This is what you expect!
  E.g. treatment is lowering blood pressure more than placebo, transgenic animals become obese, bio reactor A is more efficient than B, concentration of substance is correlated with speed of reaction …
- Null hypothesis: This is what you test!
  No difference / relation, BP under therapy = BP under placebo

4 possibilities:

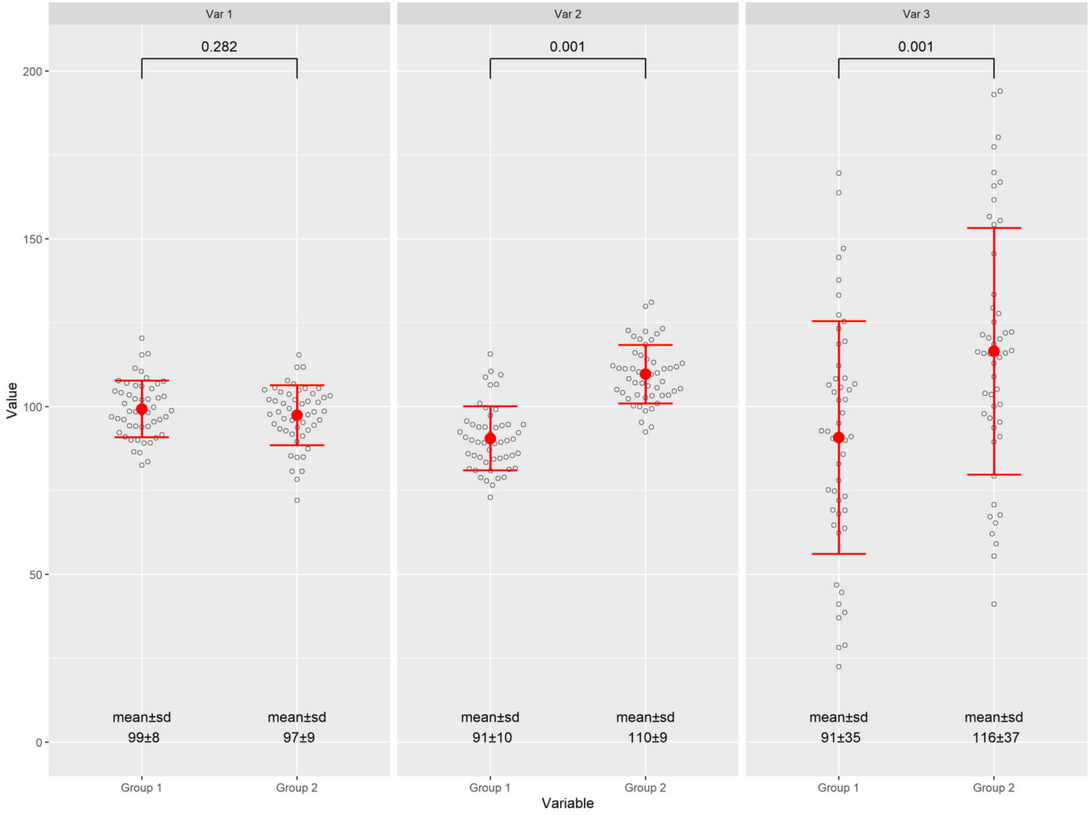
- Null hypothesis correct, test false positive (case A): alpha-error
- Null hypothesis correct, test correct negative (case B)
- Null hypothesis false, test false negative (case C): beta-error
- Null hypothesis false, test correct positive (case D)

Significance: NOT probability of case A, but probability of your data given the NULL hypothesis, calculated from your data, conventionally <0.05

Power: Probability of case D, *estimated* based on assumptions about effects and sample size, *calculation* would require knowledge of true difference, conventionally set at 0.80

Test functions

t-test / Wilcoxon-test (aka Mann-Whitney U-test)



t-test

- Assumptions: Continuous data with Normal distribution
- 1 or 2 (independent or dependent) samples with/without equal variances
- how big is the mean difference relative to uncertainty?
  t = (mean$_1$ - mean$_2$)/SEM
- t follows a t-distribution, allows estimation of probability of t under the NULL hypothesis

Wilcoxon-test

- nonparametric, no distribution is assumed
- based on rank-transformed data
- insensitive to extreme values

Test examples: *single variables*

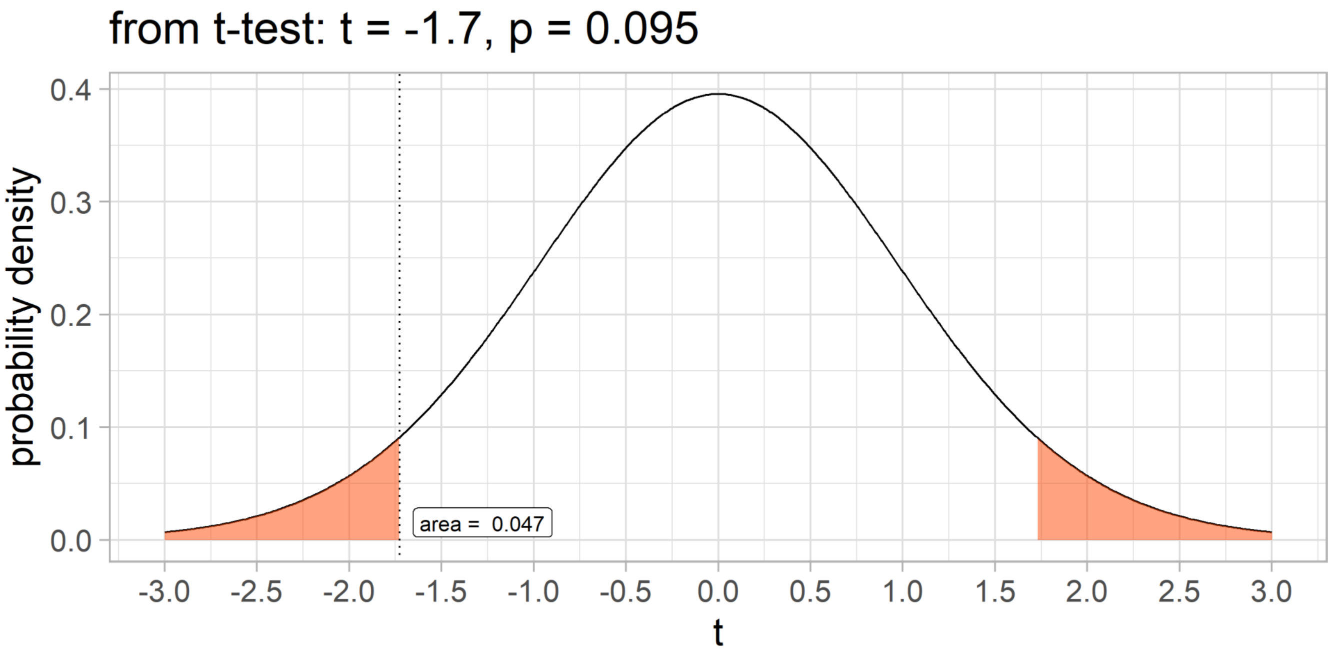
```r
#t-Test with test for equal variances
t_out <- t.test(formula=`Moisture Content (%)`~`Region`, data=rawdata,
                var.equal=var.test(
                  formula=`Moisture Content (%)`~`Region`,
                  data=rawdata)$p.value>.05)
t_out
```

```
    Welch Two Sample t-test

data:  Moisture Content (%) by Region
t = -1.7274, df = 29.239, p-value = 0.09465
alternative hypothesis: true difference in means between group A and group D is not equal
to 0
95 percent confidence interval:
 -2.9624835  0.2490486
sample estimates:
mean in group A mean in group D
       90.31199        91.66870
```

```r
#Wilcoxon-Test
wilcox.test(`Moisture Content (%)`~`Region`,
            data = rawdata)
```

```
    Wilcoxon rank sum exact test

data:  Moisture Content (%) by Region
W = 107, p-value = 0.7547
alternative hypothesis: true location shift is not equal to 0
```

From t to p



Model: Test 2 / multiple variables

```r
test_gauss <- compare2numvars(data = rawdata,
                              dep_vars = gaussvars$names,
                              indep_var = 'Region',
                              gaussian = TRUE,
                              round_p = 5)
test_gauss |> flextable() |>
  theme_zebra(even_body = 'aquamarine',odd_body = 'antiquewhite')
```

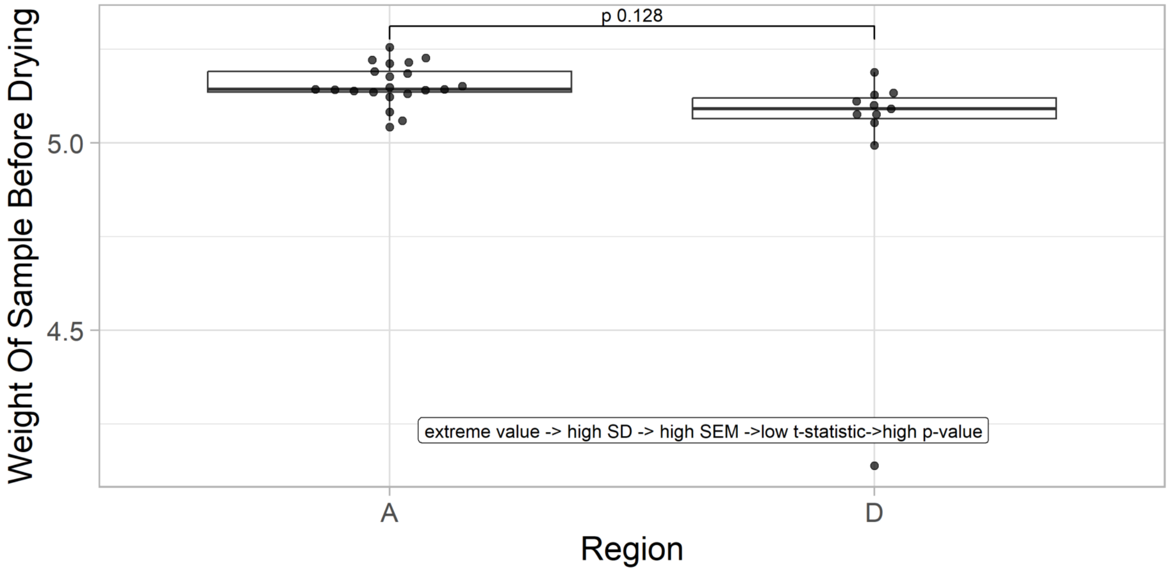
Variable	desc_all	Region A	Region D	p
Weight Of Sample Before Drying	5.1 ± 0.2	5.2 ± 0.1	5.0 ± 0.3	0.12830
Weight Of Sample After Drying	0.47 ± 0.14	0.50 ± 0.16	0.42 ± 0.07	0.04937
Moisture Content (%)	91 ± 3	90 ± 3	92 ± 1	0.09465
Dry Content (%)	9.2 ± 2.7	9.7 ± 3.1	8.3 ± 1.3	0.09465

```
test_ord <- compare2numvars(data = rawdata,
                            dep_vars = ordvars$names,
                            indep_var = 'Region',
                            gaussian = FALSE,round_desc = 3)
test_ord |> flextable() |>
  theme_zebra(even_body = 'aquamarine',odd_body = 'antiquewhite')
```

Variable	desc_all	Region A	Region D	p
Weight Of Empty Cup	4.09 (4.06/4.19)	4.07 (4.06/4.09)	4.20 (4.19/4.21)	0.001
Weigth Of Cup + Sample	9.25 (9.22/9.29)	9.24 (9.22/9.25)	9.29 (9.27/9.31)	0.003
Weigth Of Cup + Sample After Drying	4.61 (4.45/4.72)	4.46 (4.42/4.74)	4.62 (4.58/4.66)	0.611

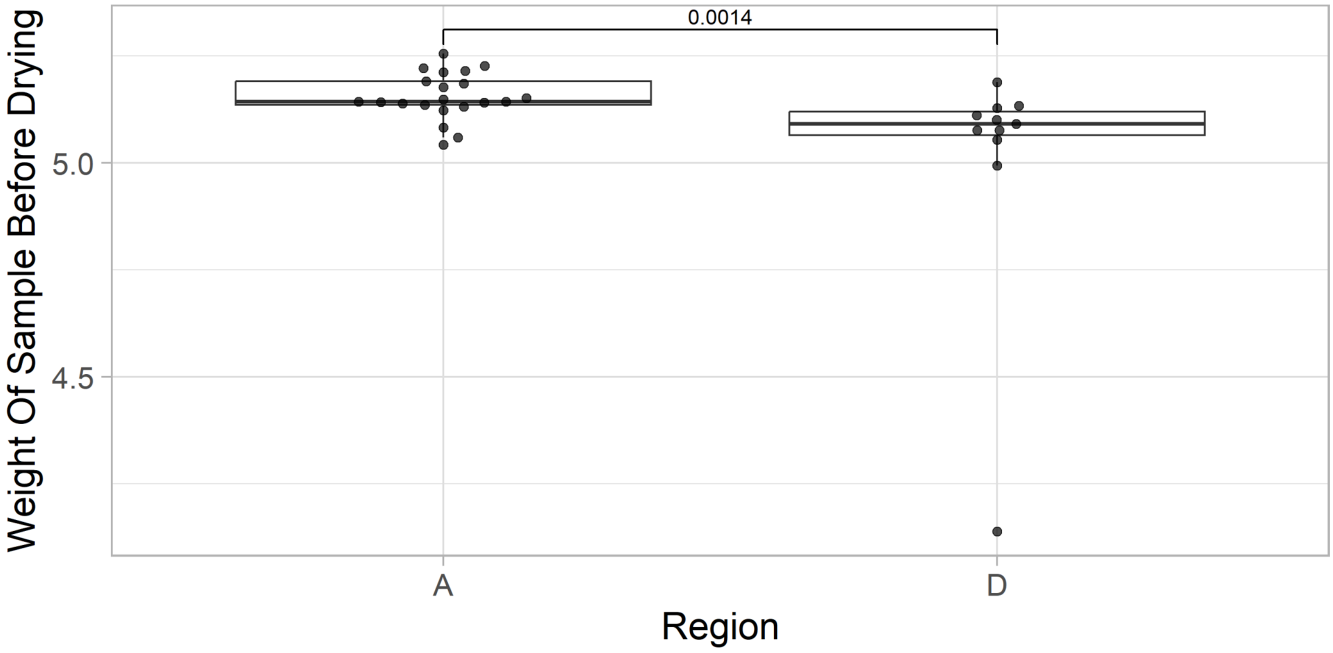
Show results

```
ggplot(rawdata, aes(x = `Region`,y = `Weight Of Sample Before Drying`))+
  geom_boxplot(outlier.alpha = 0)+
  geom_beeswarm(alpha=.7, size=2,cex = 2)+
  annotate(geom = 'label',x=2,y=4.2,
           label='extreme value -> high SD -> high SEM ->low t-statistic->high p-value
           hjust=0.8,vjust=0)+
  geom_signif(comparisons = list(c(1,2)),
              annotations = paste('p',formatP(t_out$p.value)))
```



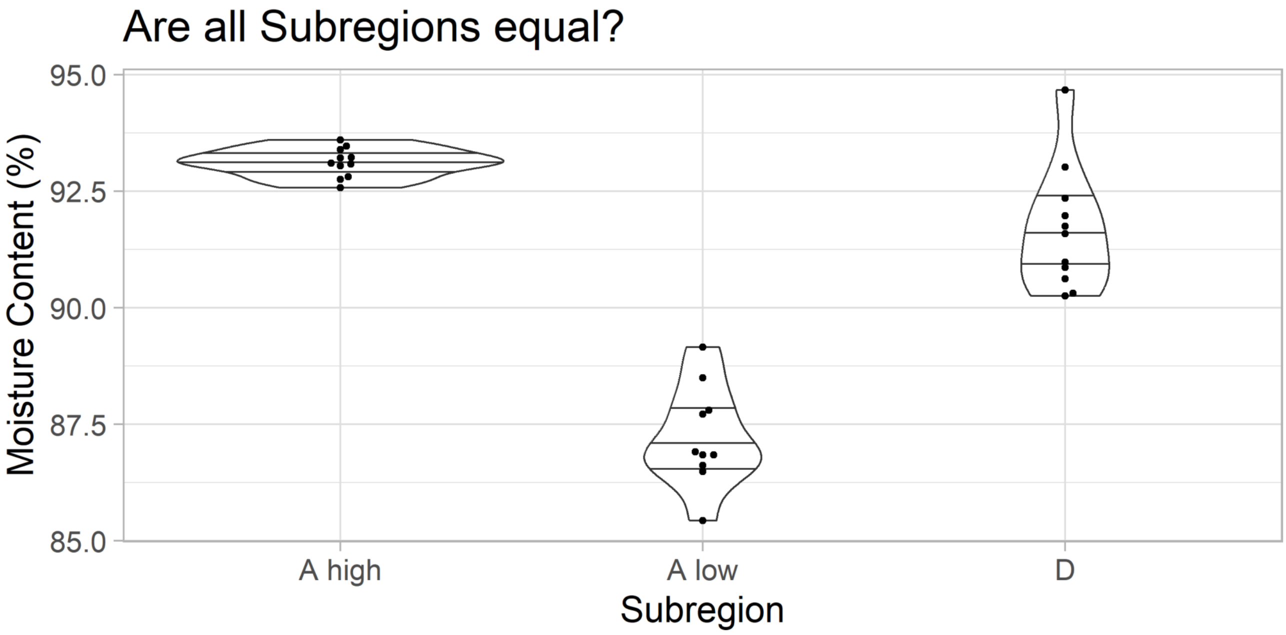
Re-thinking test decision?

```r
ggplot(rawdata, aes(x = `Region`,y = `Weight Of Sample Before Drying`))+
  geom_boxplot(outlier.alpha = 0)+
  geom_beeswarm(alpha=.7, size=2,cex = 2)+
  geom_signif(comparisons = list(c(1,2)),test = wilcox.test)
```



Model: linear models 1 / univariable

```r
plottmp <- ggplot(rawdata,aes(Subregion,`Moisture Content (%)`))+
  geom_violin(draw_quantiles = c(.25,.5,.75))+
  geom_beeswarm()+

  ggtitle('Are all Subregions equal?')
print(plottmp)
```



ANOVA: build model

```
rawdata |> group_by(Subregion)  |>
  summarize(MeanMoisture=mean(`Moisture Content (%)`) |> roundR(4)) |>
  pivot_wider(names_from = Subregion,values_from = MeanMoisture) |>
  rename_with(~paste('Mean moisture %\n',.x)) |> flextable()|>
  theme_zebra(even_body = 'aquamarine',odd_body = 'antiquewhite')
```

Mean moisture % A high	Mean moisture % A low	Mean moisture % D
93.11	87.23	91.67

```
lm1<- lm(`Moisture Content (%)`~Subregion, data=rawdata)
lm1
```

```
Call:
lm(formula = `Moisture Content (%)` ~ Subregion, data = rawdata)

Coefficients:
   (Intercept)  SubregionA low      SubregionD  
        93.112          -5.879          -1.443  
```

ANOVA: get p-values

```
anova(lm1) |> broom::tidy() |> flextable()|>
  theme_zebra(even_body = 'aquamarine',odd_body = 'antiquewhite')
```

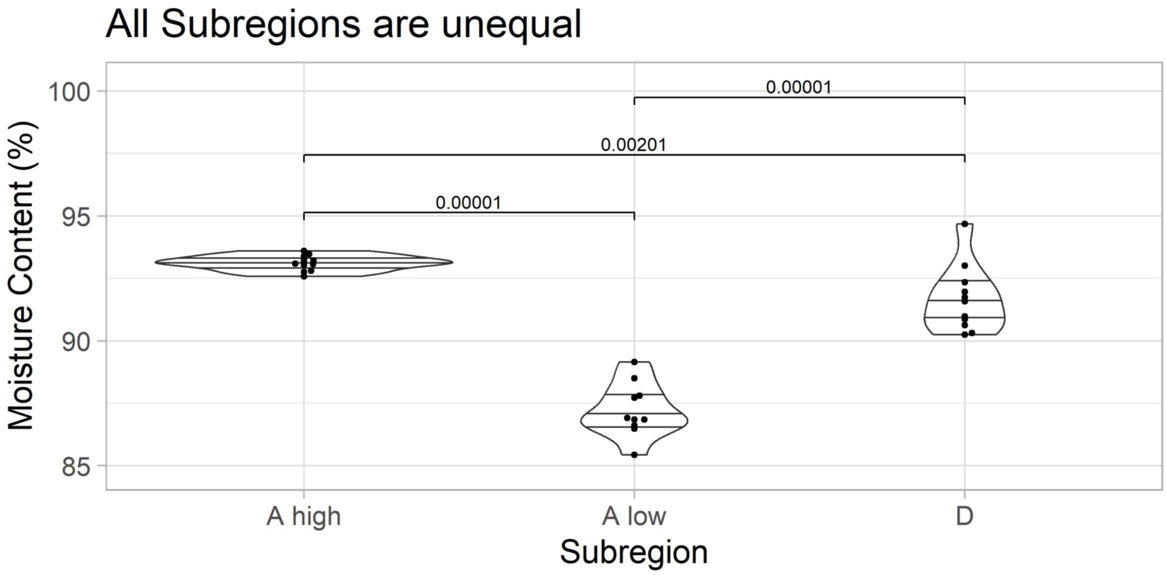
term	df	sumsq	meansq	statistic	p.value
Subregion	2	194.34239	97.1711969	97.76477	0.000000000001292127
Residuals	29	28.82393	0.9939285		

```
#post-hoc
(posthoc_out <- pairwise.t.test(x = rawdata$`Moisture Content (%)`,
                                g = rawdata$Subregion,
                                p.adjust.method = 'fdr')$p.value |>
    formatP(ndigits = 5))
```

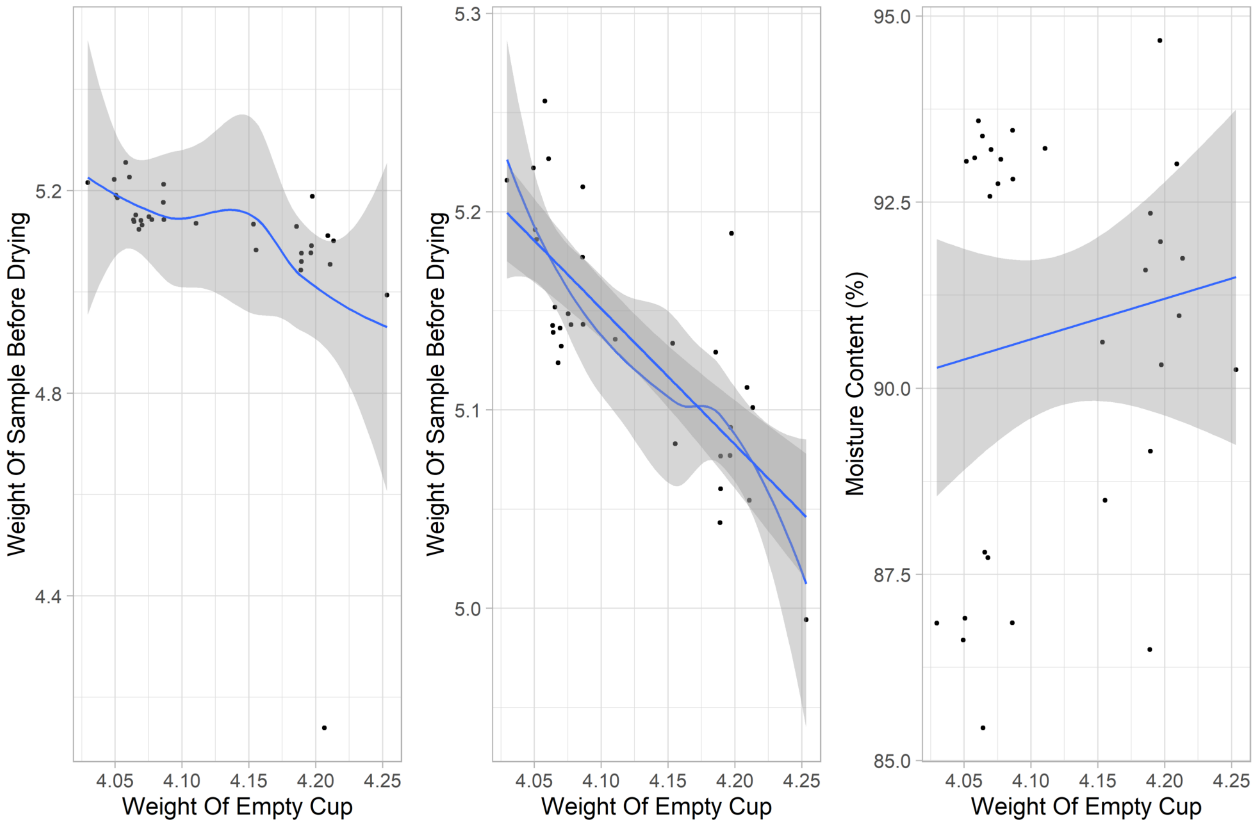
```
      A high    A low    
A low "0.00001" "    NA" 
D     "0.00201" "0.00001"
```

Visualize ANOVA

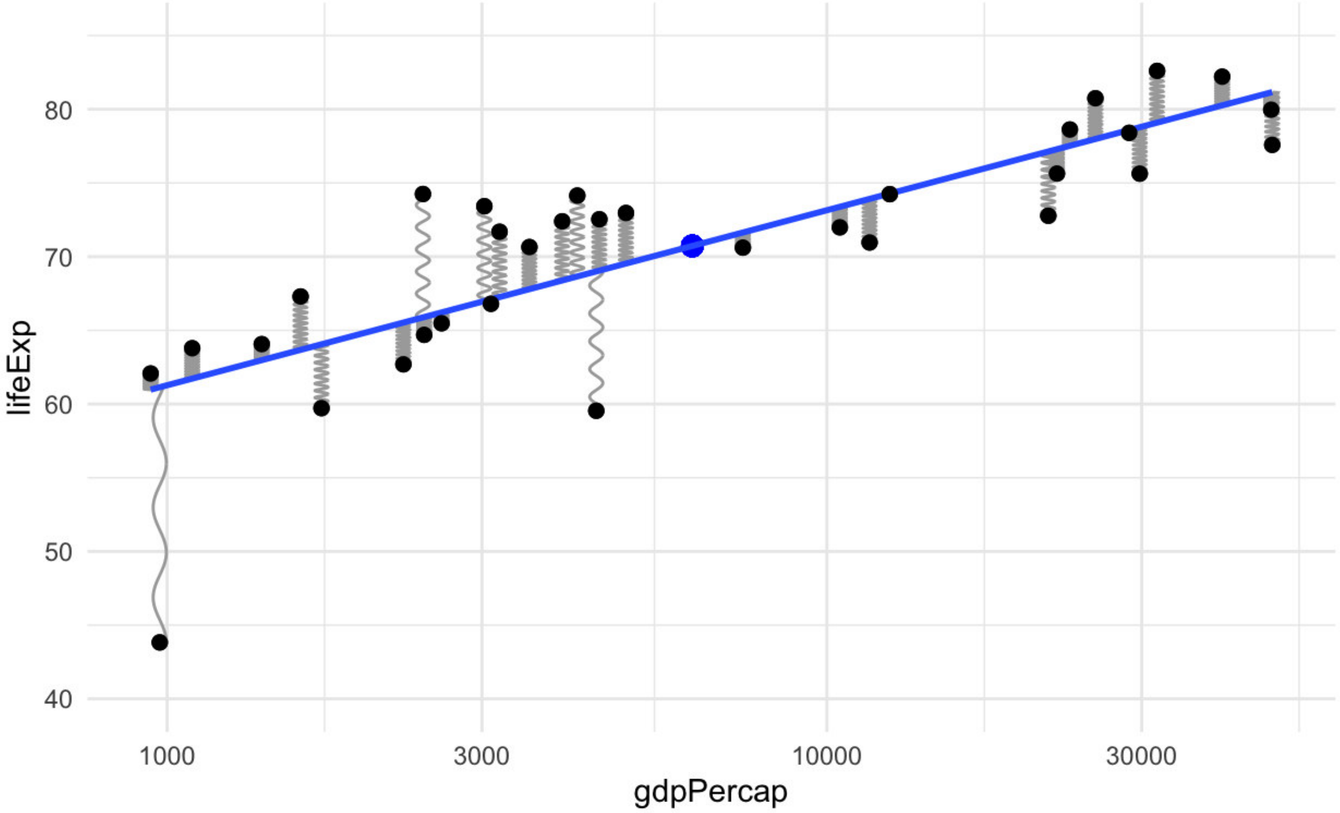
```
ggplot(rawdata,aes(Subregion,`Moisture Content (%)`))+
  geom_violin(draw_quantiles = c(.25,.5,.75))+
  geom_beeswarm()+
  geom_signif(comparisons = list(c(1,2),c(1,3),c(2,3)),
              annotations = c(posthoc_out[,1], posthoc_out[2,2]),
              step_increase = .25)+
  scale_y_continuous(expand = expansion(mult = .1))+
  ggtitle('All Subregions are unequal')
```



Regression: Scatterplot



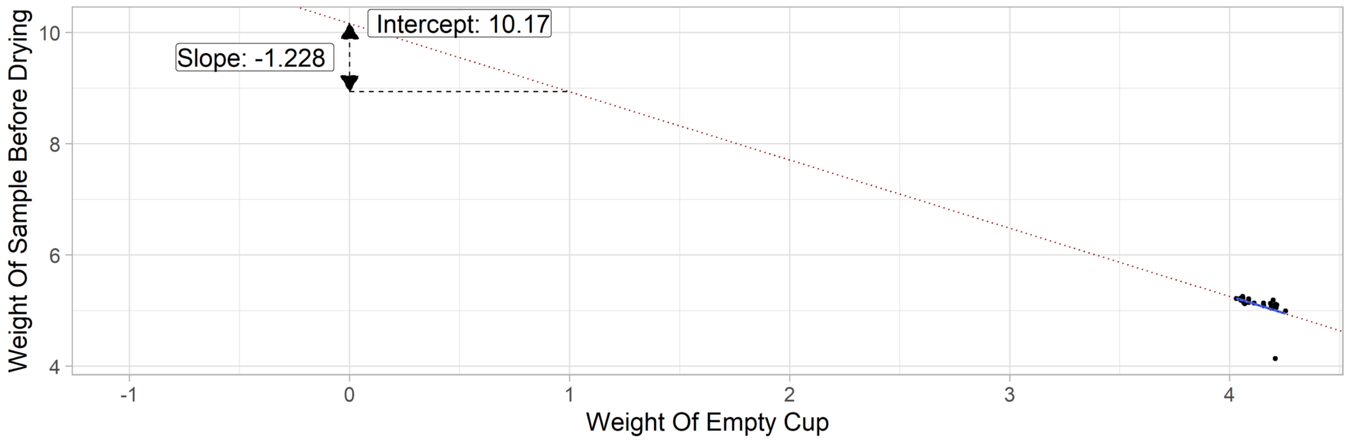
Regression: Underlying *mechanics*



Regression: Statistics

```r
lm_out <- lm(`Weight Of Sample Before Drying` ~ `Weight Of Empty Cup`,
             data=rawdata)
lm_out
```

```
Call:
lm(formula = `Weight Of Sample Before Drying` ~ `Weight Of Empty Cup`, 
    data = rawdata)

Coefficients:
          (Intercept)  `Weight Of Empty Cup`  
               10.169                 -1.228  
```



Regression: Significance

```
anova(lm_out) |> broom::tidy()
```

```
# A tibble: 2 × 6
  term                      df sumsq meansq statistic p.value
  <chr>                  <int> <dbl>  <dbl>     <dbl>   <dbl>
1 `Weight Of Empty Cup`      1 0.215 0.215       7.56  0.0100
2 Residuals                 30 0.855 0.0285     NA    NA
```

```
model_parameters(lm_out)
```

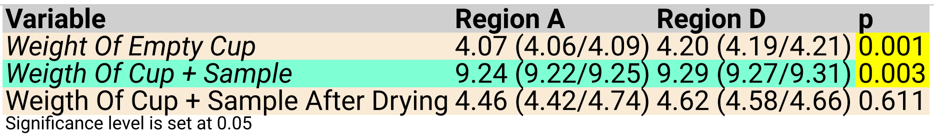
```
Parameter           | Coefficient |   SE |         95% CI | t(30) |      p
--------------------------------------------------------------------------
(Intercept)         |       10.17 | 1.84 | [ 6.41, 13.93] |  5.52 | < .001
Weight Of Empty Cup |       -1.23 | 0.45 | [-2.14, -0.32] | -2.75 | 0.010
```

Report

- *RMarkdown and quarto are powerful tools to create reports and presentations*
- Export figures: ggsave() / png() / pdf()
- Export tables: write_xlsx()
- *Package flextable provides nice features for table formatting*

Flextable example

```
test_ord |> select(-desc_all) |> rename_with(~str_remove(.,'Code Of ')) |>
  flextable() |>
  theme_zebra(even_body = 'aquamarine',odd_body = 'antiquewhite')|>
  italic(~p<=0.05,j = 1) |> bg(~p<=0.05,j = 4,bg = 'yellow') |>
  set_caption('Treatment effects, measures following a normal distribution') |>
  add_footer_lines('Significance level is set at 0.05') |>
  fontsize(size = 12,part = 'footer')
```



Variable	**Region A**	**Region D**	**p**
Weight Of Empty Cup	4.07 (4.06/4.09)	4.20 (4.19/4.21)	0.001
Weigth Of Cup + Sample	9.24 (9.22/9.25)	9.29 (9.27/9.31)	0.003
Weigth Of Cup + Sample After Drying	4.46 (4.42/4.74)	4.62 (4.58/4.66)	0.611

Significance level is set at 0.05

Useful tools along the way

- Pick columns / rows: select() / pull() / filter() / slice()
- Change format of tibble wide <--> long (e.g. for repeated measures): pivot_longer()/pivot_wider()
- Regular expressions: str_replace() / str_detect() / str_…
- Merge text elements: paste() / str_glue()
- Apply functions: purrr::map_xxx