Function; Variable; Reserved Keywords — done by: Ling 1 Dec 19

	SAS	R	SPSS
Filepath	Filepath = F:\folder\file\	F:/folder/file	
Missing entry	.	NA	.
Comment lines	/* comments */	# comments	-
Importing data	data varname; infile 'filepath' delimiter="," firstobs=2 obs=n; [obs is the final line]	var1 = read.table('filepath', header=TRUE, sep=",", col.names, row.names) [attach column names] attach(var2)	"File" -> "Open"
Header names	input col1 col2$ @@; run;	[col names] names(df1) = c('name1','name2') [row names] row.names(df1) = c('name1','name2')	*edit directly
Manual entry	datalines; data1 data2 data3 ;	[create vectors] var1 = c(data1,data2,data3); var4 = seq(from=n, to=m, by=x) [create numeric] var2 = numeric(n) [replicates elements] var3 = rep(list,times) [convert vectors to matrix] dim(vector) = c(row,col) [create matrix] var5 = matrix(vector,nrow=n,ncol=m) [create data frames] var6 = data.frame(vector/matrix) [manual entry] var7 = scan()	"Variable View" -> Define variables "Data View" -> Enter data
Fixed format entry	input col1 1-2 col2$ 4;	var1 = read.fwf('filepath',width=c(1,2,3))	"File" -> "Open" -> "Fixed width"
Creating subset	set varname; (filepath if SAS dataset) if condition1 then output;	*reassign variable	"Data" -> "Merge Files"
Filter	if condition1 then output; *note: var must be created first	var[condition,]	[Data View] "Data" -> "Select Cases" -> "If condition is satisfied" -> "If" -> enter conditions
Create new col	col_name = (a+b)/2;	[expand dataframe] var $col = var [bind a row to matrix] var3 = rbind(var1,var2) [bind a col to matrix] var3 = cbind(var1,var2)	"Edit" -> "Insert Variable"
Keeping subset	keep col1 col2 col3;	var1[rows,cols] *could be logics(conditions) too {note: when filter, apply on rows}	-
Dropping subset	drop col1 col2 col3;		"Edit" -> "Clear"

Function; Variable; Reserved Keywords — done by: Ling 1 Dec 19

	SAS	R	SPSS
Concatenating data set	set var1 var2;	[combining data frames by rows] `var3 = rbind(var1,var2)`	[Data View] "Data" -> "Merge Files" -> "Add Cases"

If-else	if condition then outcome;
	else if conditon then output;
		`if(condition) {statement}`
		`else if (condition) {statement}`
		`else {statement}`
			*applicable in filter

Loop	do var = start to end;
	statements;
	end;
	
	e.g.
	do batch = 1 to n;
	do treat = 1 to n;
	input value @@;
	output;
	end;
	end;
		[while loop] `while(condition){statements}`
		[for loop] `for (var in 1:n){statement}`
			-

Merging data	[Must be sorted by criteria first]
	merge var1 var2;
	by criteria;
		[merge data by extending cols]
		`var6 =`
		`merge(var4,var5,by='criteria',all=TRUE)`
			[sorted data] "Data View" -> "Data" -> "Merge Files" -> "Add Variables"

Output data	*change the dataname to a permanent address	[save output on screen]
		`sink('filepath') operations sink()`
		[print output] `cat(value1,value2,value3,"\n")`
		[save dataframes]
		`write.table(var1,'filepath',quote=FALSE)`
			"File" -> "Save"

Sorting	proc sort data=name;
	by descending col1;
		`var1[order(criteria),]`
		`var2[rev(order(criteria)),]`
		`rank(var)`
			"Data" -> "Sort Cases..."

Writing functions	-	`functionname = function(input1,input2){`
		statement
		`return(output)}`
			-

Transformation	*refer to create new col	*create new var *e.g. BMI = height/weigt**2	[Calculation] "Transform" -> "Compute Variable" [Recode] "Transform" -> "Recode into Different Variable" {note: Click "Output variable are strings" if output values are characters}

<table>
<tr><td colspan="3">Function; Variable; Reserved Keywords</td><td>done by: Ling 1 Dec 19</td></tr>
<tr><th></th><th>SAS</th><th>R</th><th>SPSS</th></tr>
<tr><td>Format values</td><td>proc format; *gives lables to values
value $value_name '1' = 'Name1';
value value_name 1-20 = "range"
. = "No reply"
Other = "Out of range";

data var;
label col_name = 'col name'
col_name = 'col name'; #label headers
format col_name $ value_name.
col_name $ value_name.;</td><td>#value labels
label = c("label1","label2")
var.group = label[var] #making use of indexing

#recode var
value = ifelse(test=,yes=,no=) #convert value to index</td><td></td></tr>
<tr><td>Logical Operators</td><td><table>
<tr><td>equal</td><td>=</td><td>greater than and equal</td><td>>=</td></tr>
<tr><td>less than</td><td><</td><td>not equal</td><td>^=</td></tr>
<tr><td>less than and equal</td><td><=</td><td>negation(NOT)</td><td>^</td></tr>
<tr><td>greater than</td><td>></td><td></td><td></td></tr>
</table></td><td><table>
<tr><td>equal</td><td>==</td><td>not equal</td><td>!=</td></tr>
<tr><td>less than</td><td><</td><td>negation(NOT)</td><td>!</td></tr>
<tr><td>less than and equal</td><td><=</td><td>AND</td><td>&</td></tr>
<tr><td>greater than</td><td>></td><td>OR</td><td>|</td></tr>
<tr><td>greater than and equal</td><td>>=</td><td></td><td></td></tr>
</table></td><td>-</td></tr>
<tr><td>Modify value</td><td></td><td>var[row,col] = value</td><td></td></tr>
<tr><td>Applying functions</td><td></td><td>list apply, apply function to a list, recoding the list
lapply(list, function)
table apply
tapply(vector, index, function)
*the index works like grouping variable
apply(array, margin, function)
* margin: 1 indicates rows, 2 indicates columns, c(1, 2) indicates rows and columns. Where X has named dimnames, it can be a character vector selecting dimension names.</td><td></td></tr>
</table>

<table>
<tr><td colspan="2">Function; Variable; Reserved Keywords</td><td>SAS</td><td>SPSS</td></tr>
<tr><td colspan="4">Describing numerical data</td></tr>
<tr><td>Numerical descriptive measures</td><td>- Location
- Variability
- Other measures</td><td>proc means data=var n mean std min max stderr maxdec=n;
title "this is a title";
var col_name;
(class col_name; *different groups)
OR
proc univariate data=var;
title "this is a title";
var col_name;</td><td>"Analyze" -> "Descriptive Statistics" -> "Frequencies"</td></tr>
<tr><td rowspan="3">Graphical methods</td><td>- Histogram</td><td>proc univariate data=var;
var col_name;
histogram col_name / midpoints=n to m by 10 normal;
class col_name; *seperate by group
qqplot col_name;</td><td rowspan="2">"Analyze" -> "Descriptive Statistics" -> "Explore..."
OR ~->"QQ Plots..."
"Graphs" -> "Legacy Dialogs" -> "Histogram"/"Box plot"
*add <Factor List> for 2 variable analysis</td></tr>
<tr><td>- Boxplot, Stem and leaf plot, QQ(quantile-quantile) plot</td><td>proc univariate data=var plot;
var col_name;
OR
proc boxplot data=var;
plot col_name*col_name;</td></tr>
<tr><td>- Scatter plot for bivariate data</td><td>proc sgplot data=var;
title "Title";
by col_name; #different graph
scatter x=col_name y=col_name / group=criteria; #same graph</td><td>"Graphs" -> "Chart Builder"
*need to drag x and y factors, set colour
OR in "Legacy Dialogs"</td></tr>
<tr><td rowspan="5">Robust Location estimator</td><td>- Trimmed mean</td><td rowspan="2">proc univariate data=var trimmed=0.2 winsorized=0.2;
var col_name;</td><td>-</td></tr>
<tr><td>- Winsorized mean</td><td>-</td></tr>
<tr><td>- Huber's M-estimators</td><td>-</td><td rowspan="3">"Analyze:" -> "Descriptive Statistics" -> "Explore..."</td></tr>
<tr><td>- Tukey's bisquare estimator</td><td>-</td></tr>
<tr><td>- Hampel's M-estimator</td><td>-</td></tr>
<tr><td rowspan="3">Robust measures of scale parameter</td><td>- Interquartile Range (IQR)</td><td rowspan="3">proc univariate data=var robustscale;
var col_name;</td><td></td></tr>
<tr><td>- Median Absolute Deviation (MAD)</td><td></td></tr>
<tr><td>- Gini's mean difference</td><td></td></tr>
</table>

Function; Variable; Reserved Keywords		SAS	SPSS

Categorical data	- Frequency tables	proc freq data=var;
		  title "Title";
		  tables col_name -- col_name;
			"Analyze" ->
			"Descriptive Statistics"
			-> "Frequencies"

	- Contingency tables (two-way frequency table)	tables col_name*col_name;	"Analyze" -> "Descriptive Statistics" -> "Crosstabs..." -> "Stat
	- Chi-square tests	tables col_name*col_name / chisq; weight col_name; #for manual entry summarised table	"Data" -> "Weight Cases" -> contingency
	- Charts		

Categorical data (paired data)	- McNemar's test	proc freq data=var;
		  tables before*after/agree;
		  weight count; #for frequency counts
			"Analyze" ->
			"Descriptive Statistic" ->
			"Crosstabs"

	- Bar chart	proc sgplot data=var;
		  vbar col_name;
		#for two way frequency table
		  panelby col_name;
		  vbar col_name /group=group_name;
			"Graphs" -> "Legacy Dialogs.." -> "Bar…"

Numerical data Test for location	Parametric test - One sample t-test (normal population)	proc univariate data=var mu0=value; var col_name;	"Analyze" -> "Compare Means" -> "One Sample t tests..."
	Nonparametric test - Sign test		"Analyze" -> "Nonparametric Tests" -> "Legacy Dialogs" -> "2 Related Samples" *note: must duplicate sample mean in a new col first
	- Sign rank test		

Tests comparing two groups	Two independent sample
	- Parametric Test (t-test)
		proc ttest data=var hu0=value sides=2/L/U;
		  class col_name; #two groups
		  var col_name; #values
			"Analyze" -> "Compare Means" -> Independent Sample T tests..."

Function; Variable; Reserved Keywords		SAS	SPSS
	- Nonparametric tests (Wilcoxon rank sum test) (Mann-Whitney U-test)	proc npar1way data=var wilcoxon; class col_name; #two groups var col_name; #values exact wilcoxon;	#recode into numeric variable "Transform" -> "Recode" -> "Into different variable..." #wilcoxon rank sum test "Analyze" -> "Nonparametric Tests" -> "Legacy Dialogs" -> "2 Independent Samples..."
	Related samples - Paired t-test	proc ttest data=var mu0=value sides=2/L/U; paired var1*var2; #or create a difference and run 1 var t-test	"Analyze" -> "Compare Means" -> "paired Sample T test..."
	Related samples: Nonparametric tests - Wilcoxon signed rank test	#create a difference at data import stage proc univariate data=var; var col_name;	"Analyze" -> "Nonparametric Tests" -> "Legacy Dialogs" -> "2 Related Samples..."

Tests comparing three groups or more	Parametric Test
	- ANOVA table
		proc glm data=var;
		class col_name;
		model y=x;
			"Analyze" -> "Compare Means" -> "One-way ANOVA..."
			"Options", "Descriptive"

	- LSD	means col_name/ lsd;	"Analyze" -> "Compare Means" -> "One-way ANOVA..." "Post Hoc..", "LSD"
	- Contrast	contrast "X vs. Y and Z" y_col -2 1 1; $C\colon -2\mu_x + \mu_y + \mu_z = 0$ contrast "Y vs. Z" y_col 0 1 -1; $C\colon \mu_y - \mu_z = 0$	"Analyze" -> "Compare Means" -> "One-way ANOVA..." "Contrast"
	Testing for Equal Variances - Levene's test & Bartlett test	means col_name/ hovtest = levene hovtest = bartlett;	"Analyze" -> "Compare Means" -> "One-way ANOVA..." "Options", "Homogeneity of variance test"

Function; Variable; Reserved Keywords		SAS	SPSS

	Testing for Independence and Normality Assumptions	*obtain the residuals;
		proc glm data=var;
		class col_name;
		model y=x;
		output out=var2 p=yhat r=resid;
		
		*test for normality;
		proc univariate data=var;
		var resid;
		histogrm resid/ normal noplot;
		qqplot resid;
		
		*residual plots;
		proc sgplot data=var;
		scatter x=yhat y=resid;
			"Analyze" ->"General Linear Model" -> "Univariate.." "Dependent Variable", "Fixed Factor"

	Nonparametric 1-way test - Kruskal-Wallis Test	proc npar1way data=var; class col_name; var col_name; exact wilcoxon;	"Analyze" -> "Nonparametric Tests" -> "Legacy Dialogs" -> "K Independent Samples..." "Test Variable List", "Grouping Variable" "Define Range"
Regression Analysis	Pearson Correlation Coefficient	proc corr data=var nosimple; var col_name1 col_name2 col_name3;	"Analyze" -> "Correlate" -> "Bivariate..." "Variables window", "Pearson"

	Simple regression	proc reg data=var;
		model y = x;
		output out=var p=yhat r=resid;
		quit;
			"Analyze" _> "Regression" -> "Linear"

	Testing assumptions - Linear relationship: scatter plot	proc sgplot data=var; scatter y=col_name x=col_name;	"Graphs" -> "Legacy Dialogs..." -> "Scatter/plot" "Simple Scagter" -> "Define"

Function; Variable; Reserved Keywords			SAS	SPSS

	- Normality and independence		proc sgplot data=var;
			  scatter y=resid x=yhat;
			proc sgplot data=var;
			  scatter y=resid x=height;
				“Analyze” ->
				“Regression” ->
				“Linear”
				[and save residuals]
				“Analyze” ->
				“Descriptive Statistics”
				-> “QQ plots...”

	- Prediction	Draw fitted lines	proc sgplot data=var;
			  pbspline y=col_name x=col_name;
			*pbspline = penalized beta spline method;	

		Case1: value in dataset	Add statement output out=var p=yhat r=resid and print dataset to search	“Analyze” -> “Regresion” -> “Linear” “Save”, “Unstandardized”
		Case2: value not in	Add data point into dataset with the missing values (using . for missing value)	Add missing data point

	- Transform	Quadratic	data var2;
			  set var1;
			  x2 = x**2;
			proc reg data=var;
			  model y=x x2;
				“Transform” ->
				“Compute Variable...”

		Log	data var2;
			  set var1;
			  lx = log(x);
			proc reg data=var;
			  model y=x lx;	

Function; Variable; Reserved Keywords		R	
Describing numerical data			
Numerical descriptive measures			
- Location	- Mean	mean(var)	summary(var) Min/1stQ/Median/Mean/3rdQ/Max
	- Median	median(var)	
	- Mode	var[order(var)[1:n]]#smallest 5	
- Variability	- Variance	var(var)	
	- Standard deviation	sd(var)	
	- Range	range(var)	
	- Interquartile range	IQR(var)	
	- Coefficient of Variation	cv = function(x) sd(x)/mean(x)*100	
- Other measures	- Min and Max	min(var); max(var)	
	- First & third quartile	quantile(var, level)	
	- Percentiles		
	n = length(x); mn = mean((x-mean(x)^n))		
	- Skewness [measure the symmetric]	skew = function(x) { sk = m3/m2^(3/2)*sqrt(n*(n-1))/(n-2) return(sk)}	
	- Kurtosis [measure the tail]	kurt = function(x) { ku = (n-1)/((n-2)*(n-3))*((n+1)*m4/m2^2-3*(n-1)) return(ku)}	
Graphical methods			
- Histogram	par(mfrow=c(2,1)) #2 graphs in one column in 1 page hist(var, include.lowest=TRUE, freq=TRUE, col='color', main=paste("Title"), sub=paste("sub-title"), xlab ="x", ylab ="y", axes=TRUE) *#normal curve imposed on the histogram* xpt = seq(n,m,0.1); ypt=dnorm(seq(n,m,0.1),mean(col_name),sd(col_name)) aypt=ypt*length(col_name)*10 *#scaling the line to match area of histogram, 10 is the width of bin* lines(xpt,aypt) par(mfrow=c(1,1)) #1 graphs in one column in 1 page		
- QQplot	qqnorm(col_name); qqline(col_name)		
- Boxplot	boxplot(col_name~col_name2) #~means split by group		
- Stem and leaf plot	stem(col_name)		
- Scatter plot for bivariate data	plot(x=col_name,y=col_name, xlim=c(lower,upper), ylim=c(lower,upper), axes=FALSE, pch=0, col=2) par(new=TRUE) #plot new graph and old graph together		
Robust Location estimator			
- Trimmed mean		mean(var, trim=0.2)	
- Winsorized mean			
- Huber's M-estimators			
- Tukey's bisquare estimator			
- Hampel's M-estimator			
Robust measures of scale parameter			
- Interquartile Range (IQR)		IQR(var)/1.34898	
- Median Absolute Deviation (MAD)		median(abs(x-median(x))) mad(x)	
- Gini's mean difference			
Categorical data			
- Frequency tables		table(var)	
- Contingency tables		table(var,var2)	
- Fisher test		fisher.test(table(var), alternative="less/greater/two.sided")	

Function; Variable; Reserved Keywords		R
- Chi-square tests		chisq.test(table(var), correct=F) #applied w/o continuity correction
- Charts		
Paired data	- McNemar's test	mcnemar.test(table1, correct=T)

	- Bar chart	barplot(height= c(table1),
		names.arg=c("Nam1","Name2"))
		#two-way frequency table
		barplot(tablecount, beside=T, col=c(1,8))
		legend("top-left", c("Name1","Name2"),
		fill=c(1,8))

Numerical data		
Tests for location	Parametric test - One sample t-test	t.test(var, mu=value, alterntive="two.sided/less/greater")
	Nonparametric test - Sign test	testvalue = var[var!=mu0] binom.test(sum(testvalue>mu0), length(testvalue))
	- Signed Rank test	wilcox.test(testvalue, mu=value, alternative="two.sided")

Two independent samples	- t-test	#first create different groups
		#test for variances
		var.test(var1, var2)
		#t-test
		t.test(var1,var2,mu=0,var.equal=T/F,
		alternative="two.sided")

	- Nonparametric (Wilcoxon rank sum test)	#first create different groups
		wilcox.test(groupa, groupb)
		or wilcox.test(y~x)

Two related samples	- Paired t-test	t.test(var1,var2,mu=0,paired=T)

	Nonparametric test
	- Signed test
		#create a diff first
		diff = varA-varB
		ncount = sum(diff>0)
		binom.test(ncount, length(diff), 0.5)

	- Signed rank test	wilcox.test(diff)
Three or more independent samples	Parametric test - ANOVA	model1 = anov(y~x) summary(model1) tapply(x,y,mean); mean(x); tapply(x,y,mean)- mean(x); boxplot(y~x)

	- LSD	group.1 = y[x=="1"]; group.2 = y[x=="1"];
		group.3 = y[x=="3"];
		group.mean = tapply(x,y,mean)
		treat.group = cbind(group.1, group.2, group.3)
		#d=(N-k) degree of freedom, where k is the no. of groups
		mse = sum(model1$res^2)/d
		lsd = qt(0.975,d)*sqrt(mse* (1/n1 + 1/n2))
		diff = mean(treat.group[,i]) -
		mean(treat.group[,j])
		if(abs(diff)>lsd) {cat("There is significant
		difference between groups")}

	- Contrast	contrasts(x) = matrix(c(2,-1,-1,0, -1,1),nrow=3) modelc = lm(y~x) summary(modelc)

	Testing for Equal Variances
	- Levene's test
		group.means = tapply(y,x,mean)
		gmean = group.means[x]
		#Absolute deviation from group meams is used
		words.abs.dev = abs(y-gmean)
		modellev = aov(words.abs.dev~x)
		summary(modellev)

	- Bartlett's Test	bartlett.test(y,x)

Function; Variable; Reserved Keywords		R

	Test for independence and Normality assumptions	#Test for normaity (Kolmogorove-Smirnov test)
		`resid=model1$res`
		`ks.test(resid,”pnorm”,mean(resid),sd(resid))`
		#Model checking: residual plots
		`fv = model1$fitted`
		`par(mfrow=c(2,1))`
		`plot(fv,resid); abline(h=0,lty=2)`
		`qqnorm(resid); qqline(resid,lty=2)`
		`par(mfrow=c(1,1))`

	Nonparametric 1-way Test
	- Kruskal-Wallis Test
		`kruskal.test(y,x)`
		or
		`kruskal.test(y~x)`

Simulation Study

Conditon of interest
- Compare 3 estimators for location μ through a simulation study
- 3 estimators are: sample mean, sample median, sample 10% trimmed mean
- Underlying condition: N(1,1)
- Sample size: 15
- Simulation size: 1000

Comparing the estimators	Environment set up	`ns = 1000; n = 15; mu = 1; sd = 1;`
		`meax = numeric(ns); medx = numeric(ns);`
		`trmx = numeric(ns); stdx = numeric(ns);`
		`set.seed(12345)`

	Compute the numeric value of the estimators	`for (i in 1:ns){`
		`  x = rnorm(n, mu, sd)`
		`  meax[i] = mean(x)`
		`  medx[i] = median(x)`
		`  trmx[i] = mean(x, trim=0.1)`
		`  stdx[i] = sd(x) }`

	Compute the mean of estimators	`simumean = apply(cbind(meax,medx,trmx),2,mean)`
	Compute the sd of estimators	`simustd = apply(cbind(meax, medx, trmx), 2, sd)`
	Compute bias	`simubias = simumean - rep(mu,3)`
	Compute MSE	`simumse = simubias^2 + simustd^2`

	Presentation	`sumdata = rbind(c(mu,mu,mu), ns, simumean,`
		`simustd, simubias, simumse)`
		`col.name = c(“mean”,”median”,”10% tmean”)`
		`row.name = c(“True value”, “No. of simu”, “MC`
		`Mean”, “MC sd”, “MC Bias”, “MC MSE”)`
		`dimnames(sumdata) = list(row.name,col.name)`
		`round(sumdata,4)`

Checking coverage probability of confidence interval	getting $t_{0.025,n-1}$	`t05 = qt(0.975,n-1)`
	Compute CI	`lower = meax-t05*stdx/sqrt(n)` `upper = meax+t05*stdx /sqrt(n)`
	Test logic	`coverage = sum(lower <= mu & upper >= mu)`
	Compute coverage	`coverage = coverage/ns`
Checking size of t-test	Calculate t stat	`ttests = (meanx-mu)/(stdx/sqrt(n))`
	Compute rejection	`reject = abs(ttests) > t05`
	Calculate Size	`size = sum(reject)/ns`

Checking power of t-test when $\mu_1 = \mu_0 + \frac{1}{2}\sigma$	Generate data under H_1	`ns = 1000; n = 15; mu0 = 1; sigma = 1;`
		`mu1 = mu0 + 0.5*sigma`
		`meanx = numeric(ns); stdx = numeric(ns);`
		`set.seed(12345)`
		`for (i in 1:ns){`
		`  x = rnorm(n, mu1, sigma)`
		`  meanx[i] = mean(x)`
		`  stdx[i] = sd(x) }`

Function; Variable; Reserved Keywords		R
	Calculate t stat	`t05 = qt(0.975,n-1)` `ttests = (meanx-mu0)/(stdx/sqrt(n))`
	Compute rejection	#alternate method of calculating rejection `reject = ttests >= t05 | ttests <= -t05`
	Calculate power	`power = sum(reject)/ns`
Bootstrap method		
Generate bootstrap samples	Compute true value	`options(digit=4)` `theta.hat = corr(var1,var2)`

	Generate bootstrap samples
	& compute stat of interest
	(corr in this case)
		`B = 1000; n = nrow(data);`
		`theta.b = numeric(B);`
		`for (b in 1:B){`
		`  index = sample(1:n, size=n, replace=TRUE)`
		`  bsample1 = var1[index]`
		`  bsample2 = var2[index]`
		`  theta.b[b] = corr(bsample1, bsample2)`

Bootstrap Estimation of Standard Error	Compute sd	`sd(theta.b)`
Bootstrap Estimation of Bias	Compute bias	`bias = mean(theta.b) - theta.hat`

Alternate method for SE and Bias	Using Library boot	`bcor = function (data, b) {`
		`  return(cor(data[b,1],data[b,2])) }`
		`library(boot)`
		`boot.cor = boot(data, statistic=bcor, R=5000)`
		#gives original value from sample, bias and standard error

Basic bootstrap Confidence interval	Compute lower and upper quantile	`alfa = 0.05`
		`low = quantile(theta.b,alfa/2)`
		`high = quantile(theta.b,1-alfa/2)`

	Compute CI	`lower = 2*theta.hat-high` `upper = 2*theta.hat-low`

Percentile bootstrap confidence interval	Compute lower and upper quantile	`alfa = 0.05`
		`low = quantile(theta.b,alfa/2)`
		`high = quantile(theta.b,1-alfa/2)`

Alternate method for CI	Using Library boot	`bcor = function (data, b) {`
		`  return(cor(data[b,1],data[b,2])) }`
		`library(boot)`
		`boot.cor = boot(data, statistic=bcor, R=5000)`
		`boot.ci(boot.cor,type=c("basic","norm","perc"))`
		#gives normal, basic and percentile CI
		#must generate boot.cor object first

Numerical methods in R		

Finding Root	One-dimension
	*uniroot only finds 1 root in region
	*function must have opposite signs in two end points
		`a = 0.5; n = 20;`
		`fy = function(y) {a^2+y^2+2*a*y/(n-1)-(n-2)}`
		`rt = uniroot(fy,lower=0,upper=100)`
		`root = rt$root` #to get root
		`value = rt$f.root` #to get the value of fn

	Finding roots
	*polyroot finds multiple roots, including complex roots
		#input the coefficient of polynomial
		`a0 = a^2 - (n-2); a1 = 2*a/(n-1); a2 = 1;`
		`polyroot(c(a0,a1,a2))`
		#a_i is the coefficient of x^i

Integration	One variable integral	`fcn = function(y,mu,sigma) {`
		`abs(y)*exp(-(y-mu)^2/(2*sigma^2)) /`
		`sqrt(2*pi)/sigma }`
		`integrate(fcn,lower=-Inf, upper=Inf, mu=0,`
		`    sigma=1)`

Optimiation	One-dimentional	`f = function(x) {log(x+log(x))/log(1+x)}` `optimize(f,lower=4,upper=8,maximum=TRUE)`

Function; Variable; Reserved Keywords		R

	Two-dimensional
	*"optim" performs minimization by default
		`LL = function(theta, sx, slogx, n) {`
		`  r = theta[1]; lambda = theta[2]`
		`  loglik = n*r*log(lambda)+(r-1)*slogx-`
		`        lambda*sx-n*log(gamma(r))`
		`  return(loglik)}`
		#or `return(-loglik)` if lazy to use fnscale
		
		`n = 200; r = 5; lambda = 2;`
		`theta.start = c(1,1)`
		`x = rgamma(n,shape=r,rate=lambda)`
		`out = optim(par=theta.start,fn=LL,sx=sum(x),`
		`        slogx=sum(log(x)),`
		`        n=n,control=list(fnscale=-1))`
		`r = out$par[1]; lambda = out$par[2]`
		`result = out$value`

Regression Analysis		

Pearson Correlation Coefficient		`cor(dataframe, method="person")`
		#test if true correlation is 0
		`cor.test(var1,var2)`

Simple regression		`model1 = lm(y~x)`
		#to see sum of squared regression and error
		`anova(model1)`
		#to see R-squared and linear model, F-test
		`summary(model1)`

Testing assumptions
- Linear relationship: scatter plot
		`model1 = lm(y~x)`
		`par(mfrow=c(2,2))`
		#scatter plot
		`plot(y~x); abline(model1);`
		`title("Scatter plot and Regression Line")`

- Normality and independence
observe if there's any trend in residuals, it shouldn't have
		#residual plot
		`rs = model1$resid; fv = model1$fitted;`
		`plot(rs~x);abline(h=0)`
		#normal QQ plot
		`qqnorm(rs); qqline(rs)`
		`par(mfrow=c(1,1))`

- Prediction	case1: data in set	`model1$coeff[1]+value*model1$coeff[2]`
	case2: data not in	`predict(model1,data.frame(x=value))` #or the previous method
- Transformation	Investigate rs	`plot(x,y);lines(smooth.spline(x,y))`

	quadratic	`x2 = x^2;`
		`model2 = lm(y~x+x2)`
		`anova(model2);summary(model2)`

	log	`lx = log(x);`
		`model3 = lm(y~lx)`
		`anova(model3);summary(model3)`

Function; Variable; Reserved Keywords — done by: Ling 1 Dec 19

		SAS	R
Simulation			

Generate random numbers	Linear congruential generators	data try1;
		seed = 1234;
		do i = 1 to 10;
		x = ranuni(seed);
		output;
		end; keep x;
			ran = NULL
			for (i in 1:n){
			x1 = (a*x0+c)%%m
			x0 = x1
			ran = c(ran,x1/m)}

	Uniform random numbers	data unif;
		seed = 1234; call streaminit(seed);
		n=100; a=0; b=10;
		do i = 1 to n;
		x = a + (b-a) * rand(“UNIFORM”); output;
		end; keep x;
			#uniform with (0,1)
			x = runif(1000)
			#uniform with (a,b)
			set.seed(1234)
			x = runif(n,a,b)

	Normal(μ, σ^2)	data norm;
		seed = 1234; call streaminit(seed);
		n=100; mu=0; sigma=1;
		do i = 1 to n;
		x = rand(“NORMAL”, mu, sigma); output;
		end; keep x;
			x = rnorm(n,mean=mu,sd=sigma)

	Exponential(λ)	data expno;
		seed = 1234; call streaminit(seed);
		n=100; lambda=5;
		do i = 1 to n;
		x = lambda * rand(‘EXPONENTIAL’); output;
		end; keep x;
			x = rexp(n,mean=lambda)

	Gamma(α, β)	data gammano;
		seed = 1234; call streaminit(seed);
		n=100; alpha=1; beta=2;
		do i = 1 to n;
		x = beta * rand(‘GAMMA’, alpha); output;
		end; keep x;
			x = rgamma(n,shape=alpha,scale=beta)

	Chi-square(p)	data chisqno;
		seed = 1234; call streaminit(seed);
		n=100; df=10;
		do i = 1 to n;
		x = rand(‘CHISQUARE’, df); output;
		end; keep x;
			x = rchisq(n,df=p)

	Beta(α, β)	data betano;
		seed = 1234; call streaminit(seed);
		n=100; alpha=2; beta=3;
		do i = 1 to n;
		x = rand(‘BETA’, alpha, beta); output;
		end; keep x;
			`x = rbeta(n,shape1=a,shape2=b)`

	t(k)	data tno;
		seed = 1234; call streaminit(seed);
		n=100; df=5;
		do i = 1 to n;
		x = rand(‘T', df); output;
		end; keep x;
			`x = rt(n, df=k)`

	F(m,n)	data fno;
		seed = 1234; call streaminit(seed);
		n=100; df1=5; df2=10;
		do i = 1 to n;
		x = rand(‘F', df1, df2); output;
		end; keep x;
			`x = rf(n, df1 = n1, df2 = n2)`

	Binomial(n,p)	data binomno;
		seed = 1234; call streaminit(seed);
		ns=100; n=10; p=0.3;
		do i = 1 to ns;
		x = rand(‘BINOMIAL’, p, n); output;
		end; keep x;
			`x = rbinom(n, size, prob = p)`

	Poisson(λ)	data poisno;
		seed = 1234; call streaminit(seed);
		n=100; lambda=3;
		do i = 1 to n;
		x = rand(‘POISSON’, lambda); output;
		end; keep x;
			`x = rpois(n, lamba)`

	Hypergeo(n,N,S)	data hypergeono;
		seed = 1234; call streaminit(seed);
		ns=100; popnsize=3; numbsucc=5; samplesize=3;
		do i = 1 to ns;
		x = rand(‘HYPERGEOMETRIC’, popnsize, numbsucc, samplesize);
		output;
		end; keep x;
			`x = rhyper(nn, m=S, n=N, k=n)`

	NBinom(r,p)	data negbinomno;
		  seed = 1234; call streaminit(seed);
		  n=100; p=0.3; k=5;
		  do i = 1 to n;
		   x = rand('NEGBINOMIAL', p, k); output;
		  end; keep x;
			`x = rnbinom(n, size=r, prob=p)`

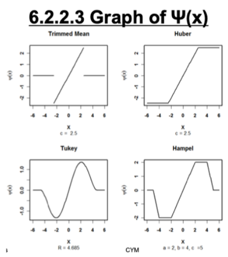
Simulation Study
Conditon of interest - Compare 3 estimators for location μ through a simulation study - 3 estimators are: sample mean, sample median, sample 10% trimmed mean - Underlying condition: N(1,1) - Sample size: 15 - Simulation size: 1000

Size of t-test $\alpha\ level$	Generate sample set	data simu;
		  seed = 123; ns=1000; n=15; mu=0; sigma=1;
		  call streaminit(seed);
		  do mcrep = 1 to ns;
		   do i = 1 to n;
		    value = rand("NORMAL",mu,sigma); output;
		   end;
		  keep mcrep value;
		  end;	

	Calculate t test	proc sort data=simu;
		  by mcrep;
		proc univariate data=simu noprint mu0=0;
		  by mcrep;
		  var value;
		  output out=outtest probt = p;	

	Calculate rejection	data outtest1;
		  set outtest;
		  reject = (p<0.05);	

	Calculate rejection rate	proc means data=outtest1 noprint;
		  var reject;
		  output out = results mean=rejrate;
		proc print data=results;
		  var _freq_ rejrate;	

done by: Ling 1 Dec 19			
Describing numerical data			
Numerical descriptive measures	- Location	- Mean - Median - Mode	
	- Variability	- Variance or standard deviation - Range - Interquartile range - Coefficient of Variation	
	- Other measures unbiased estimator of kurtosis: $\frac{(n-1)}{(n-2)(n-3)}\left(\frac{(n+1)m_4}{{m_2}^2} - 3(n-1)\right)$	- Minimum and Maximum - First quartile and third quartile - Percentiles - Skewness [measure the symmetric] - Kurtosis [measure the tail]	unbiased estimator of skewness: $\frac{\sqrt{n(n-1)}}{n-2}\left(\frac{m_3}{{m_2}^{3/2}}\right)$
Graphical methods	- Histogram		
	- Boxplot, Stem and leaf plot		
	- Scatter plot for bivariate data		
Robust Location estimator			
Robust location estimators $\sum \psi(y_i-\mu)=0$	- Trimmed mean (top-left) - Winsorized mean - Huber's M-estimators (top-right) - Tukey's bisquare estimator (bottom-left) - Hampel's M-estimator (bottom-right)	mean: $\psi(x) = x$ median: $\psi(x) = sign(x)$ trimmed: $\psi(x) = x \; \lvert x \rvert < c, else\ 0$ winsorized: $\psi(x) = -c, x < -c\ e.g$ turkey: $\psi(x) = x[1-(\frac{x}{R})^2]^2$ for $\lvert X \rvert<R$	
Robust measures of scale parameter	- Interquartile Range (IQR)	$Normal: \hat{\sigma} = IQR/1.34898$	
	- Median Absolute Deviation (MAD) (median of the diff in median)	$MAD = median_i(\lvert y_i - median_j(y_j)\rvert)$ $Normal: \hat{\sigma} = 1.4826 \times MAD$	
	- Gini's mean difference	$G = \frac{1}{\binom{n}{2}}\sum_{i<j}\lvert y_i - y_j\rvert$ $\sigma \approx \frac{G}{2}\sqrt{\pi}$ e.g. G = (a1 + .. + an)/n	
Categorical data			
	- Frequency tables		
	- Contingency tables		
	- Chi-square tests (single tail test)	$H_0: \mu_1 and\ \mu_2\ are\ independent$ $H_1: \mu_1 and\ \mu_2\ are\ not\ independent$	
	- Charts		
Paired Data	Same subjects under two different conditions		
	- McNemar's test	$H_0: before = after$ $H_1: before \neq after$	
	- Bar Chart		
Numerical data			
One Sample Tests (Hypothesis testing)			
Location	Parametric test - One sample t-test (normal population)	$\mu_0 = 0$ $\mu_0 \neq 0$	

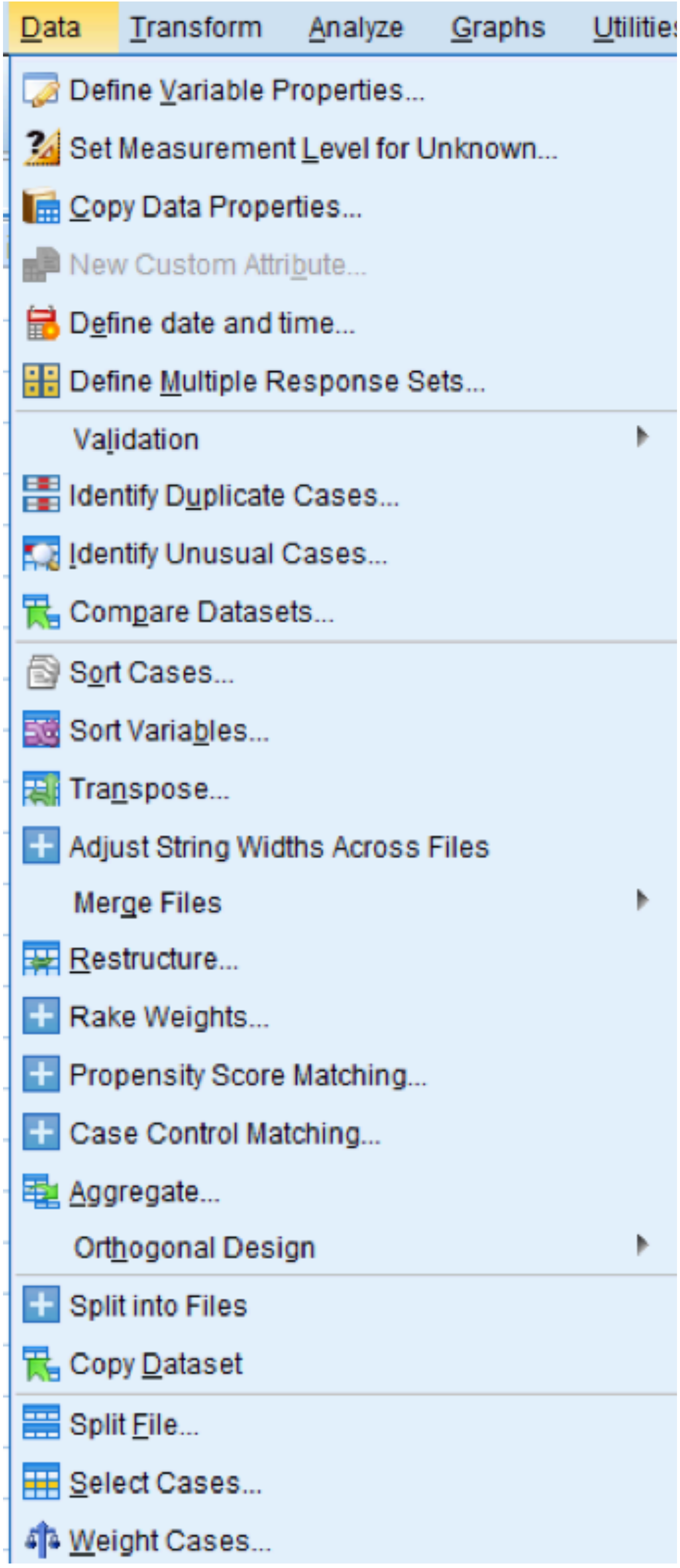
<table>
<tr><td></td><td>Nonparametric tests
- Sign test
- Sign rank test</td><td>$\mu_0 = 0$
$\mu_0 \neq 0$</td></tr>
<tr><td colspan="3">Two comparing two groups (Hypothesis testing)</td></tr>
<tr><td rowspan="2">Independent samples</td><td>Two-sample t-test (normal population)</td><td rowspan="2">$\mu_0 = \mu_1$
$\mu_0 \neq \mu_1$</td></tr>
<tr><td>Wilcoxon rank-sum test (non-parametric)</td></tr>
<tr><td rowspan="2">Related samples</td><td>Paired t-test (normal population)</td><td rowspan="2">$\mu_{before} = \mu_{after}$
$\mu_{before} \neq \mu_{after}$</td></tr>
<tr><td>Sign Test or Wilcoxon signed rank test(non-parametric)</td></tr>
<tr><td colspan="3">3 or more independent groups (One-way analysis of variance)</td></tr>
<tr><td rowspan="5">Parametric method</td><td>One-way analysis of variance
- ANOVA</td><td>$\mu_x = \mu_y = \mu_z$
$\mu_x \neq \mu_y \neq \mu_z$</td></tr>
<tr><td>Assumptions of ANOVA (based on F-test)
& Model Checking</td><td>1. Random sample
2. Equal variances for all the groups
  a. Bartlett test or Levene's test
3. Independence of errors
  a. Residual plots
    i. QQplot on residuals
    ii. Plot residuals against the groups
4. Normal distribution of errors
  a. Normality test on residuals
5. Additivitiy of treatment effects</td></tr>
<tr><td>Multiple comparison tests
- Least significant difference (LSD)</td><td>LSD for $\mu_i - \mu_j$
$$t_{\frac{\alpha}{2},N-k}\sqrt{MSE(\frac{1}{n_1}+\frac{1}{n_2})}$$
We conclude μ_i is different from μ_j if
$$\left|\bar{X}_i - \bar{X}_j\right| > LSD$$</td></tr>
<tr><td>- Contrast method</td><td>A contrast of means is a linear combination of the means such that the coefficients of the means such that the coefficients of the means should sum up to zero
e.g. $C_1 = \mu_2 - \mu_3$, $C_2 = 2\mu_1 - \mu_2 - \mu_3$
$$H_0: C_1 = 0 \; and \; C_2 = 0$$</td></tr>
<tr><td colspan="2">Others
- Duncan's multiple-range test
- Student-Newman-Keul's multiple-range test
- Scheffe's multiple-comparison procedure</td></tr>
<tr><td>Nonparametric method</td><td>Kruskal-Wallis test</td><td></td></tr>
<tr><td colspan="3">Simulation</td></tr>
<tr><td>Usage</td><td>- Checking distribution theory</td><td><u>Theory:</u>
Take a sample of size 4 from normal distribution with mean 0
$$T = \frac{\bar{X}}{S/\sqrt{4}} \sim t(3)$$
<u>Simulation:</u>
Generate 1000 random samples with size 4. Compute statistic T and construct a historygram.
Superimpose the t(3) curve and compare</td></tr>
</table>

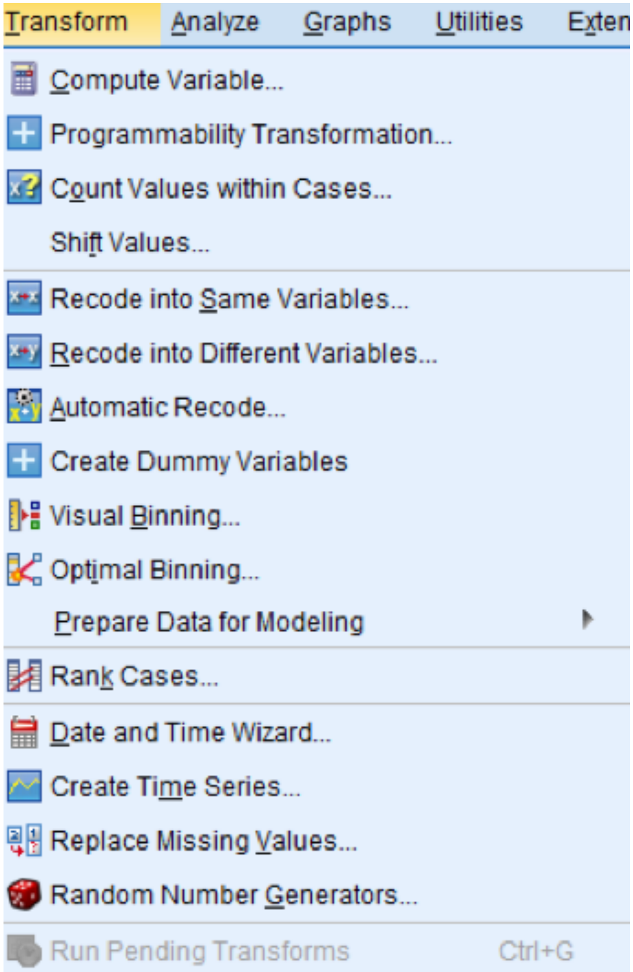
	- Comparing estimators	Question: Comparing robustness of estimators in different conditions (underling distributions) Consider $MSE = Bias^2 + Variance$ Simulation: Generate random samples. Compute MSE for each estimators.
	- Buffon's needle experiment	Question: Compute a question of interest through a function. Consider the probability that needle intersects a line $theory: \frac{2/l}{\pi d}$ Simulation: Simulate the experiment by N times and count the chances of success n/N give the estimate of the probability of success
Generate numbers		
Random number	- Congruential generators	$def: X_i = (aX_{i-1} + c)\ mod\ M$ Uniform random numbers: $U_i = X_i/M$
Uniform numbers	- Uniform random numbers	
Non-uniform random numbers	- Inversion method	If X has a continuous distribution function F(x), where $F(x) = \Pr(X \leq x)$, then $F(x) \sim Uniform(0,1)$ Generate U from Uniform(0,1) and set $X = F^{-1}(U)$
	- Exponential distribution	$Let\ U = F(X) = 1 - e^{-x/\lambda}$ $Then\ X = F^{-1}(U) = -\lambda \log(1-U)$
	- Weibull distribution	$F(x) = 1 - \exp(-x^{\beta}), D = (0, \infty)$ $X = (-\log(1-U))^{1/\beta}$
	- Cauchy distribution	$F(x) = \frac{1}{2} + \frac{1}{\pi} tan^{-1}(\frac{x-\mu}{\sigma})$ $X = \sigma \tan[\pi(U - 0.5)] + \mu$
Generate Normal random variable	- Box-Muller Algorithm	Generate U_1 and U_2 from uniform(0,1) $set\ \theta = 2\pi U_1; R = (-2logU_2)^{1/2}$ $set\ X = R\cos(\theta)\ and\ Y = R\ sin(\theta)$ Now we have X and Y, two independent standard normal variables
Random variable from other random variables	- Cauchy distributions	$Y, Z\ independent\ and\ \sim N(0,1)$ $X = \frac{Y}{Z} \sim Cauchy(0,1)$ $If\ Y \sim N(\mu, \sigma^2)\ and\ Z \sim N(0,1)$ $X \sim Cauchy(\mu, \sigma^2)$
	- Chi-square distribution	$If\ Y \sim N(0,1)$ $X = Y^2 \sim \chi^2(1)$ $If\ Y_1 \ldots Y_n\ are\ i.i.d\ and\ X = Y_1^2 + .. + Y_n^2$ $X \sim \chi^2(n)$
	- Student's t-distribution	$If\ Y \sim N(0,1) and\ Z \sim \chi^2(p), then$ $X = \frac{Y}{\sqrt{\left(\frac{Z}{p}\right)}} \sim t(p)$
	- F distribution	$Y \sim \chi^2(m) and\ Z \sim \chi^2(n)$ $X = \frac{Y/m}{Z/n} \sim F(m,n)$
Functions	Other functions to generate the random no.	https://stat.ethz.ch/R-manual/R-devel/library/stats/html/Distributions.html http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/viewer.htm#a001466748.htm
Generate random numbers	Uniform(a,b)	$f(x) = \frac{1}{b-a}, a < x < b$
	Normal(μ, σ^2)	$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), -\infty < x < \infty$

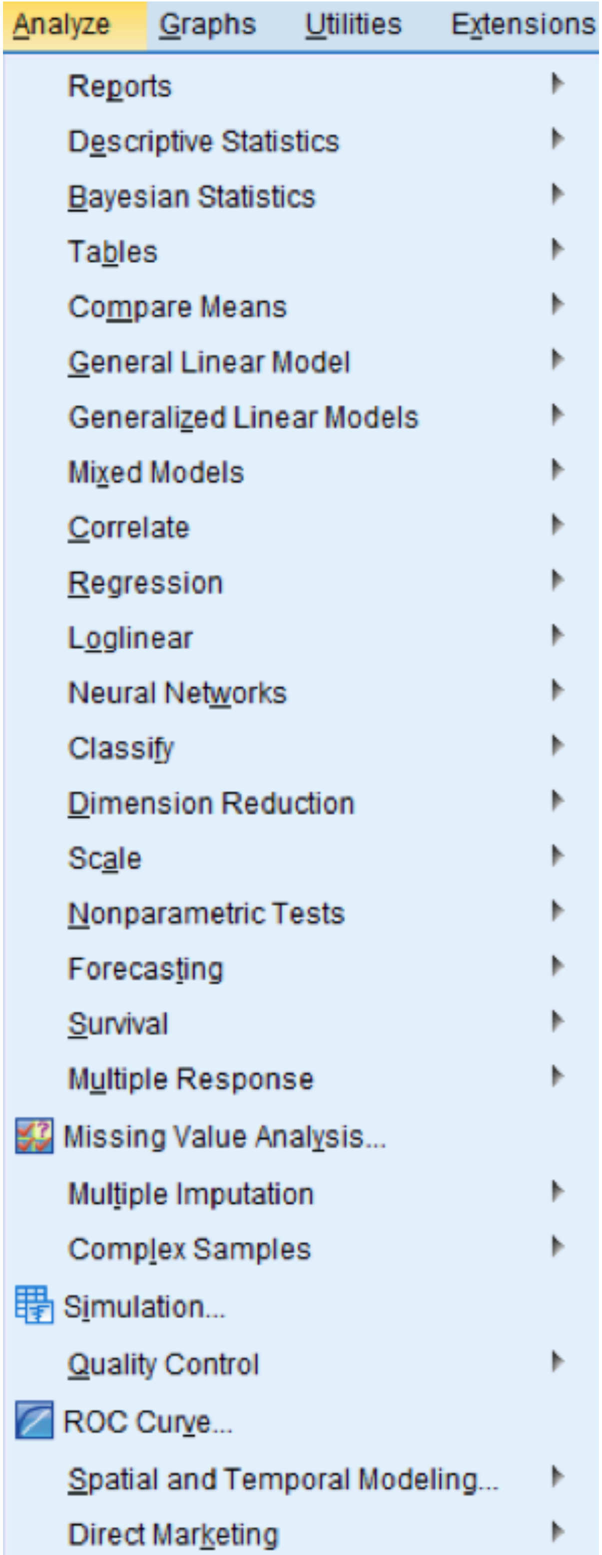
	Exponential(λ)	$f(x)=\frac{1}{\lambda}\exp\left(-\frac{x}{\lambda}\right), x>0$
	Gamma(α,β)	$f(x)=\frac{1}{\beta^{\alpha}\Gamma(\alpha)}x^{\alpha-1}\exp\left(-\frac{x}{\beta}\right), x>0$
	Chi-square(p)	$f(x)=\frac{1}{2^{\frac{p}{2}}\Gamma(\frac{p}{2})}x^{\frac{p}{2}-1}\exp\left(-\frac{x}{2}\right), x>0$
	Beta(α,β)	$f(x)=\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}x^{\alpha-1}(1-x)^{\beta-1}, for\ 0<x<1$
	t(k)	$f(x)=\frac{\Gamma(\frac{k+1}{2})}{\Gamma(\frac{k}{2})}\frac{1}{\sqrt{k\pi}}\frac{1}{\left(1+\frac{x^2}{k}\right)^{\frac{k+1}{2}}}, for -\infty<x<\infty$
	F(m,n)	$f(x)=\frac{\Gamma\left(\frac{n_1+n_2}{2}\right)}{\Gamma\left(\frac{n_1}{2}\right)\Gamma\left(\frac{n_2}{2}\right)}\left(\frac{n_1}{n_2}\right)^{\frac{n_1}{2}}\frac{x^{\frac{n_1}{2}-1}}{\left(1+\frac{n_1}{n_2}x\right)^{\frac{n_1+n_2}{2}}}, 0<x<\infty$
	Binomial(n,p)	$f(x)=\binom{n}{k}p^x(1-p)^{n-x}, for\ x=0,1,\dots,n$
	Poisson(λ)	$f(x)=\frac{e^{-\lambda}\lambda^x}{x!}, x=0,1,2,\dots$
	Hypergeo(n,N,S)	$f(x)=\frac{\binom{S}{x}\binom{N-S}{n-x}}{\binom{N}{n}}, for\ x=0,1,..,\min(n,S)$
	NBinom(r,p)	$f(x)=\binom{r+x-1}{x}p^r(1-p)^x, for\ x=0,1,2,\dots$
Simulation Studies in Stats		
Rationale (in statistics)	Properties of estimators	- Bias - Consistent - Sampling variance - Comparison with competing estimators on bias, precision etc - Does confidence interval achieve the advertised nominal level of coverage (e.g.95%)?
	Properties of hypothesis testing procedures	- Does hypothesis testing procedure attain the advertised level of significance or size (e.g. α = 0.05)? - What power is possible against different alternatives to the null hypothesis? Do different test procedures deliver different power?
Properties of estimators	Monte Carlo simulation approximation	- Generate S independent data sets under the conditions of interest - Compute the numerical value of the estimators - If S large enough, summart statistics should a good approximations to the true properties of the estimator <u>under the conditions of interest</u>
	Checking the coverage probability of confidence interval	$(\bar{Y}-t_{0.025,n-1}\frac{S}{\sqrt{n}},\bar{Y}+t_{0.025,n-1}\frac{S}{\sqrt{n}})$
Properties of hypothesis tests	Testing size/ level of test	Generate data under $H_0: \mu=\mu_0$ Calculate the proportion of rejections of H_0 Approximate the true probability of rejecting H_0 when it is true. Proportion should be $\approx\alpha$
	Evaluate power	Generate data under $H_1: \mu\neq\mu_0$ Calculate the proportion of rejections of H_0 Approximate the true probability of rejecting H_0 when it is false.
Bootstrap Method		
Bootstrap distribution	Empiritcal distribution	$f_n(x)=\begin{cases}1/n, & for\ x=x_1\dots x_n\\ 0, & otherwise\end{cases}$
	Empirical cumulative distribution	$F_n(t)=\frac{\#\ of\ x's<t}{n}$

Method	Estimate distribution function $F_{\bar{\theta}}(\cdot)$	Generate bootstrap replicate $x^{*(b)} = (x_1{}^*, \dots, x_n{}^*)$ into $\hat{\theta}^{*(b)} = (\hat{\theta}_1{}^*, \dots, \hat{\theta}_n{}^*)$ Calculate bootstrap estimate of $F_{\bar{\theta}}(\cdot)$
Statistics	Standard Error	$\widehat{se}\left(\hat{\theta}^*\right) = \sqrt{\frac{1}{B-1}\sum_{b=1}^{B}(\hat{\theta}^{*(b)} - \overline{\hat{\theta}^*})^2}$ where $\overline{\hat{\theta}^*} = \frac{1}{B}\sum_{b=1}^{B} \hat{\theta}^{*(b)}$
	Bias	$bias\left(\hat{\theta}\right) = E\left(\hat{\theta} - \theta\right) = E\left(\hat{\theta}\right) - \theta$ $\widehat{bias}\left(\hat{\theta}\right) = \overline{\hat{\theta}^*} - \hat{\theta}$
	Confidence interval	Basic $(2\hat{\theta} - \hat{\theta}^*{}_{1-\alpha/2}, 2\hat{\theta} - \hat{\theta}^*{}_{\alpha/2})$ Percentile $(\hat{\theta}^*{}_{\alpha/2}, \hat{\theta}^*{}_{1-\alpha/2})$ where $\hat{\theta}^*{}_{\alpha}$ is the α sample quantitle from the empirical cdf of the replicates $\hat{\theta}^*$
	Numerical methods in R	
Root-finding	One-dimension	e.g. solve for y in equation $a^2 + y^2 + \frac{2ay}{n-1} = n - 2, a = 0.5, n = 20$
Integration	One-variable	e.g. solve for the integral $\int \frac{1}{\sqrt{2\pi\sigma}} \exp(-\frac{(y-\mu)^2}{2\sigma^2})$
Maximum Likelihood Method	Concept	A method to estimate the parametes of a distribution, the parameter should maximizes the likelihood function $\max L(\theta) = f(x_1, x_2, \dots, x_n \vert \theta)$ Assuming independent and identically distributed random values $f(x_1, x_2, \dots, x_n \vert \theta) = \prod_{i=1}^{n} f(x_i \vert \theta)$ Since population parameter θ is unknown, we estimate with $\hat{\theta}$ In reality the product function is difficult to differentiate, so we take log of the function
	Method1: finding θ which maximizes $L(\theta) or\ l(\theta)$	Newton's method to find the value which maximises the function
	Method2: finding the first order condition	$\frac{d}{d\theta}L(\theta) = 0\ or\ \frac{d}{d\theta}l(\theta) = 0$
Optimization	One-dimension	e.g. Maximize the fuction $f(x) = \frac{\log(1 + \log(x))}{\log(1 + x)}, wrt\ x\ and\ x > 0$
	Two-dimension	e.g. Find the maximum likelihood estimator of r and λ $\max L(r, \lambda) = \frac{(\lambda)^{nr}}{\Gamma(r)^n}\prod_{i=1}^{n} x_i^{r-1} \exp\left(-\lambda\sum_{i=1}^{n} x_i\right), x_i \geq 0$ or consider the log-likelihood function $\max\ l(r, \lambda) = nr\ log\lambda - nlog\Gamma(r) + (r-1)\sum_{i=1}^{n} logx_i - \lambda\sum_{i=1}^{n} x_i$
	Regression Analysis	
Correlation	Pearson Correlation Coefficient - Indicates the strength of linear relationship between two variabels	$\rho = \frac{Cov(X,Y)}{\sqrt{Var(x)}\sqrt{Var(Y)}}$ $\rho = \frac{E[(X-\mu_x)(Y-\mu_y)]}{\sqrt{E[(X-\mu_x)^2}\sqrt{E[(Y-\mu_y)^2]}}$

Sample regression	Linear model establish a relationship between two variables	$Y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$ where $\varepsilon_i \sim N(0, \sigma^2) independently$ $E(Y_i\|x_i) = \beta_0 + \beta_1 x_i$ $Var(Y_i\|x_i) = Var(\varepsilon_i) = \sigma^2$
	Assumptions	- Y_i's are independent - Y_i's follow normal distribution with given parameters Study residuals - Y_i's have the same variances Study residuals - Linear relationship exist between Y and X Test using scatter plot
	Prediction	Case1: values in the dataset Case2: values not within the dataset
	Transformation	- Quadratic term to the regression line - Log transformation
Bonus		
Machine Learning	Extra	https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/


Data
Transform
Analyze
Graphs
Utilities
Define Variable Properties...
Set Measurement Level for Unknown...
Copy Data Properties...
New Custom Attribute...
Define date and time...
Define Multiple Response Sets...
Validation
Identify Duplicate Cases...
Identify Unusual Cases...
Compare Datasets...
Sort Cases...
Sort Variables...
Transpose...
Adjust String Widths Across Files
Merge Files
Restructure...
Rake Weights...
Propensity Score Matching...
Case Control Matching...
Aggregate...
Orthogonal Design
Split into Files
Copy Dataset
Split File...
Select Cases...
Weight Cases...


Transform
Analyze
Graphs
Utilities
Exten
Compute Variable...
Programmability Transformation...
Count Values within Cases...
Shift Values...
Recode into Same Variables...
Recode into Different Variables...
Automatic Recode...
Create Dummy Variables
Visual Binning...
Optimal Binning...
Prepare Data for Modeling
Rank Cases...
Date and Time Wizard...
Create Time Series...
Replace Missing Values...
Random Number Generators...
Run Pending Transforms
Ctrl+G


Analyze
Graphs
Utilities
Extensions
Reports
Descriptive Statistics
Bayesian Statistics
Tables
Compare Means
General Linear Model
Generalized Linear Models
Mixed Models
Correlate
Regression
Loglinear
Neural Networks
Classify
Dimension Reduction
Scale
Nonparametric Tests
Forecasting
Survival
Multiple Response
Missing Value Analysis...
Multiple Imputation
Complex Samples
Simulation...
Quality Control
ROC Curve...
Spatial and Temporal Modeling...
Direct Marketing