

I. Q. 2 - Tests of independence.

* Chi-square test.

* Fisher's exact test.

* Cochran - Mantel-Haenszel test.

I. Q. 3. - Measures of association :-

* associations().

I. 3.1. Types of correlation

* Pearson, Spearman, Kendall, cor().

* partial correlation, pcor().

I. 3.2. Testing correlations for significance:

(*) ~~corr.test()~~, ~~pcor.test()~~, cor.test().

(*) corr.test().

(*) pcor.test().

T.4. T-Test

T.4.1. Independent T-Test.

* `t.test()`

~~paired~~

T.4.2. Dependent T-Test

`t.test(y1, y2, paired=TRUE)`,

I.5. Nonparametric Tests

T.5.1. Comparing 2 groups.

* `wilcox.test()`,

* `wilcox.test(x, y, paired=TRUE)`,

T.5.2. Comparing more than 2 groups:-

* `kruskal.test()`

* `friedman.test()`,

8. Regression,

2.1. Fitting regression models with lm(),

* lm().

* Table 8.2, symbols commonly used in
~~Regressions, R formulas.~~

* summary(), coefficients(), confint(),
fitted(), residuals(), anova(),
vcov(), AIC().

plot() - generate diagnostic plots for
evaluating fit of model.

vcov() - lists covariance matrix for
model parameters.

8.3. Regression diagnostics

* plot(lm).

* qqplot() - Quantile comparison plot(),

* durbinWatsonTest() - for autocorrelated
errors

* ~~cple~~ cple() - component plus residual
plots.

- * knitrTest() - F-test for nonconstant error variable.
- * spreadLevelPlot()
- * outlierTest() - Bonferroni outlier test.
- * arplots() - added variable plots.
- * influencePlot() - regression influence plots.
- * vif() - variation inflation factors.

Normality

- * qqplot()

Independence of errors

- * durbinWatsonTest(),

Linearity:

- * rypplots(),

Homoscedasticity - (constant error variance).

- * knitrTest(),

- * spreadLevelPlot(),

library(glmma).

18.3.31.

variable \leftarrow glmma(pp-f).

* performs a global validation of ~~linear~~ linear model assumptions as well as separate evaluations of skewness, kurtosis, and heteroscedasticity.

8.3.4. Multicollinearity, detection

* vif()

* $\sqrt{vif} > 2$, indicates multicollinearity.

8.4.1: Outliers

* outlierTest().

8.4.2: High leverage points: (hat statistics)

8.4.3: Influential observations:

* Cook's distance.

* covPlots(), influencePlot()

8.5.2: Transforming variables:

- ① When model violates normality assumption.
* powerTransform().
- ② When model violates linearity assumption.
* boxTidwell().
- ③ When model violates homoscedasticity.
* spreadLevelPlot().

Comparing Models:- 8.6.1

* Akaike().

* AIC().

Variable Selection. 8.6.2.

- ① Stepwise Regression.

* StepAIC().

All subsets Regression

- * regsubsets(), in leaps package
- * subsets() in crrs package.

Scal., Cross Validation.

Ridge, Relative Importance.

Collinearity

Correlation matrix

One small group fully collinear but no
other groups are partially collinear

Partial Correlation

selection of the best feature (Lasso)
after the regularization, the first few
variables are selected

cross-validation method of Lasso
is called k-fold cross validation
and it measures the mean square error
of prediction by propagating with 1) fold removed
of present variables

Q. Analysis of Variance :-

Q.2.1:- aov() function.

~~(Q.2.1.)~~

* Table 9.5. Formulas for common research designs.

Q.2.2. The order of formula terms.

Q.3. One-way Anova

* aov ($y \sim x_1$).

* plot means () - plot group means and confidence interval.

Q.3.1. Multiple Comparisons

* Tukey HSD () - test of all pairwise differences between groups.

* glht() in multcomp package.

Q.3.2:- Assessing test assumptions:-

* Q-Q Plot - to assess normality assumption.

* bartlett.test () - for homogeneity of variance, equal variance in each group.

Other tests for homoscedasticity [ed]
~~variance~~

- * fligner.test(),
 - * hov() in HH package.
-

q.H. One Way Anova

- * aov()
- * effect(), in effects library.
- * glht() in multcomp library.

q.H.2: Visualizing the results

- * anova(), in HH library.

q.H.1: Assessing test assumption

- * homogeneity of regression slopes.

aov(weight ~ gesttime + dose)

Quantitative

Qualitative

9.5. Two-way factorial ANOVA:-

- * interaction.plot() - to display interaction in two-way ANOVA.

- * plotmeans().

- * interactionawt().

`aov(len ~ supp * dose)`

Qualitative

9.6. Repeated measures ANOVA:-

- * shows ANOVA with one between and within groups factors.

- * interaction.plot().

`aov(uptake ~ conc * Type + Error(Plant / (conc)))`.

- * boxplot().

- * reshape() package

- * sphericity - variances of differences b/w any 2 levels of within group factors are equal.

q.t. Manova.

- * `y <- bind(cbind(calories, fat, sugar))`
- * `manova(lynx ~ diet)`.
- * `summary.aov(lift) - prints univariate results.`
- * `TukeyHSD.`

q.t. 1: Assessing test assumptions:

Two assumption

multivariate normality. (Q-Q' plot)

(1). multivariate covariance

(2). homogeneity of covariance matrices.

`mahalanobis.` } for 1st assumption.

`qqplot()`

`identify()`.

`Box's M test.` - No R function for this.

`agoplot()` in `mvoutlier` package for multivariate outliers.

9.7.2 Robust Manova

- * wilks.test() in mvtnorm package.
- * ~~Non parametric Manova~~.

* Robust Manova, one-way Manova.
wilks.test() in mvtnorm package.

* Nonparametric Manova,
adonis() in vegan package

9.8. Anova as Regression

Table 9.6. Built in Contrasts

- Contr.helmert
- Contr.poly.
- Contr.sum.
- Contr.treatment
- Contr.SAS.

- Contracts ()
- Options ()

◦ Futures, forward

◦ Commodity contracts

◦ spot and forward

◦ spot and forward

() Futures, forward

10. Power Analysis:-

- pwr package.

Table 10.1 pwr package:

- pwr.2p.test()
- pwr.2p2n.test()
- pwr.anova.test()
- pwr.chisq.test()
- pwr.F2.test()
- pwr.p.test()
- pwr.r.test()
- pwr.t.test()
- pwr.t2n.test(),

10.2.1:- t-tests

- `purr.t.test(n= , d= , sig.level= , power= , type= , alternative=)`

10.2.2:- Anova:-

- `purr.anova.test(k= , n= , f= , sig.level= , power=)`

10.2.3. Correlations:-

[purr.r.test]

- `purr.r.test(n= , r= , sig.level= , power= , alternative=)`

10.2.4. Linear Models:-

- `purr.lq.test(lk= , V= , lQ= , sig.level= , power=)`

10.Q.5. Tests of proportions:

- `pwr.prop.test(ha = , n = , sig.level = , power =)`.
- `ES.h()`.

10.Q.6. Chi-square tests:-

- `pwr.chisq.test(w = , N = , df = , sig.level = , power =)`.
- `ES.w2()`.

10.Q.7. Choosing an appropriate [redacted] effect size in novel situations:-

- Listing 10.1. - sample sizes for detecting significant effects in a one-way ANOVA.

11. Intermediate graphs:-

11.1. Scatterplots:-

- `plot()`.
- `lowess()`, `loess()`.
- `scatterplot()` in `car` package.

11.1.1. Scatterplot matrices:-

- `pairs()`.
- `scatterplotMatrix()`.
- There are more functions mentioned.

11.1.2. High density scatterplots:-

11.1.3. 3D Scatterplots:-

11.1.4. Spinning 3D scatter plots:-

11.1.5. Bubbleplots:-

- `symbols()`

11.2. Line charts: plot(), lines().

- type = "b"

Line chart options. Table 11.1.

P - Points only.

L - Lines only.

O - Overplotted points.

b, c - points joined by lines.

s, S - stair steps.

h - Histogram-line vertical lines.

n - Doesn't produce any points.

11.3. Corrgrams:

- corrgram().

◦ When order = TRUE, the variables are reordered using principal component Analysis of the correlation matrix.

- corrgram(x, order = , panel = , text.panel =)

diag.panel =).

- Table 11.2 panel options for `program()` function.
- `ColorRampPalette()`, `col.regions` option.

4. Mosaic plots:-

`mosaic()` in `vcd` library.

Test	Test statistic	Test	Test statistic
Chi-squared test	(GOF) test focuses on overall association	GOF test	(GOF) test focuses on overall association
Fisher's exact test	(Fisher) test focuses on individual cell associations	Fisher's exact test	(Fisher) test focuses on individual cell associations
Exact test	(Exact) test focuses on individual cell associations	Exact test	(Exact) test focuses on individual cell associations
Monte Carlo test	(Monte Carlo) test focuses on overall association	Monte Carlo test	(Monte Carlo) test focuses on overall association
Mid-P test	(Mid-P) test focuses on individual cell associations	Mid-P test	(Mid-P) test focuses on individual cell associations
Pearson's chi-squared test	(Pearson) test focuses on overall association	Pearson's chi-squared test	(Pearson) test focuses on overall association
Phi test	(Phi) test focuses on overall association	Phi test	(Phi) test focuses on overall association
Taylor-Hsiung test	(Taylor-Hsiung) test focuses on overall association	Taylor-Hsiung test	(Taylor-Hsiung) test focuses on overall association

12. Resampling statistics and Bootstrapping

12.1. Permutation tests

- coin package.
- lmPerm package.

12.2. Permutation tests with coin package

- Table 12.2 coin functions providing permutation tests alternatives to traditional tests.
- Oneway-test ($y \sim A$)
- wilcox-test ()
- kruskal-test ()
- chisq-test ()
- emh-test ()
- lbl-test ()
- Spearman-test ()
- Friedman-test ()
- wilcoxonsign-test ()

12.2.1. Independent two sample and k-sample tests

- One-way-test()
- transform()
- Wilcox-test()

12.2.2. Independence in contingency tables

- chisq-test()
- (mh-test(),
- fbl-test()

12.2.3. Independence b/w numeric variables

- Spearman-test()

12.2.4. Dependent two sample and k-sample tests

- Wilcoxonsign-test()

12.3. Permutation tests with
lmpoem package:-

- lmp (lme modified)

- aovp (aov modified)

- perm = Exact, Prob, SPR.Exact.

12.3.1. Simple and polynomial regression,

12.3.2. Multiple regression,

12.3.3. One-way Anova and Ancova:-

12.3.4. Two-way Anova :-

12.5. Bootstrapping:-

- boot package.

- boot () .

- boot.ci () .

12.6. Bootstrapping with ~~single~~ Boot (statistic) package.

- 12.6.1. Bootstrapping a single statistic.
- 12.6.2. Bootstrapping several statistic.

boot

bootstrapping one statistic

* boot command -> boot

* prep function | bootstrap

* options() | sampling

* sample() | mapping

* set.seed() | randomization

* rnorm() | distribution

* pnorm() | peek

* quantile() | distribution

* quantile() | distribution

* quantile() | distribution

13. Generalized Linear models:-

13.1.1. glm() function

- `glm(formula, family=family(link=link, data))`.
- Table 13.1. glm parameters:-

<u>Family</u>	<u>Default link function,</u> (<code>link = "logit"</code>), = " <code>identity</code> ") = " <code>inverse</code> ") = " <code>1/mu^2</code> ") = " <code>log</code> ") = " <code>identity</code> ", variance=constant)
• binomial	
• gaussian	
• gamma	
• inverse, gaussian	
• poisson	
• quasi	
quasibinomial	= " <code>logit</code> ")
quasipoisson	= " <code>log</code> ").

13.1.3. Model fit and regression diagnostics:-

- hatvalues()
- rstudent()
- cooks.distance()
- influencePlot()

13.2 Logistic Regression:-

- To compare 2 nested models, use anova().
- In generalized linear models, you'll want a chi-square version of this test.
- anova(model1, model2, test = "chisq").

13.2.1. Interpreting the model parameters:-

- coef(fit.reduced).
- exp(coef(fit.reduced)).
- exp(confint(fit.reduced)).

13.Q.2 :- Assessing the impact of predictors on the probability of an outcome :-

13.Q.3. Overdispersion :-

- Overdispersion occurs when the observed variance of response variable is larger than what would be expected from a binomial distribution.
- Over dispersion can lead to distorted test standards errors and inaccurate tests of significance.
- $\phi = \frac{\text{Residual deviance}}{\text{Residual df.}}$
- deviance ()

Test for overdispersion.

- you fit the model above, but in first instance you use family = "binomial", in second instance family = "quasibinomial"
- The glm object returned in first case is "fit" and in second case is "fit.od".
- `pchisq`(`summary(fit.od)`)\$dispersion *
- `pchisq`(`summary(fit.od)`)\$df.residual, `fit` & `df.residual`, `fit` & `df.residual`, `lower=F`).
- $H_0: \phi=1$
- $H_1: \phi \neq 0$.

13. Q.H. Extensions:-

1. Robust logistic Regression

- `glmRob()` in `robust` package.
- fitting logistic regression ~~to~~ models to data containing outliers and influential observations.

2. Multinomial logistic Regression,

- If the response variable has more than two unordered categories.
- `mlogit()` in `mlogit` package.

3. Ordinal logistic Regression:-

- response variable is a set of ordered categories.
- `lrm()` in `rms` package.

13.3. Poisson regression:-

- The gee package provides a test for overdispersion in poisson case.
- `gee::overdispersion.test()`, in gee package

13.3.1. Interpreting model parameters.

13.3.2. Overdispersion.

13.3.3. Extensions.

1. Poisson regression with varying time periods.

2. Zero inflated poisson regression,

• `zeroinfl` in pscl package.

3. Robust poisson regression,

• `glmRob()` in robust package.

14. Principal components and factor analysis:-

14.1. Principal components and Factor analysis in R:-

- `princomp()`. } In base R
- `factanal()`.

Table 14.1. Useful factor analytic functions in psych package:-

- `principal()`
- `fa()`,
- `fa.parallel()`,
- `factor.plot()`
- `fa.diagram()`
- `scree()`.

14. 2. Principal components.

14.2.1. Selecting no. of components to extract

- Kaiser-Harris criterion. - retain components with eigen values greater than one.
- Bartlett spheric test - eigen values are plotted against their component numbers.
- run simulations, extracting eigen values from random data matrices of same size, as original matrix. If an eigen value based on real data is $>$ than average corresponding eigen values from a set of random matrices, that component is retained.
- `fa.parallel()`,

H.Q.2. Extracting principal components:-

- principal(r, nfactors=, rotate=, scores=).

H.Q.3. Rotating principal components:-

- varimax.

H.Q.4. Obtaining principal component scores:-

H.3. Explanatory Factor Analysis :-

H.3.1. Deciding How many factors to extract.

- For EFA, the Kaiser-Harris criterion is eigen values greater than 1.

H.3.2. Extracting common factors:-

- fa(r, nfactors=, n.obs=, rotate=, scores=, fm=).

- fm - factoring method (minres default),
◦ "ml" - maximum likelihood,
◦ "pa" - iterated principal axis.
◦ "wls" - weighted least squares.
◦ "gls" - generalized weighted least squares
◦ "minres" - minimum residuals.

IH.3.3. Rotating factors

a varimax - orthogonal

◦ promax - oblique

◦ factor.plot()

◦ fa.diagram()

IH.3.4. Factor slopes

IH.3.5: Other EFA-related packages

M.4. ~~Confirmatory~~ Other latent variable models:

- Confirmatory Factor Analysis.
- Structural equation modelling. - sem, OpenMx, lavaan.
- Ibm - latent models.
- Latent class models - Flexmix, lmm, randomLCA, and ~~pol~~ polCA.
- Lda - latent class discriminant analysis.
- Lsa - latent semantic analysis.
- Ca - simple and (multidimensional) multiple correspondence analysis.
- multidimensional scaling. - cmdscale()
g MDS() in MASS.

15. Time series

Table 15.1. Functions for time series analysis:-

Function	package
• ts()	stats
• start()	stats
• end()	stats
• frequency()	stats
• window()	stats
• mat()	forecast
• stl()	stats
• monthplot()	stats
• seasonplot()	forecast
• HoltWinters()	stats
• forecast()	forecast

Function

- accuracy()
 - ets()
 - lag()
 - Acf(),
◦ Pacf()
 - diff(),
◦ ndiff()
 - adf.test()
 - arima()
 - Box.test()
 - bds.test()
 - auto.arima()
- } package,
} forecast
} stats
} forecast
} base
} forecast
} tseries
} stats
} tseries
} forecast

- `ts(data, start= , end= , frequency)`
- `start(tsales)`
- `end(tsales)`
- `frequency(tsales)`.

◦ `tsales.subset <- window(tsales,`

start= , end=).

15.2. Smoothing and seasonal decomposition
with simple moving average

◦ centred moving average.

◦ `sma()` in TTR.

◦ `rollmean()` in ZOO.

◦ `mal()` in forecast.

15.2.2. Seasonal decomposition:-

◦ trend component.

◦ seasonal component.

- irregular (error) component.

- $\text{stl}(ts, s.window = \text{?}, t.window = \text{?})$.

ts - time series to be decomposed.

$s.window$ - controls how fast the seasonal effects change over time.

$t.window$ - how fast the trend can change over time.

- $s.window = \text{"periodic"}$ - forces seasonal effects to be identical over years.

- $\text{stl}()$ can only handle additive models.
But we can handle multiplicative models using log.

- $\text{monthplot}()$.

- $\text{seasonplot}()$ in `forecast`.

15.3. Exponential forecasting models?

• HoltWinter()
ets() in forecast.

ets(ts, model = "IZZ").

Table 15.3. Functions for fitting simple, double, and triple exponential forecasting models:

Type.	parameters fit	functions.
simple	level	ets(asmodel = "ANN") ses(ts).
double	level, slope	ets(ts, model = "ADD") holt(ts).
triple	level, slope, seasonal	ets(ts, model = "AAA") hw(ts).

15.3.1. Simple exponential smoothing:-

- forecast() is used to predict time series k steps into the future.
- $\text{forecast}(f_t, k)$.
- $\text{accuracy}(f_t)$.

15.3.2. Holt and Holt-Winters Exponential

- Model equation:

$$L_t = \alpha Y_t + (1 - \alpha) L_{t-1}$$

- Y_t is actual observation at time t .
- α is smoothing parameter.
- L_t is previous level (smoothed value) or average.

15.3.2. Holt and Holt-Winters exponential smoothing:-

Holt exponential smoothing model equations,

1. Level equation.

$$L_t = \alpha Y_t + (1-\alpha)(L_{t-1} + T_{t-1})$$

2. Trend eq:

$$T_t = \beta(L_t - L_{t-1}) + (1-\beta)T_{t-1}$$

3. Forecast eq:

$$\hat{Y}_{t+k} = L_t + T_t \times k$$

\hat{Y}_{t+k} is forecast for k periods ahead.

α : Controls smoothing of level

β : Controls smoothing of trend.

Holt-Winters exponential smoothing:-

• Model equations:

Additive seasonality.

• Level eq:

$$L_t = \alpha (Y_t - S_{t-s}) + (1-\alpha) (L_{t-1} + T_{t-1})$$

• Trend eq:

$$T_t = \beta (L_t - L_{t-1}) + (1-\beta) T_{t-1}$$

• Seasonality eq:

$$S_t = \gamma (Y_t - L_t) + (1-\gamma) S_{t-s}$$

• Forecast eq:

$$\hat{Y}_{t+k} = L_t + T_t * k + S_{t+k-s}$$

15.3. The ets() function and automated forecasting:

• ets(AirPassengers, model = "MAM")

• hw(AirPassengers, seasonal = "multiplicative")

• et(JohnsonJohnson) → automatic select best fitting model.

15.4. ARIMA forecasting models:-

15.4.1. Prerequisite concepts:-

• $\log(\text{ts}, k)$, k = no. of logs.

• Autocorrelation function plot with

(1) acf() in stats.

(2) Acf() in forecast

• partial autocorrelation plot with,

(1) pacf() in stats.

(2) Pacf() in forecast.

- ARIMA models are designed to fit stationary time series.
- In stationary ts, the statistical properties don't change over time.
- The mean and variance are constant.
- The autocorrelations of any lag k don't change over time.
- differencing, $y_{t-1} - y_t$.
 - differencing once, removes linear trend
 - differencing twice, removes quadratic trend
 - differencing thrice removes cubic trend
- diff(ts, differences=d).
- ndiff() in forecast \rightarrow can help to determine best value of d.
- Daugmented Dickey-Fuller (ADF) test \rightarrow to evaluate stationarity assumption.

→ `adftest()` in tseries

In summary,

- (1) ACF and PACF plots to determine parameters of ARMA.
- (2). Transformation and differencing used to help achieve stationarity.

I.S.H.Q. ARMA and ARIMA models:-

ARMA(p,q)

$$Y_t = \alpha + B_1 Y_{t-1} + B_2 Y_{t-2} + \dots + B_p Y_{t-p} - \Theta_1 \epsilon_{t-1} - \Theta_2 \epsilon_{t-2} - \dots - \Theta_q \epsilon_{t-q} + \epsilon_t$$

From past ~~predictions~~ p values and q residuals

- ACF plot used to identify q parameter
- PACF plot for p parameter.

Evaluating model fit :-

- qqnorm(), qqline().
- Box.test(fit\$residuals, type = "Ljung-Box")
- If model is appropriate, residuals should be normally and independently distributed. (no relationship between them) or (no autocorrelation).
- Box.test() provides a test for autocorrelation.

15.4.3. Automated ARIMA forecasting :-

- auto.arima() in forecast.