ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΣΧΟΛΗ ΓΕΩΠΟΝΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΦΥΤΙΚΗΣ ΠΑΡΑΓΩΓΗΣ ΚΑΙ ΑΓΡΟΤΙΚΟΥ ΠΕΡΙΒΑΛΛΟΝΤΟΣ
ΕΡΓΑΣΤΗΡΙΟ ΒΙΟΜΕΤΡΙΑΣ

Πτυχιακή Εργασία με Θέμα:

Ανάπτυξη κώδικα, χρήση βιβλιοθηκών και εφαρμογή τους στην ανάλυση πειραματικών σχεδίων στο λογισμικό στατιστικής ανάλυσης Jamovi και τις γλώσσες προγραμματισμού R, Python.

Φοιτητής: ΤΣΑΝΟΣ ΝΙΚΟΛΑΟΣ

Βόλος, 2022

Πτυχιακή Διατριβή:

" Ανάπτυξη κώδικα, χρήση βιβλιοθηκών και εφαρμογή τους στην ανάλυση πειραματικών σχεδίων στο λογισμικό στατιστικής ανάλυσης Jamovi και τις γλώσσες προγραμματισμού R, Python."

"Code implementation, use of libraries and their application in the analysis of experimental designs using the statistical software Jamovi and programming languages R and Python"

Η τριμελής συμβουλευτική επιτροπή αποτελείται από τους:

1. Νάκα Χρήστο, Καθηγητή (Βιομετρία, Πανεπιστήμιο Θεσσαλίας) (Επιβλέπων)
2. Κυπαρίσση-Σαπουντζάκη Άρη, Αναπληρωτή Καθηγητή (Οικοφυσιολογία Φυτών)
3. Βαγγέλα Ιωάννη, Επίκουρο Καθηγητή (Φυτοπαθολογία)

«Βεβαιώνω ότι είμαι συγγραφέας αυτής της πτυχιακής εργασίας, η οποία εκπονήθηκε σύμφωνα με τον Κανονισμό Εκπόνησης Πτυχιακής Εργασίαςτου ΤΓΦΠΑΠ»

Ευχαριστίες

Με την περάτωση της παρούσας πτυχιακής εργασίας θα ήθελα να ευχαριστήσω τον επιβλέπων καθηγητή, κύριο Χρήστο Νάκα για την εμπιστοσύνη που μου έδειξε εξ' αρχής, αναθέτοντάς μου το συγκεκριμένο θέμα καθώς και για την καθοδήγηση που μου προσέφερε. Ακόμα θα ήθελα να ευχαριστήσω την κυρία Έφη Μπατάκα για τις εποικοδομητικές τις υποδείξεις και τον χρόνο που διέθεσε για την επιμέλεια της εργασίας.

Πίνακας Περιεχομένων

Εισαγωγή

Η σύγχρονη γεωργική έρευνα βασίζεται σε μεγάλο βαθμό στον γεωργικό πειραματισμό. Ο γεωργικός πειραματισμός πραγματεύεται γεωργικά πειράματα που έχουν στόχο τον έλεγχο της επίδρασης διαφορετικών «Επεμβάσεων» σε κάποιο χαρακτηριστικό των φυτών. Κάποια από αυτά τα χαρακτηριστικά μπορεί να είναι η παραγωγή των φυτών, το βάρος του καρπού, η συγκέντρωση σε σάκχαρα κ.α. Οι επεμβάσεις μπορούν να αποτελούν το διαφορετικό βαθμό λιπάσματος, το βαθμό άρδευσης, το υπόστρωμα ή ακόμα και τις διαφορετικές ποικιλίες ενός φυτού, αναλόγως το υπόβαθρο του πειράματος.

Σκοπός της συγκεκριμένης εργασίας είναι η εξέταση διαφορετικών πειραματικών σχεδιασμών και η στατιστική ανάλυση των δεδομένων που προκύπτουν από κάθε σχεδιασμό, μέσω του στατιστικού λογισμικού Jamovi αλλά και με την χρήση δύο γλωσσών προγραμματισμού, της R και της Python.

Το Jamovi είναι ένα δωρεάν, ανοιχτού κώδικα στατιστικό υπολογιστικό λογισμικό, σχεδιασμένο ώστε να είναι όσο το δυνατόν πιο απλό στη χρήση, ενώ παράλληλα προσφέρει προηγμένες αναλύσεις όπως μικτά μοντέλα και παραγοντική ανάλυση. Αποτελεί μια αξιόπιστη δωρεάν εναλλακτική του SPSS ενώ τα εξωτερικά πακέτα που χρησιμοποιεί είναι βασισμένα στην R.

Η R και η Python είναι και οι δύο γλώσσες προγραμματισμού ανοιχτού κώδικα. Η R χρησιμοποιείται κυρίως για στατιστική ανάλυση, ενώ η Python παρέχει μια γενικότερη προσέγγιση στην επιστήμη των δεδομένων.

Συγκεκριμένα η R έχει αναπτυχθεί τις τελευταίες δύο δεκαετίες από ακαδημαϊκούς και στατιστικούς και πλέον διαθέτει ένα από τα πλουσιότερα συστήματα για την εκτέλεση ανάλυσης δεδομένων, καθώς παρέχει βιβλιοθήκες (ή πακέτα) για οποιαδήποτε ανάλυση.

Η Python, από την άλλη, είναι μια γλώσσα προγραμματισμού γενικής χρήσης με πιο εύχρηστη αναπαραγωγή και προσβασιμότητα μιας και είναι συμβατή με περισσότερες εφαρμογές. Για παράδειγμα εάν είναι αναγκαία η χρήση των αποτελεσμάτων της ανάλυσής σε μια εφαρμογή ή ιστότοπο, η Python είναι η καλύτερη επιλογή.

Στο πρώτο κεφάλαιο της παρούσας εργασίας αναλύεται ο τρόπος με τον οποίο τα δεδομένα ενός πειράματος μπορούν να εισαχθούν σωστά στο πρόγραμμα Jamovi και στις δύο γλώσσες προγραμματισμού έτσι ώστε να είναι δυνατή η στατιστική τους ανάλυση.

Το δεύτερο κεφάλαιο πραγματεύεται τον τρόπο με τον οποίο μπορεί να γίνει έλεγχος των προϋποθέσεων για την κατασκευή γραμμικού μοντέλου ενώ το τρίτο κεφάλαιο αναλύει την εφαρμογή του T-Test για να προσδιοριστεί εάν μια θεραπεία δύο επιπέδων έχει πράγματι επίδραση στον πληθυσμό που εξετάζεται. Σε καθένα από τα υπόλοιπα κεφάλαια αναλύεται ένας διαφορετικός πειραματικός σχεδιασμός και παρέχεται καθοδήγηση για τη στατιστική ανάλυση των αποτελεσμάτων του κάθε σχεδιασμού. Οι εκδόσεις των Jamovi, R και Python καθώς και όλων των πακέτων βάση των οποίων αναπτύχθηκαν τα παραδείγματα των κεφαλαίων βρίσκονται στο παράρτημα της σελίδας 139.

1.Εισαγωγή δεδομένων

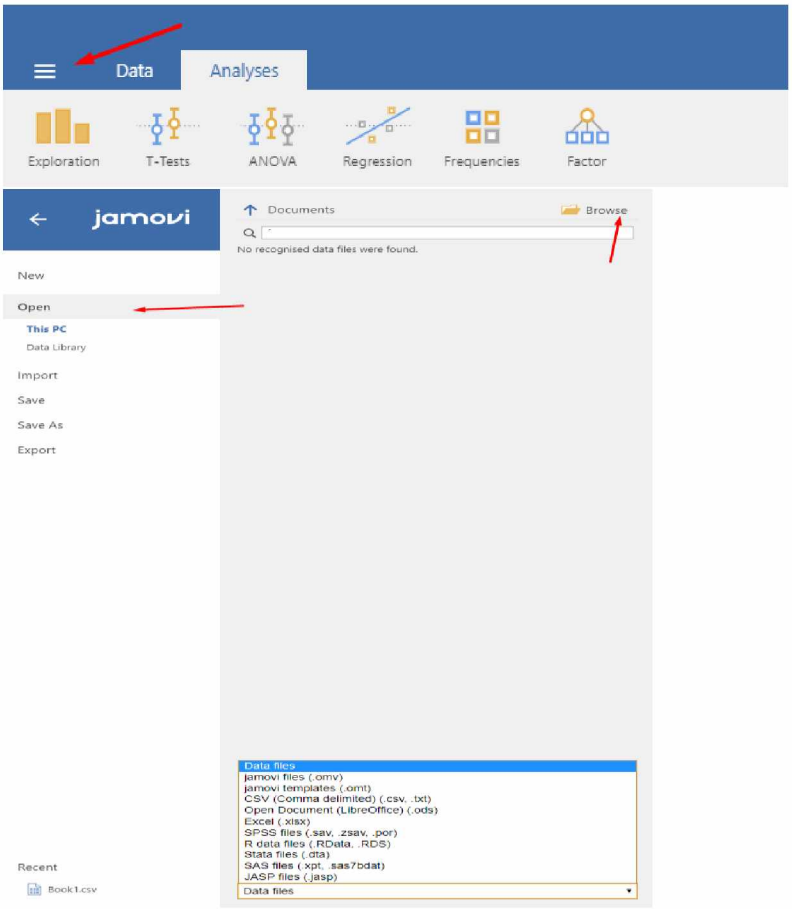
Συχνά τα δεδομένα ενός πειράματος είναι αποθηκευμένα σε ένα αρχείο excel. Για την στατιστική ανάλυση αυτών των δεδομένων προηγείται η εισαγωγή τους σε ένα πρόγραμμα στατιστικής ανάλυσης. Παρακάτω περιγράφεται ο τρόπος με τον οποίο μπορεί να γίνει η εισαγωγή ενός αρχείου Excel στο στατιστικό λογισμικό Jamovi, στη γλώσσα προγραμματισμού R και στην γλώσσα Python. Ως παράδειγμα θα χρησιμοποιηθεί το αρχείο csv της εικόνας 1.1

	A	B	C
1	Price	Field	
2	100	c1	
3	88.7	c1	
4	95.3	c1	
5	101.11	c1	
6	75.6	c1	
7	83.2	c1	
8	98.8	c1	
9	63.6	c1	
10	81.1	c1	
11	67.2	c1	
12	76.6	c2	
13	99.99	c2	
14	77.88	c2	
15	88.77	c2	
16	66.5	c2	
17	104.7	c2	
18	91.1	c2	
19	89.04	c2	
20	59.9	c2	
21	69.8	c2	

(Εικόνα 1.1)Αρχείο csv προς εισαγωγή.

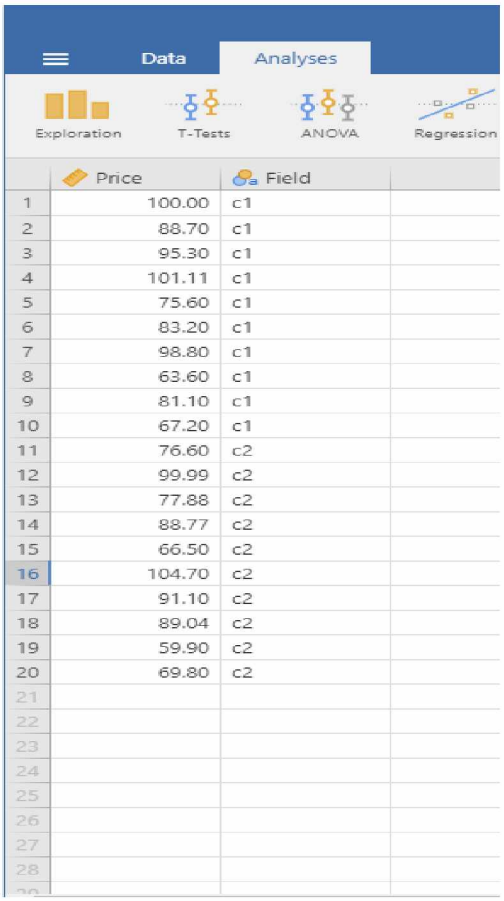
1.1 Εισαγωγή δεδομένων σε Jamovi

Η εισαγωγή δεδομένων σε Jamovi είναι μια αρκετά απλή διαδικασία.
Πατώντας τις 3 γραμμές στο toolbar, πάνω από το μενού Exploration, δίνεται η επιλογή για άνοιγμα αρχείου.



(Εικόνα 1.2)Άνοιγμα αρχείου σε jamovi

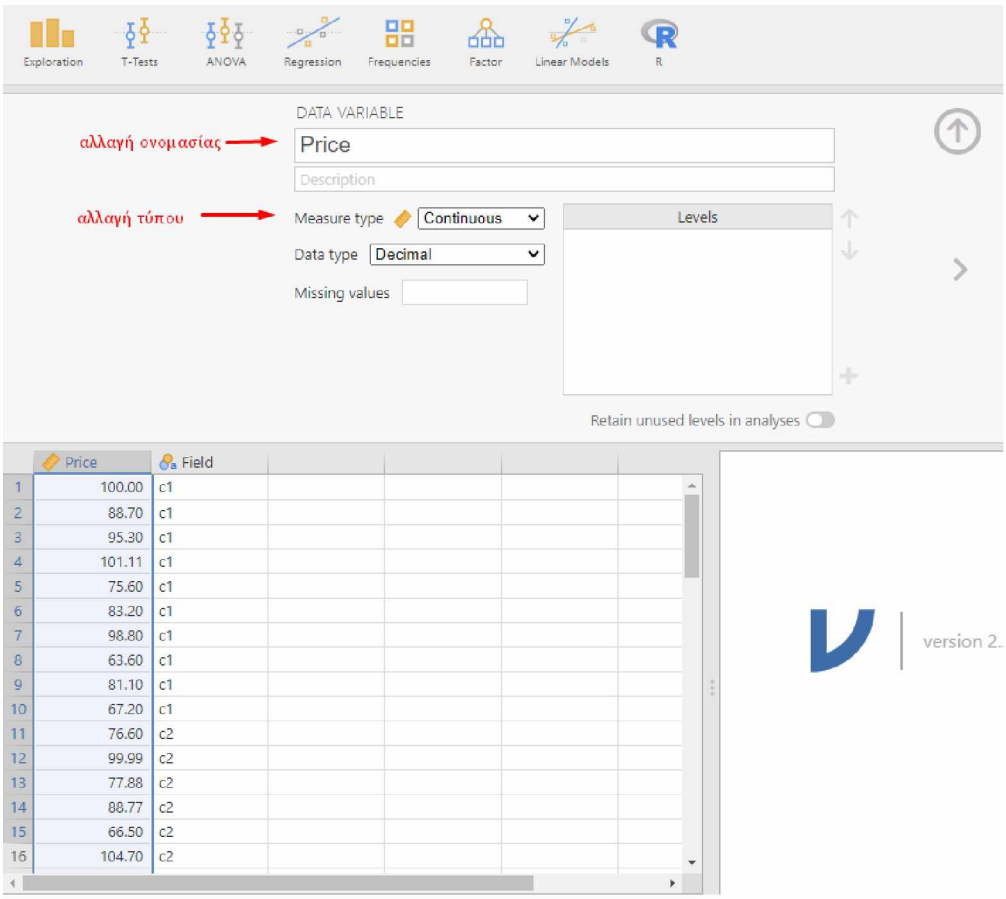
Αφού γίνει η επιλογή του ανάλογου αρχείου τα δεδομένα εμφανίζονται όπως στην εικόνα 1.3.



	Price	Field
1	100.00	c1
2	88.70	c1
3	95.30	c1
4	101.11	c1
5	75.60	c1
6	83.20	c1
7	98.80	c1
8	63.60	c1
9	81.10	c1
10	67.20	c1
11	76.60	c2
12	99.99	c2
13	77.88	c2
14	88.77	c2
15	66.50	c2
16	104.70	c2
17	91.10	c2
18	89.04	c2
19	59.90	c2
20	69.80	c2
21		
22		
23		
24		
25		
26		
27		
28		

(Εικόνα 1.3)Πλαίσιο δεδομένων σε Jamovi

Στην περίπτωση που οι μεταβλητές έχουν ελληνική ονομασία, το Jamovi μπορεί να παρουσιάσει σφάλματα κατά την επεξεργασία των δεδομένων. Για την αποφυγή τέτοιων σφαλμάτων συνιστάται η μετονομασία των μεταβλητών χρησιμοποιώντας μόνο λατινικούς χαρακτήρες.
Η αλλαγή ονόματος μιας μεταβλητής μπορεί να γίνει με διπλό κλικ πάνω σε μία από τις 2 στήλες. Είναι ακόμα δυνατή η αλλαγή του τύπου των μεταβλητών (Norminal, Ordinal, Continuous).



(Εικόνα 1.4)Αλλαγή τύπου μεταβλητών

1.2 Εισαγωγή δεδομένων σε R

Εάν και η εφαρμογή δεδομένων χειροκίνητα είναι πάντα μια επιλογή R επιτρέπει την αυτόματη εισαγωγή δεδομένων από ένα αρχείο .txt ή .csv

Για αρχεία .txt:
my_data <- read.delim(file.choose())

Για αρχεία .csv:
my_data <- read.csv(file.choose())

Με αυτόν τον τρόπο εισάγονται τα δεδομένα του κάθε αρχείου στην μεταβλητή "my_data" η οποία αποτελεί μια λίστα τιμών του εισακτέου αρχείου.

Για παράδειγμα η εισαγωγή του αρχείου csv της εικόνας 1.1 μπορεί να γίνει με την εντολή
my_data <- read.csv(file.choose())

Η δομή δεδομένων που παράγεται μετά την εκτέλεση της εντολής είναι μία λίστα.

Στην συνέχεια με την εντολή my_data μπορεί να γίνει έλεγχος για το εάν τα στοιχεία εισήχθησαν σωστά.

```
> my_data <- read.csv(file.choose())
> my_data
    Price Field
1  100.00    c1
2   88.70    c1
3   95.30    c1
4  101.11    c1
5   75.60    c1
6   83.20    c1
7   98.80    c1
8   63.60    c1
9   81.10    c1
10  67.20    c1
11  76.60    c2
12  99.99    c2
13  77.88    c2
14  88.77    c2
15  66.50    c2
16 104.70    c2
17  91.10    c2
18  89.04    c2
19  59.90    c2
20  69.80    c2
>
```

(Εικόνα 1.5)Πλαίσιο δεδομένων σε R.

Μετά την εντολή attach(my_data) είναι δυνατή η ανάκληση της κάθε στήλης του πίνακα(Price,Field) μόνο με το όνομά της χωρίς να καλούμε κάθε φορά το πλαίσιο δεδομένων my_data.

```
> my_data <- read.csv(file.choose())
> attach(my_data)
> Field
 [1] c1 c1 c1 c1 c1 c1 c1 c1 c1 c1 c2 c2 c2 c2 c2 c2 c2 c2 c2 c2
Levels: c1 c2
> Price
 [1] 100.00  88.70  95.30 101.11  75.60  83.20  98.80  63.60  81.10  67.20  76.60  99.99  77.88  88.77  66.50 104.70  91.10  89.04
[19]  59.90  69.80
>
```

(Εικόνα 1.6)Μετατροπή στήλης σε μεταβλητή.

Σύνοψη κώδικα

my_data <- read.csv(file.choose())
attach(my_data)

1.3 Εισαγωγή δεδομένων σε Python

Για την εισαγωγή δεδομένων από ένα αρχείο csv χρησιμοποιείται η βιβλιοθήκη pandas η οποία μπορεί να εγκατασταθεί με την εντολή.

```
pip install pandas
```

Με την εντολή

```
import pandas
data = pandas.read_csv('filename.csv')
```

αποθηκεύονται τα περιεχόμενα του cvs αρχείου στην μεταβλητή data.
Πρέπει να σημειωθεί πως το cvs αρχείο θα πρέπει να βρίσκεται στο working directory της Python.
Για το αρχείο της εικόνας 1.1 θα πρέπει οι τιμές των χωραφιών c1 και c2 να ταξινομηθούν σε υποσύνολα.

Αυτό επιτυγχάνεται με τις παρακάτω εντολές

```
Field1 = data.query('Field == "c1"')['Price']
Field2 = data.query('Field == "c2"')['Price']
```

Με τον τρόπο αυτό οι μεταβλητές Field1 και Field2 αποτελούν λίστες με τις τιμές του c1 και c2 αντίστοιχα.
Με την εντολή

```
print(data.groupby('Field').describe())
```

Γίνεται ο υπολογισμός των συνοπτικών στατιστικών στοιχείων

	Price							
	count	mean	std	min	25%	50%	75%	max
Field								
c1	10.0	85.461	13.639147	63.6	76.975	85.950	97.925	101.11
c2	10.0	82.428	14.677788	59.9	71.500	83.325	90.585	104.70

(Εικόνα 1.7)Συνοπτικά στατιστικά στοιχεία.

Σύνοψη κώδικα

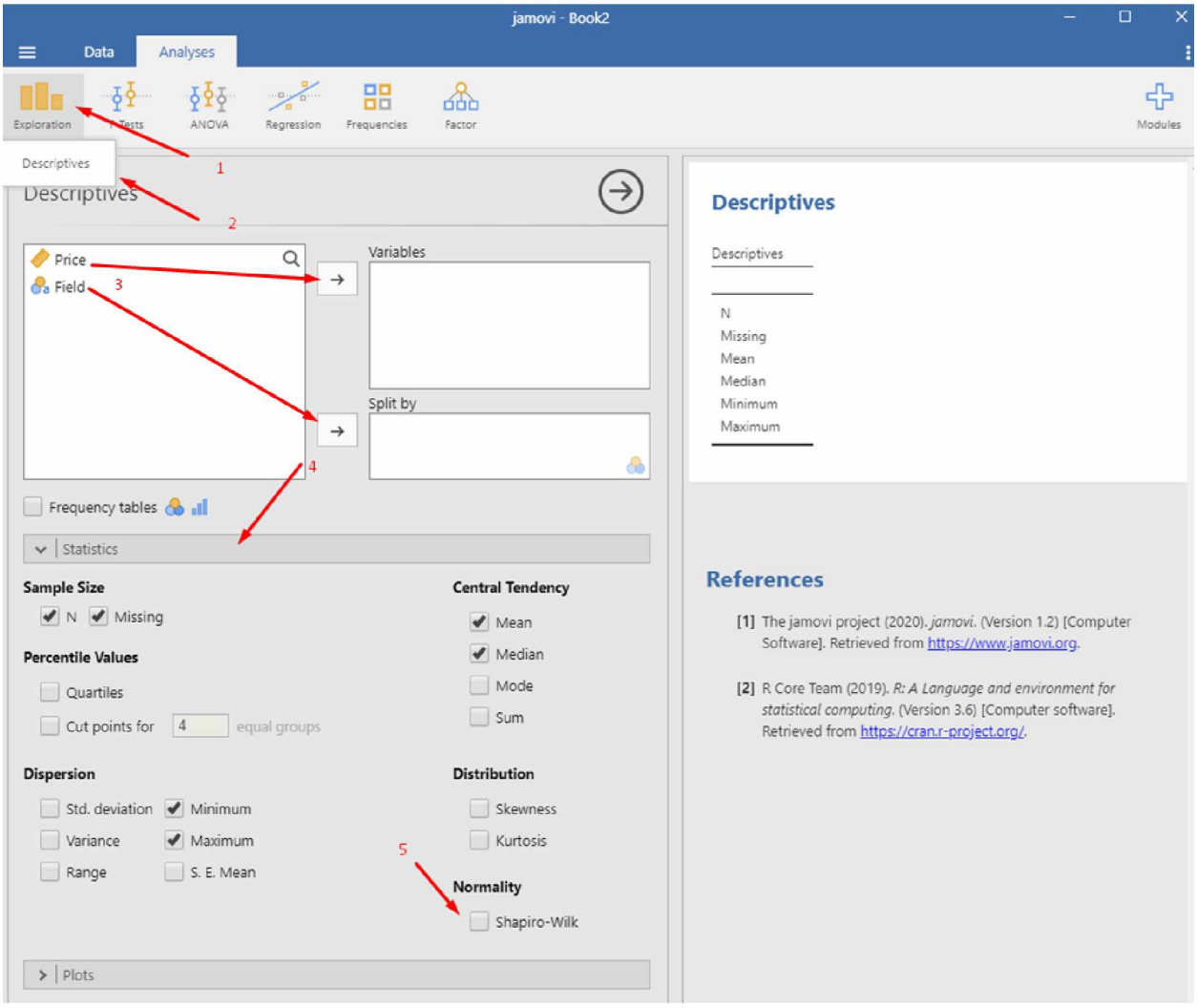
```
pip install pandas
import pandas
data = pandas.read_csv('filename.csv')
Field1 = data.query('Field == "c1"')['Price']
Field2 = data.query('Field == "c2"')['Price']
print(data.groupby('Field').describe())
```

2 Normality test

2.1 Jamovi: (Shapiro-Wilk test)

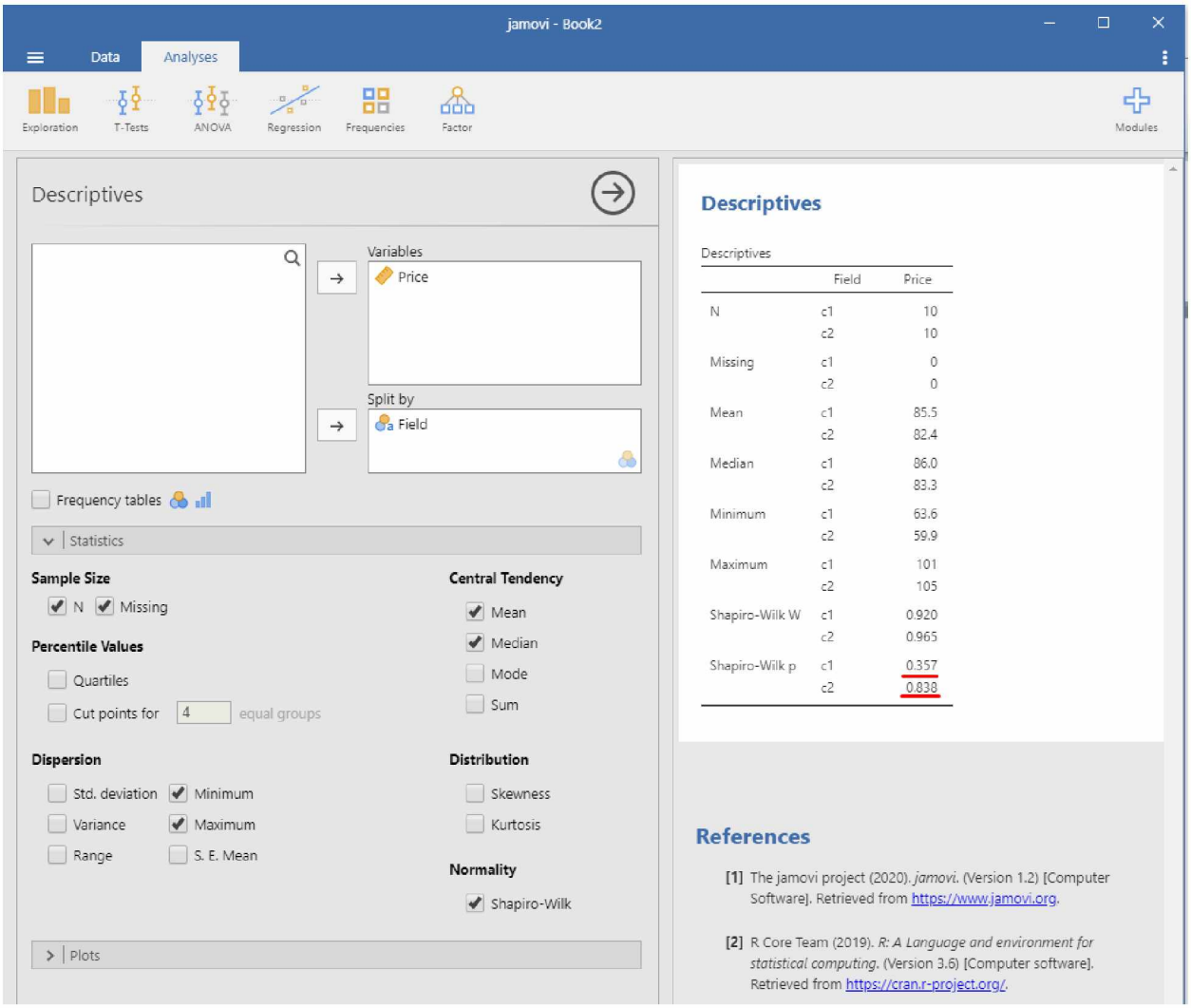
Μετά την σωστή εισαγωγή των δεδομένων στο Jamovi μπορεί να γίνει η διεξαγωγή Normality test από την καρτέλα "Analyses" επιλέγοντας το "Exploration" και στην συνέχεια "Descriptives".

Αφού γίνει επιλογή των μεταβλητών στις οποίες θα γίνει το wilk test, από τα statistics και επιλέγεται το Shapiro-Wilk.



(Εικόνα 2.1)Διεξαγωγή Shapiro-Wilk Test σε Jamovi

Στα δεξιά εμφανίζεται ένας πίνακας στον οποίο μεταξύ άλλων φαίνεται και το p-value. Εάν το p-value είναι μεγαλύτερο από 0.05 τα δεδομένα δεν διαφέρουν σημαντικά από την κανονική κατανομή. Με άλλα λόγια, μπορούμε να υποθέσουμε την κανονικότητα[1].



(Εικόνα 2.2) Αποτελέσματα Shapiro-Wilk test σε Jamovi.

2.2 R: (Shapiro-Wilk test)

Αφού έχει γίνει η εισαγωγή δεδομένων όπως φαίνεται στο κεφάλαιο 1.2, μπορεί να πραγματοποιηθεί το normality test με την παρακάτω εντολή .

shapiro.test(my_data$μεταβλητή απόκρισης)

Για το παράδειγμα της εικόνας 1.1 αυτό αντιστοιχεί σε:

shapiro.test(my_data$Price)

Από το αποτέλεσμα , εάν το p-value είναι μεγαλύτερο από (ή ίσο με) 0.05 τα δεδομένα δεν διαφέρουν σημαντικά από την κανονική κατανομή. Με άλλα λόγια, μπορούμε να υποθέσουμε την κανονικότητα[1].

```
> shapiro.test(my_data$Price)

        Shapiro-Wilk normality test

data:  my_data$Price
W = 0.94684, p-value = 0.3216
```

(Εικόνα 2.3) Αποτελέσματα Shapiro-Wilk test σε R.

Σύνοψη κώδικα

```
my_data <- read.csv(file.choose())
attach(my_data)
shapiro.test(my_data$Price)
```

2.3 Python: (Shapiro-Wilk test)

Η εξαγωγή του shapiro-wilk test πραγματοποιείται με την χρήση της βιβλιοθήκης scipy

```
pip install scipy
```

Η βιβλιοθήκη scipy παρέχει την εφαρμογή scipy.stats.shapiro(x) όπου x είναι μια μεταβλητή που περιέχει λίστα δεδομένων όπως είναι το Field1 και Field2 από το παραπάνω παράδειγμα.Έτσι μέσω των εντολών:

```
import scipy
from scipy import stats
print(stats.shapiro(Field1))
print(stats.shapiro(Field2))
```

Εμφανίζονται οι τιμές για το Field1 και Field2.

Field1: ShapiroResult(statistic=0.9200298190116882, pvalue=0.3572040796279907)

Field2:ShapiroResult(statistic=0.9647067189216614, pvalue=0.8379066586494446)

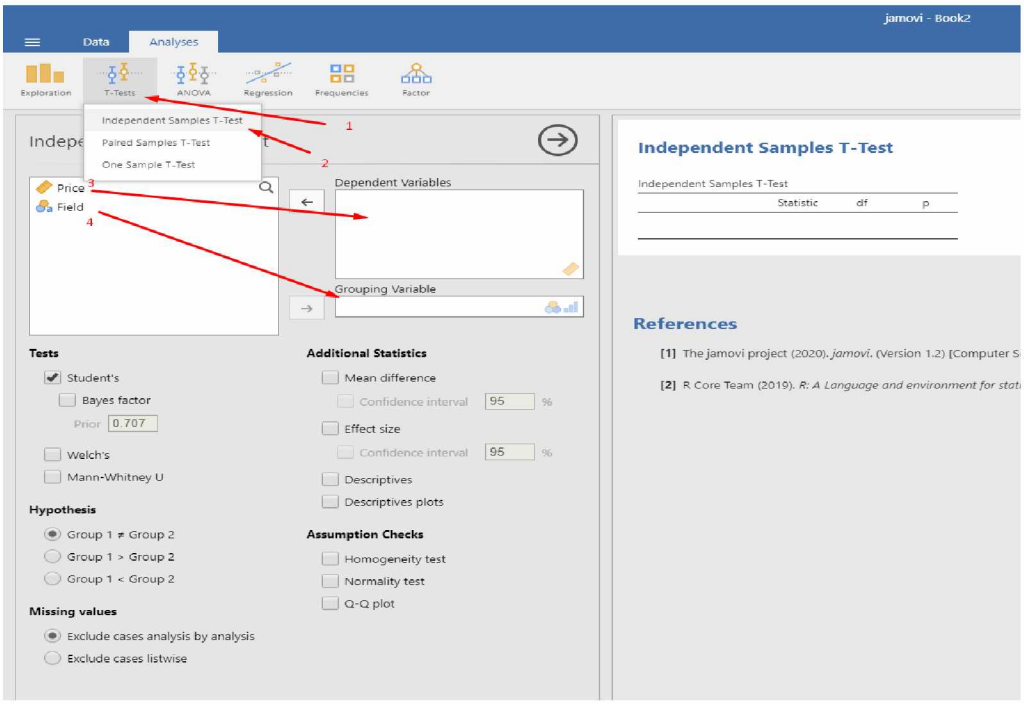
Σύνοψη κώδικα

```
pip install pandas
pip install scipy
import scipy
from scipy import stats
import pandas
data = pandas.read_csv('filename.csv')
Field1 = data.query('Field == "c1"')['Price']
Field2 = data.query('Field == "c2"')['Price']
print(stats.shapiro(Field1))
print(stats.shapiro(Field2))
```

3. T-Test

3.1 T-Test Jamovi:

Η πραγματοποίηση t-test μετά την εισαγωγή των δεδομένων μπορεί να γίνει από την καρτέλα analyses που βρίσκεται στο πάνω μέρος του παραθύρου. Αφού γίνει η επιλογή του τύπου t-test αναλόγως θα πρέπει να επιλεχθούν οι μεταβλητές απόκρισης καθώς και οι μεταβλητή με την οποία γίνεται ο διαχωρισμός τους.



(Εικόνα 3.1)Διεξαγωγή T-Test σε Jamovi.

Στα δεξιά θα εμφανιστεί ο ανάλογος πίνακας με το t-statistic και το p-value.

Independent Samples T-Test

Independent Samples T-Test

		Statistic	df	p
Price	Student's t	0.479	18.0	0.638

(Εικόνα 3.2) Αποτελέσματα T-test σε Jamovi.

3.2 T-Test R:

Αφού η εισαγωγή δεδομένων έχει γίνει σωστά το t-test μπορεί να πραγματοποιηθεί με τους εξής τρόπους.

Για 2 ανεξάρτητα δείγματα:

t.test(y~x)

όπου y είναι αριθμητική μεταβλητή και χ είναι δυαδική μεταβλητή

ή

t.test(y1,y2)

όπου y1 και y2 είναι αριθμητικές μεταβλητές

Για 2 εξαρτημένα δείγματα:

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

όπου y1 και y2 είναι αριθμητικές μεταβλητές

Για one sample t-test:

t.test(y,mu=3)

Με Ho: mu=3

Για παράδειγμα της εικόνας 1.1 το t-test μπορεί να γίνει ως εξής:

t.test(Price~Field)

```
> t.test(Price~Field)

        Welch Two Sample t-test

data:  Price by Field
t = 0.47868, df = 17.904, p-value = 0.638
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -10.28381  16.34981
sample estimates:
mean in group c1 mean in group c2
          85.461           82.428
```

(Εικόνα 3.3)Διεξαγωγή T-Test σε R.

Σύνοψη κώδικα

```
my_data <- read.csv(file.choose())
attach(my_data)
t.test(Price~Field)
```

3.3 T-Test Python:

Μετά την σωστή εισαγωγή των δεδομένων και την εξαγωγή του shapiro-wilk test το t-test μπορεί να πραγματοποιηθεί ξανά με την χρήση της βιβλιοθήκης scipy

Αφού γίνει η εισαγωγή της βιβλιοθήκης με την εντολή

```
import scipy
```

Το t-test γίνεται ως εξής:

Για 2 ανεξάρτητα δείγματα:

```
res = scipy.stats.ttest_ind(x, y,  equal_var=True)
print(res)
```

Για 2 εξαρτημένα δείγματα:

```
res = scipy.stats.ttest_rel(x, y)
print(res)
```

Για one sample t-test:

```
res = scipy.stats.ttest_1samp(a,mean)
print(res)
```

Όπου a = μια λίστα τιμών και mean = ο αναμενόμενος μέσος όρος των τιμών.
Για τον πίνακα τιμών που δόθηκε ως παράδειγμα προηγουμένως το t-test δύο ανεξάρτητων δειγμάτων όπως είναι το c1 και c2 μπορεί να γίνει ως εξής:

```
res = scipy.stats.ttest_ind(Field1, Field2, equal_var=True)
print(res)
```

Με αποτέλεσμα :
Ttest_indResult(statistic=0.47868434429112366, pvalue=0.637926257873664)

Σύνοψη κώδικα

```
import scipy
from scipy import stats
import pandas
data = pandas.read_csv('filename.csv')
Field1 = data.query('Field == "c1"')['Price']
Field2 = data.query('Field == "c2"')['Price']
res = scipy.stats.ttest_ind(Field1, Field2, equal_var=True)
print(res)
```

4. Completely randomized design (CRD)

Το πλήρως τυχαιοποιημένο σχέδιο (CRD) είναι το πιο απλό πειραματικό σχέδιο για στατιστική ανάλυση[2]. Προορίζεται για τη μελέτη των επιπτώσεων ενός παράγοντα ενδιαφέροντος ενώ οι εξωτερικοί παράγοντες ελέγχονται με τέτοιο τρόπο έτσι ώστε να προκαλούν ισάξιες μεταβολές στο αποτέλεσμα του πειράματος μέσω της τυχαιοποίησης[3].Με αυτόν τον τρόπο οποιαδήποτε σημαντική διαφορά στο αποτέλεσμα του πειράματος οφείλεται στον υπό εξέταση παράγοντα.

Παρακάτω θα αναλυθεί πως μπορεί να γίνει ο υπολογισμός του πίνακα Anova μέσω ενός αρχείου excel χρησιμοποιώντας το στατιστικό λογισμικό Jamovi, την γλώσσα προγραμματισμού R αλλά και την γλώσσα Python.

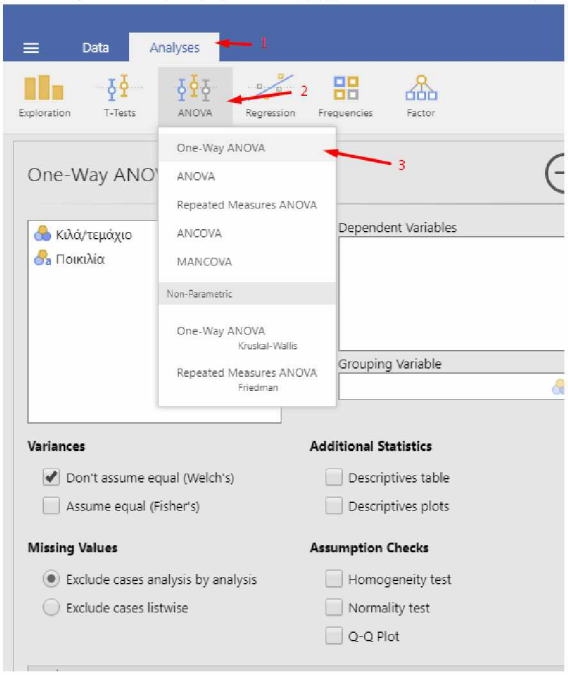
Ως παράδειγμα θα ληφθεί ένα πείραμα CRD κατά το οποίο συγκρίνονται οι αποδόσεις τεσσάρων ποικιλιών σόγιας σε κιλά ανά τεμάχιο:

Κιλά/τεμάχιο	Ποικιλία
30	ποικιλία1
26	ποικιλία1
27	ποικιλία1
28	ποικιλία1
15	ποικιλία2
17	ποικιλία2
21	ποικιλία2
19	ποικιλία2
18	ποικιλία3
15	ποικιλία3
20	ποικιλία3
14	ποικιλία3
13	ποικιλία4
11	ποικιλία4
13	ποικιλία4
12	ποικιλία4

(Εικόνα 4.1) Πλαίσιο δεδομένων πειράματος CRD.

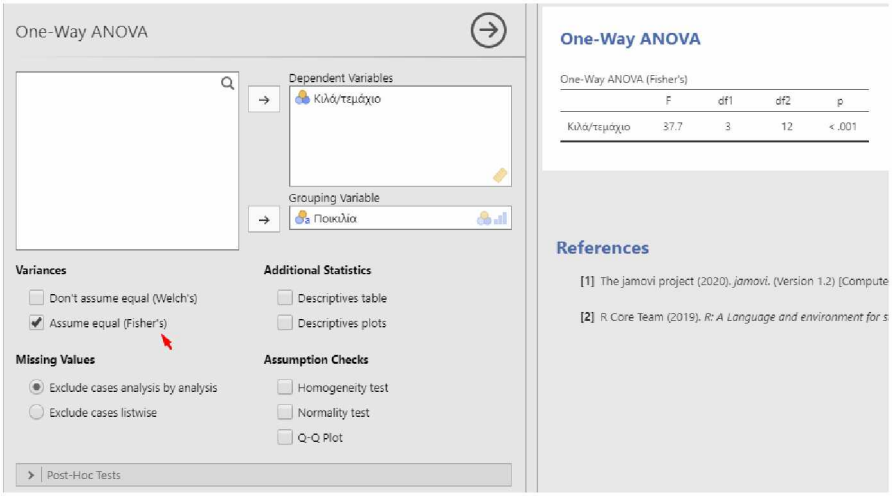
4.1 CRD Jamovi:

Με την εισαγωγή του αρχείου Excel στο jamovi μπορεί να εκτελεστεί το One-way Anova test.



(Εικόνα 4.2)Επιλογή One-Way Anova.

Ως dependent variable τίθεται η μεταβλητή απόκρισης δηλαδή τα Κιλά/τεμάχιο ενώ έως grouping variable η κατηγορική μεταβλητή , δηλαδή η Ποικιλία. Για την απόδειξη της επίδρασης ενός πρωταρχικού παράγοντα σε ένα πλήρως τυχαιοποιημένο σχέδιο(CRD) η αρχική υπόθεση είναι ότι όλες οι ποικιλίες έχουν την ίδια απόδοση σε κιλά/τεμάχιο.

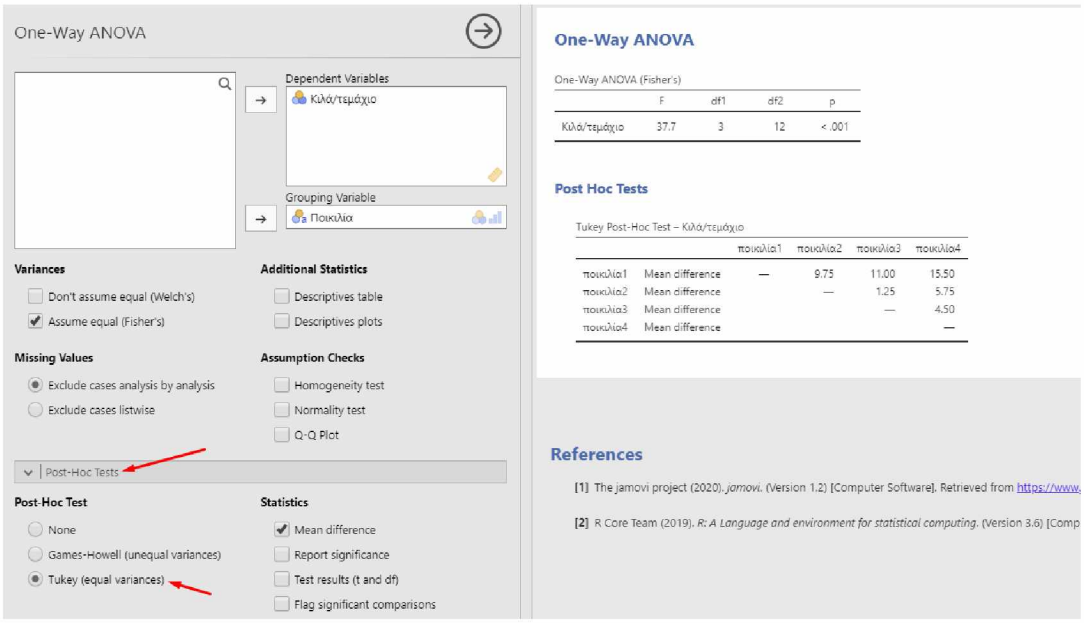


(Εικόνα 4.3)Διεξαγωγή One-Way Anova Test σε Jamovi.

Από τον πίνακα Anova φαίνεται ότι υπάρχει σημαντική διαφοροποίηση στην απόδοση των δύο ποικιλιών (p-value <0.001).

Post-hoc testing

Σε περίπτωση που η αρχική υπόθεση απορριφθεί όπως στο συγκεκριμένο παράδειγμα, μπορεί να γίνει σύγκριση των επιπέδων του παράγοντα (ποικιλίας) μέσω χρήσης της μεθόδου Tukey[4].



(Εικόνα 4.4) Σύγκριση μέσων όρων με την μέθοδο Tukey.

4.2 CRD R:

Η εισαγωγή δεδομένων από ένα αρχείο excel μπορεί να γίνει με την βιβλιοθήκη readxl
library(readxl)
μέσα από την εντολή :
df=read_excel('filename.xlsx')
Με αυτόν τον τρόπο η μεταβλητή df αποτελεί ένα tibble με το σύνολο των δεδομένων του πειράματος.
Πρέπει να σημειωθεί πως το αρχείο θα πρέπει να βρίσκεται στο working directory της R, έτσι ώστε να είναι προσβάσιμο από την εντολή read_excel.

Σε περίπτωση που το αρχείο excel εμπεριέχει ελληνικούς χαρακτήρες θα πρέπει να χρησιμοποιηθεί η εντολή:
Sys.setlocale(category = "LC_ALL", locale = "Greek")
Διαφορετικά οι ελληνικοί χαρακτήρες δεν αναγνωρίζονται από την R.

Μετά την εισαγωγή του πλαισίου δεδομένων όπως αυτό της εικόνας 4.1 μπορεί να γίνει χρήση της εντολής str(df) για τον έλεγχο της μορφής κάθε στήλης του πλαισίου δεδομένων.
str(df)

```
> str(df)
tibble [16 x 2] (S3: tbl_df/tbl/data.frame)
 $ Κιλά/τεμάχιο: num [1:16] 30 26 27 28 15 17 21 19 18 15 ...
 $ Ποικιλία    : chr [1:16] "ποικιλία1" "ποικιλία1" "ποικιλία1" "ποικιλία1" ...
```

(Εικόνα 4.5) Έλεγχος μορφής εισακτέων δεδομένων.

Ακολουθεί ο ορισμός των διαφορετικών ποικιλιών(επεμβάσεων) ως μια κατηγορική μεταβλητή.

p=as.factor(μεταβλητή με το σύνολο των δεδομένων$ονομασία επέμβασης)
ή βάση το παράδειγμα της εικόνας 4.1
p=as.factor(df$Ποικιλία)
Ο σχηματισμός του πίνακα Anova γινεται ως εξης:
results=aov(μεταβλητή απόκρισης ~κατηγορική μεταβλητή, μεταβλητή με το σύνολο των δεδομένων)
summary(results)
ή για το συγκεκριμένο παράδειγμα:
results=aov(`Κιλά/τεμάχιο`~p,df)
summary(results)

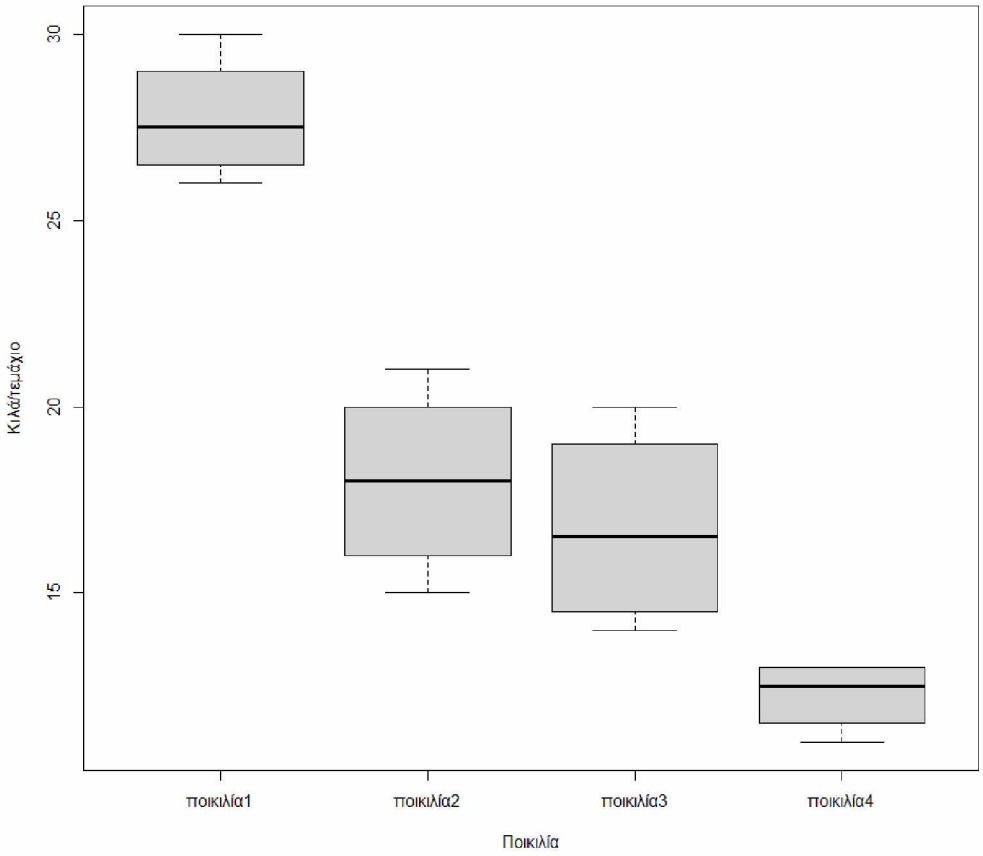
```
            Df Sum Sq Mean Sq F value   Pr(>F)
p            3  511.2  170.40   37.69 2.19e-06 ***
Residuals   12   54.2    4.52
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> 
```

(Εικόνα 4.6) Πινακας One-Way Anova

Από τον πίνακα Anova φαίνεται ότι υπάρχει σημαντική διαφοροποίηση στην απόδοση των δύο ποικιλιών.

Για την δημιουργία ενός πίνακα boxplot έτσι ώστε να είναι εμφανής η κατανομή δεδομένων ανά ποικιλία χρησιμοποιείται η βιβλιοθήκη ggplot2 με τον εξής τρόπο:

```
library(ggplot2)
boxplot(`Κιλά/τεμάχιο`~`Ποικιλία`,df)
```



(Γράφημα 4.1) Box-plot για ποικιλία / κιλά.

Post-hoc testing

Για την διάκριση της σημαντικά διαφορετικής μεταβλητής γίνεται χρήση της μεθόδου Tukey(HSD)[4], η οποία μπορεί να πραγματοποιηθεί μέσω της βιβλιοθήκης stats

```
library(stats)
TukeyHSD(results)
```

```
$p
                        diff        lwr          upr     p adj
ποικιλία2-ποικιλία1  -9.75 -14.213648  -5.28635154 0.0001527
ποικιλία3-ποικιλία1 -11.00 -15.463648  -6.53635154 0.0000477
ποικιλία4-ποικιλία1 -15.50 -19.963648 -11.03635154 0.0000014
ποικιλία3-ποικιλία2  -1.25  -5.713648   3.21364846 0.8386293
ποικιλία4-ποικιλία2  -5.75 -10.213648  -1.28635154 0.0112153
ποικιλία4-ποικιλία3  -4.50  -8.963648  -0.03635154 0.0479544
```

(Εικόνα 4.7) Σύγκριση μέσων όρων με την μέθοδο Tukey.

Από την σύγκριση των μέσων όρων παρατηρείται ότι η ποικιλία 1 είναι η πιο αποδοτική. Η λιγότερο αποδοτική ποικιλία φαίνεται να είναι η ποικιλία 4 ενώ οι ποικιλίες 2 και 3 δεν έχουν σημαντική διαφορά μεταξύ τους μιας και η τιμή p της σύγκρισής τους είναι μεγαλύτερη από 0.05.

Σύνοψη κώδικα

```
library(dplyr)
library(readxl)
library(ggplot2)
library(stats)
df=read_excel('crd2.xlsx')
Sys.setlocale(category = "LC_ALL", locale = "Greek")
p = as.factor(df$Ποικιλία)
results=aov(`Κιλά/τεμάχιο`~p,df)
summary(results)
boxplot(`Κιλά/τεμάχιο`~`Ποικιλία`,df)
TukeyHSD(results)
```

4.3 CRD Python:

Για τον σχηματισμό του πίνακα Anova χρησιμοποιούνται οι βιβλιοθήκες pandas και doex οι οποίες μπορούν να αποκτηθούν με την εντολή:

```
pip install pandas
```

```
pip install doex
```

Μετά την εγκατάστασή τους μπορούν να εισαχθούν στο πρόγραμμα σχεδιασμού του πίνακα anova μέσω των εντολών:

```
import pandas as pd
```

```
import doex
```

Στην συνέχεια γίνεται δημιουργία μεταβλητής που αποτελεί το σύνολο των δεδομένων

```
data = pd.read_excel('filename.xlsx')
```

Ακολουθεί διαμόρφωση στο πλαίσιο δεδομένων έτσι ώστε να είναι κατάλληλο για τα στατιστικά μοντέλα που για το παράδειγμα της εικόνας 4.1 αυτό θα αντιστοιχούσε σε:

```
data = pd.read_excel('crd2.xlsx')
p1 = data.query ('ποικιλία == "ποικιλία1"')['Κιλά/τεμάχιο']
p2 = data.query ('ποικιλία == "ποικιλία2"')['Κιλά/τεμάχιο']
p3 = data.query ('ποικιλία == "ποικιλία3"')['Κιλά/τεμάχιο']
p4 = data.query ('ποικιλία == "ποικιλία4"')['Κιλά/τεμάχιο']
```

Έτσι η μεταβλητή p1 περιέχει τις αποδόσεις της ποικιλιας1 ,η μεταβλητή p2 περιέχει τις αποδόσεις της ποικιλίας2 και ούτω καθεξής.
Δεν υπάρχει περιορισμός στον αριθμό στηλών αρκεί οι αποδόσεις κάθε ποικιλίας να προστεθούν σε μια νέα μεταβλητή.

Τέλος ο πίνακας Anova σχηματίζεται με την εντολή

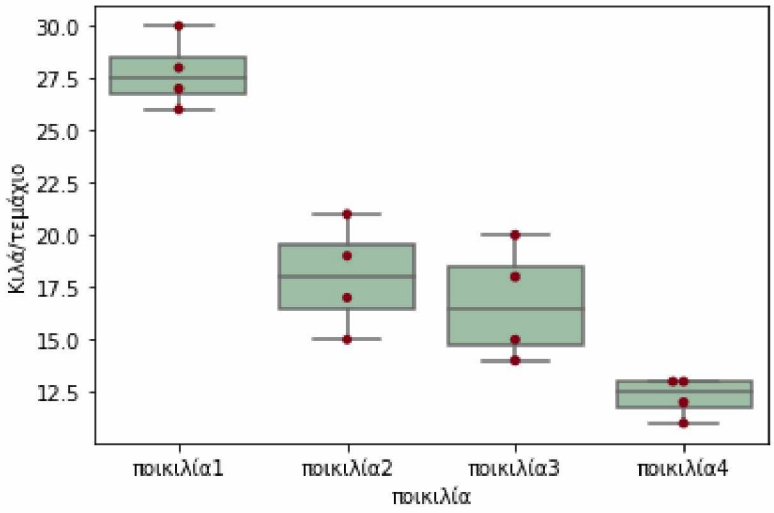
```
doex.CompletelyRandomizedDesign(p1, p2, p3, p4)])
```

Source of Variation	DOF	Sum of Squares	Mean Sum of Squares	F statistic	p value
Treatments	3	511.1875	170.3958	37.6912	0.0000
Error	12	54.2500	4.5208		
Total	15	565.4375			

(Εικόνα 4.8) Πίνακας One-Way Anova

Για την δημιουργία ενός πίνακα boxplot έτσι ώστε να είναι εμφανής η κατανομή δεδομένων ανά ποικιλία χρησιμοποιούνται οι βιβλιοθήκες metaplotlib(Verstion 3.4.3) και seaborn με τον εξής τρόπο:

```
import matplotlib.pyplot as plt
import seaborn as sns
ax = sns.boxplot(x='ποικιλία', y='Κιλά/τεμάχιο', data=data, color='#99c2a2')

ax = sns.swarmplot(x="ποικιλία", y="Κιλά/τεμάχιο", data=data, color='#7d0013')

plt.show()
```



(Γράφημα 4.2) Box-plot για ποικιλία / κιλά.

Post-hoc testing

Για την διάκριση της σημαντικά διαφορετικής μεταβλητής γίνεται χρήση της μεθόδου Tukey(HSD)[4] μέσω της βιβλιοθήκης statsmodels με τον εξής τρόπο

```
from statsmodels.stats.multicomp import pairwise_tukeyhsd
tukey = pairwise_tukeyhsd(endog=μεταβλητή με το σύνολο των δεδομένων['μεταβλητή απόκρισης'],
groups=μεταβλητή με το σύνολο των δεδομένων['κατηγορική μεταβλητή '],alpha=0.05)
```

Για το εξής παράδειγμα η αντίστοιχη εντολή θα ήταν:

```
from statsmodels.stats.multicomp import pairwise_tukeyhsd
tukey = pairwise_tukeyhsd(endog=data['Κιλά/τεμάχιο'],groups=data['ποικιλία'], alpha=0.05)
print(tukey)
```

```
   Multiple Comparison of Means - Tukey HSD, FWER=0.05
=====================================================
  group1    group2  meandiff p-adj   lower    upper   reject
-----------------------------------------------------
ποικιλία1 ποικιλία2    -9.75  0.001 -14.2142  -5.2858   True
ποικιλία1 ποικιλία3    -11.0  0.001 -15.4642  -6.5358   True
ποικιλία1 ποικιλία4    -15.5  0.001 -19.9642 -11.0358   True
ποικιλία2 ποικιλία3    -1.25 0.8246  -5.7142   3.2142  False
ποικιλία2 ποικιλία4    -5.75 0.0112 -10.2142  -1.2858   True
ποικιλία3 ποικιλία4     -4.5  0.048  -8.9642  -0.0358   True
```

(Εικόνα 4.9) Σύγκριση μέσων όρων με την μέθοδο Tukey.

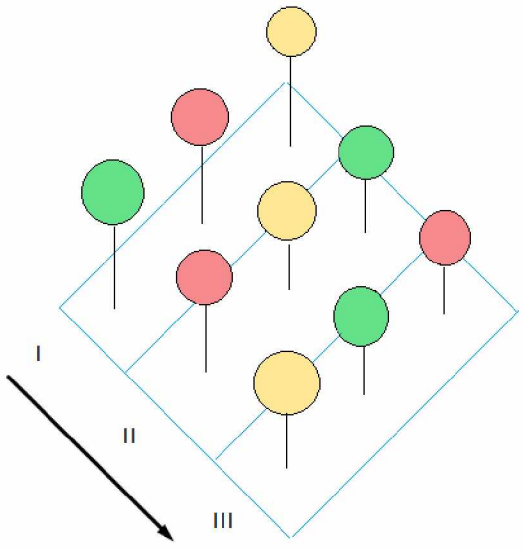
Από την σύγκριση των μέσων όρων παρατηρείται ότι η ποικιλία 1 είναι η πιο αποδοτική. Η λιγότερο αποδοτική ποικιλία φαίνεται να είναι η ποικιλία 4 ενώ οι ποικιλίες 2 και 3 δεν έχουν σημαντική διαφορά μεταξύ τους μιας και η τιμή p της σύγκρισής τους είναι μεγαλύτερη από 0.05.

Σύνοψη κώδικα

```
import pandas as pd
import doex
data = pd.read_excel('crd2.xlsx')
p1 = data.query('ποικιλία=="ποικιλία1"')['Κιλά/τεμάχιο']
p2 = data.query('ποικιλία == "ποικιλία2"')['Κιλά/τεμάχιο']
p3 = data.query('ποικιλία =="ποικιλία3"')['Κιλά/τεμάχιο']
p4 = data.query('ποικιλία == "ποικιλία4"')['Κιλά/τεμάχιο']
doex.CompletelyRandomizedDesign(p1, p2, p3, p4)
import matplotlib.pyplot as plt
import seaborn as sns
ax = sns.boxplot(x='ποικιλία', y='Κιλά/τεμάχιο', data=data, color='#99c2a2')
ax = sns.swarmplot(x="ποικιλία", y="Κιλά/τεμάχιο", data=data, color='#7d0013')
plt.show()
from statsmodels.stats.multicomp import pairwise_tukeyhsd
tukey = pairwise_tukeyhsd(endog=data['Κιλά/τεμάχιο'],groups=data['ποικιλία'], alpha=0.05)
```

5. Σχεδιασμοί τυχαιοποιημένων πλήρων ομάδων (RCBD)

Ο τυχαιοποιημένος πλήρως σχεδιασμός κατά μπλοκ αποτελεί τον πιο διαδεδομένος πειραματικό σχεδιασμό στην γεωργία[5].Στο πλήρως τυχαιοποιημένο σχέδιο (CRD) οι εξωτερικοί παράγοντες επιδρούν και έχουν το ίδιο αποτέλεσμα σε όλες τις υπό εξέταση πειραματικές μονάδες. Αυτό όμως δεν συμβαίνει πάντα. Στην πραγματικότητα, είναι δυνατόν να υπάρχει ένας συγχυτικός παράγοντας(nuisance factor) ο οποίος να επηρεάζει και αυτός τη μεταβλητή απόκρισης κάθε πειραματικής μονάδας. Στην περίπτωση αυτή εάν ο συγχυτικός παράγοντας είναι γνωστός και η μεταβλητότητα που προκαλεί μπορεί να απομονωθεί, χρησιμοποιείται ένας σχεδιασμός τυχαιοποιημένων πλήρων ομάδων (RCBD)[6]. Στον σχεδιασμό αυτό κατασκευάζονται μπλοκ (ομάδες) από πειραματικές μονάδες που παρουσιάζουν ομοιογένεια με σκοπό την ελαχιστοποίηση της μεταβλητότητας σε κάθε μπλοκ[7],ενώ παράλληλα αυξάνεται η ικανότητα του Anova test να ανιχνεύει και να εκτιμάει τις επιπτώσεις του υπό εξέταση παράγοντα[8]. Για παράδειγμα στο χωράφι στο οποίο διεξάγεται ένα πείραμα αγρού υπάρχει κλίση η οποία καθιστά το υψομετρικά χαμηλότερο μέρος του χωραφιού πιο γόνιμο. Εάν ο υπό εξέταση πρωταρχικός παράγοντας είναι η απόδοση τριών διαφορετικών ποικιλιών (Treatments) τότε η θέση τους στο χωράφι αποτελεί τον συγχυτικό παράγοντα. Για να αντιμετωπιστεί η μεταβλητότητα ανάμεσα στις πειραματικές μονάδες (φυτά) γίνεται σχεδιασμός τυχαιοποιημένων πλήρων ομάδων. Ομαδοποιώντας τα φυτά ανάλογα με την θέση τους επιτυγχάνεται αποκλεισμός του συγχυτικού παράγοντα και μείωση του πειραματικού σφάλματος, αυξάνοντας έτσι την ακρίβεια του πειράματος. Είναι απαραίτητο ο αριθμός των ποικιλιών σε κάθε block να είναι ίσος με τον αριθμό των επιπέδων του παράγοντα ποικιλία. Με την προϋπόθεση ότι κάθε ποικιλία συμπεριλαμβάνεται από μία φορά σε κάθε block, η σειρά τοποθέτησης των ποικιλιών είναι τυχαία για κάθε block.



(Εικόνα 5.1) Σχεδιασμός RCBD πειράματος.

Στην Εικόνα 5.1 οι τρεις ποικιλίες είναι η κόκκινη = A , η κίτρινη = B και η πράσινη = C ενώ κάθε οριζόντια γραμμή αποτελεί ένα διαφορετικό block. Κάθε οριζόντια γραμμή περιέχει τρία διαφορετικά επίπεδα.

Block	Ποικιλία(Treatment)		
I	C	A	B
II	A	B	C
III	B	C	A

(Πίνακας 5.1) Μορφή δεδομένων RCBD πειράματος.

Παρακάτω θα αναλυθεί πως μπορεί να γίνει ο υπολογισμός του πίνακα Anova μέσω ενός αρχείου excel χρησιμοποιώντας το στατιστικό λογισμικό Jamovi, την γλώσσα προγραμματισμού R αλλά και την γλώσσα Python.

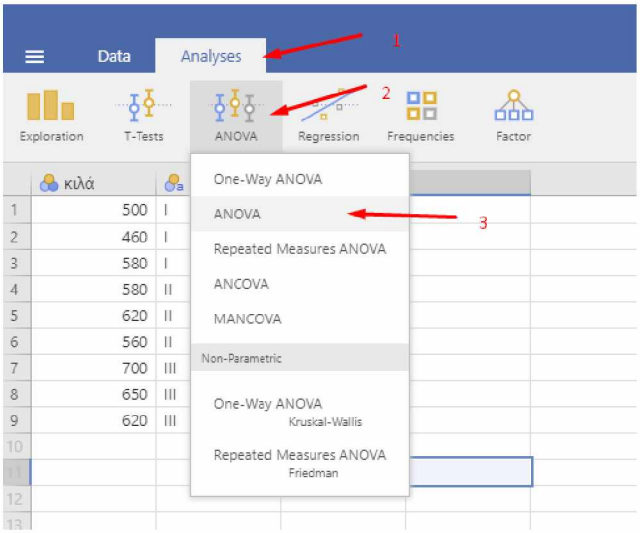
Ως παράδειγμα θα ληφθεί ένα πείραμα RCBD κατά το οποίο συγκρίνονται οι αποδόσεις τριών ποικιλιών σόγιας σε κιλά ανά τεμάχιο. Τα δεδομένα του παραδείγματος φαίνονται στην εικόνα 5.2.

κιλά	block	ποικιλία
500	I	C
460	I	A
580	I	B
580	II	A
620	II	B
560	II	C
700	III	B
650	III	C
620	III	A

(Εικόνα 5.2) Πλαίσιο δεδομένων πειράματος RCBD.

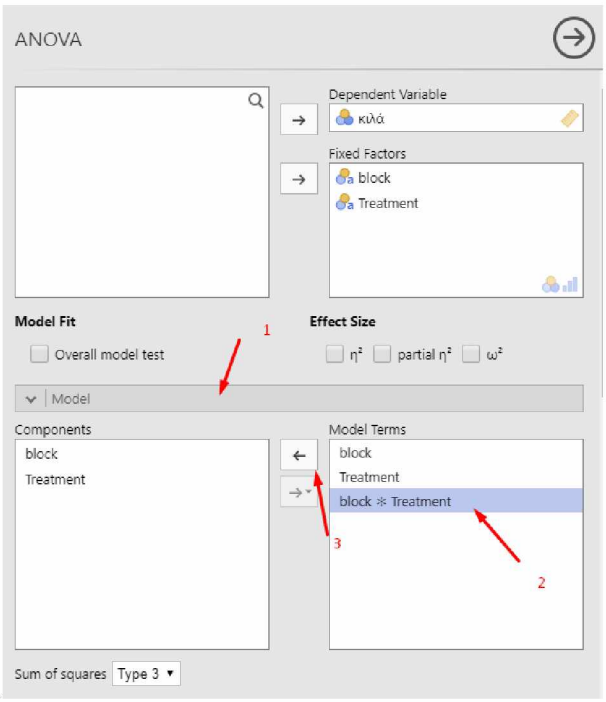
5.1 RCBD Jamovi:

Με την εισαγωγή του αρχείου excel της εικόνας 5.2 στο jamovi μπορεί να εκτελεστεί το Two-Way Anova test



(Εικόνα 5.3) Επιλογή Two-Way Anova.

Ως dependent variable τίθεται η μεταβλητή απόκρισης δηλαδή τα κιλά ενώ έως fixed factors η κατηγορική μεταβλητή , δηλαδή η ποικιλία και τα διαφορετικά block .
Λόγο του ότι η επίδραση της υψομετρικής διαφοράς θα πρέπει να απομονωθεί το block*Treatment θα πρέπει να αφαιρεθεί από τα Model Terms.



(Εικόνα 5.4) Διεξαγωγή Two-Way Anova Test σε Jamovi.

Στα δεξιά σχηματίζεται ο πίνακας Anova από τον οποίο φαίνεται πως υπάρχει σημαντική διαφορά για τις ποικιλίες και τα block.

ANOVA

ANOVA - κιλά

	Sum of Squares	df	Mean Square	F	p
block	30822	2	15411	32.3	0.003
Treatment	10689	2	5344	11.2	0.023
Residuals	1911	4	478		

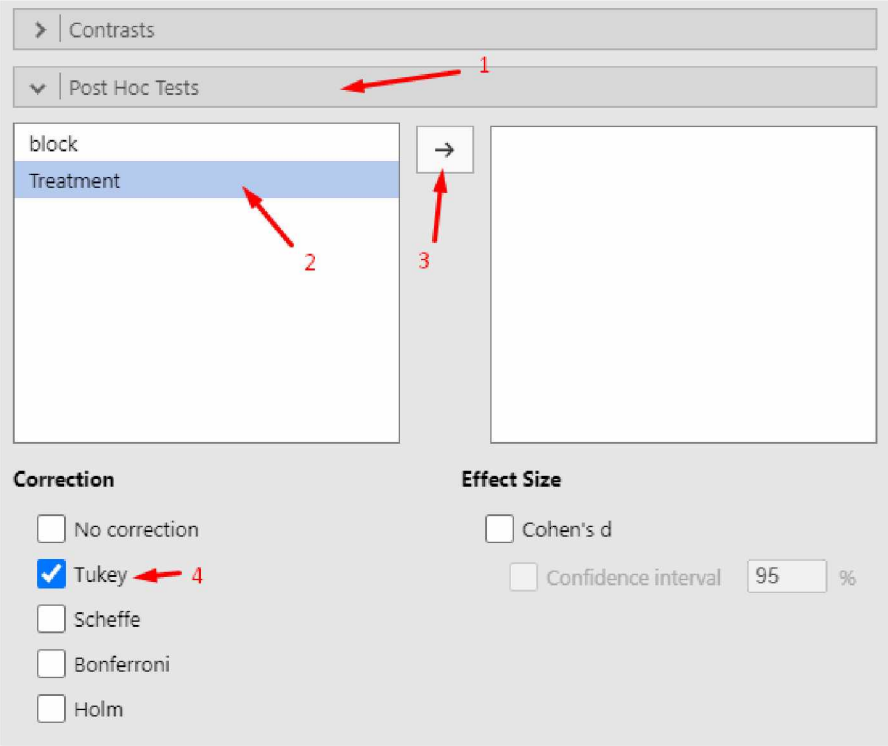
[3]

(Εικόνα 5.5) Πίνακας Two-Way Anova.

Συνεπώς η ποικιλία έχει εξαιρετικά σημαντική επίδραση στα παραγόμενα κιλά ($p < 0.01$) ενώ και η υψομετρική θέση των ποικιλιών στο χωράφι έχει σημαντική επίδραση στην απόδοση τους($p < 0.05$).

Post-hoc testing

Τέλος για να γίνει διάκριση των επιπέδων της μεταβλητής ποικιλίας μέσω χρήσης της μεθόδου Tukey επιλέγονται οι μεταβλητές από το Post Hoc Tests βάση των οποίων θα εφαρμοστεί η μέθοδος Tukey.



(Εικόνα 5.6) Διεξαγωγή Tukey test σε Jamovi.

Post Hoc Tests

Post Hoc Comparisons - Treatment

Comparison							
Treatment		Treatment	Mean Difference	SE	df	t	p_{tukey}
A	-	B	−80.0	17.8	4.00	−4.483	0.024
	-	C	−16.7	17.8	4.00	−0.934	0.650
B	-	C	63.3	17.8	4.00	3.549	0.051

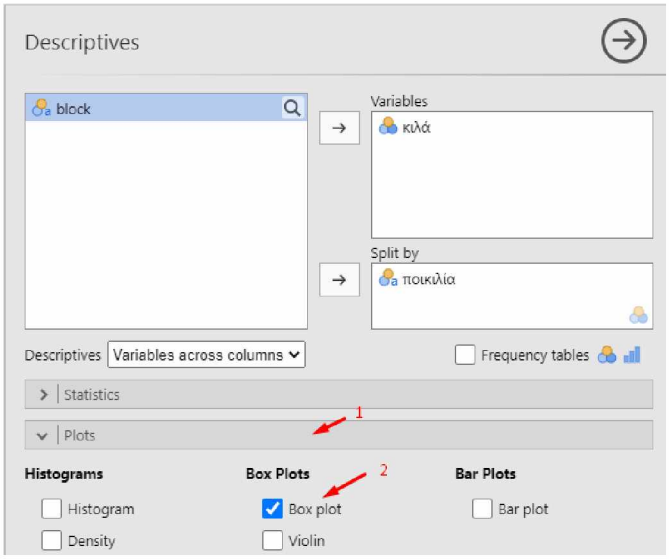
Note. Comparisons are based on estimated marginal means

(Εικόνα 5.7) Σύγκριση μέσων όρων με την μέθοδο Tukey.

Είναι εμφανές ότι η ποικιλία B έχει μεγαλύτερη απόδοση από την ποικιλία A , ενώ η ποικιλία C δεν διαφέρει στατιστικά σημαντικά από τις ποικιλίες A και B ($p > 0.05$).

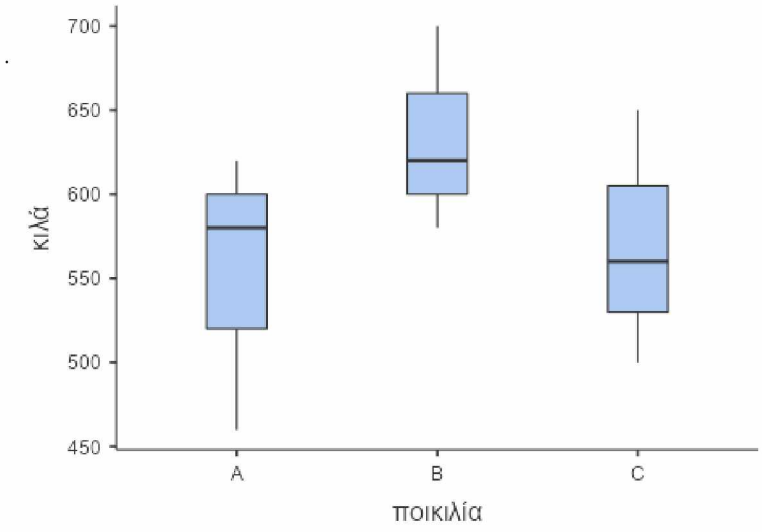
Η δημιουργία ενός πίνακα boxplot έτσι ώστε να είναι εμφανής η κατανομή δεδομένων ανά ποικιλία μπορεί να γίνει από την καρτέλα “Analyses”, μέσω της επιλογής “Exploration” και στην συνέχεια “Descriptives”.

Ως Variables τίθεται η μεταβλητή απόκρισης δηλαδή τα κιλά ενώ στο Split by τοποθετείται μια κατηγορική μεταβλητή , δηλαδή η ποικιλία. Τέλος από το παράρτημα “Plots” επιλέγεται το “Box plot”.



(Εικόνα 5.8)Σχηματισμός γραφήματος box-plot σε Jamovi.

.



(Γράφημα 5.1) Box-plot Split By ποικιλία.

5.2 RCBD R:

Η εισαγωγή δεδομένων από ένα αρχείο excel μπορεί να γίνει με την βιβλιοθήκη readxl
library(readxl)
μέσα από την εντολή :
df=read_excel('filename.xlsx')
Με αυτόν τον τρόπο η μεταβλητή df αποτελεί το σύνολο των δεδομένων .
Σε περίπτωση που το αρχείο excel εμπεριέχει ελληνικούς χαρακτήρες θα πρέπει να χρησιμοποιηθεί η εντολή:
Sys.setlocale(category = "LC_ALL", locale = "Greek")
Διαφορετικά οι ελληνικοί χαρακτήρες δεν αναγνωρίζονται από την R.
Μετά την εισαγωγή του πλαισίου δεδομένων όπως αυτό της εικόνας 5.2 μπορεί να γίνει ανάλυση των δεδομένων με 2-way Anova και ο σχηματισμός του πίνακα Anova με τις εντολές:
two.way = aov(μεταβλητή απόκρισης ~ κατηγορική μεταβλητή + block, data = df)
summary(two.way)
Για το παράδειγμα της εικόνας 5.1:
two.way = aov(κιλά ~ ποικιλία + block, data = df)
summary(two.way)

```
            Df Sum Sq Mean Sq F value  Pr(>F)
ποικιλία     2  10689    5344   11.19 0.02301 *
block        2  30822   15411   32.26 0.00341 **
Residuals    4   1911     478
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

(Εικόνα 5.9) Πίνακας Two-Way Anova.

Από τον πίνακα Anova φαίνεται πως υπάρχει σημαντική διαφορά για τις ποικιλίες και τα block.

Post-hoc testing

Για την διάκριση της σημαντικά διαφορετικής μεταβλητής γίνεται χρήση της μεθόδου Tukey(HSD) [4] η οποία μπορεί να πραγματοποιηθεί μέσω της βιβλιοθήκης stats.

```
library(stats)
TukeyHSD(two.way)
```

```
$ποικιλία
         diff        lwr         upr     p adj
B-A  80.00000   16.39318 143.6068180 0.0237517
C-A  16.66667  -46.94015  80.2734847 0.6504919
C-B -63.33333 -126.94015   0.2734847 0.0506717
```

(Εικόνα 5.10) Σύγκριση μέσων όρων με την μέθοδο Tukey.

Είναι εμφανές ότι η ποικιλία B έχει μεγαλύτερη απόδοση από την ποικιλία A , ενώ η ποικιλία C δεν διαφέρει στατιστικά σημαντικά από την ποικιλία B ($p > 0.05$). Στην περίπτωση αυτή μπορεί να γίνει υπολογισμός του συντελεστή διακύμανσης των ποικιλιών με σκοπό την μεταξύ τους σύγκριση. Για τον υπολογισμό του συντελεστή διακύμανσης χρησιμοποιείται η βιβλιοθήκη dplyr.

```
library(dplyr)
```

Ο υπολογισμός των συντελεστών διακύμανσης γίνεται με τον εξής τρόπο:

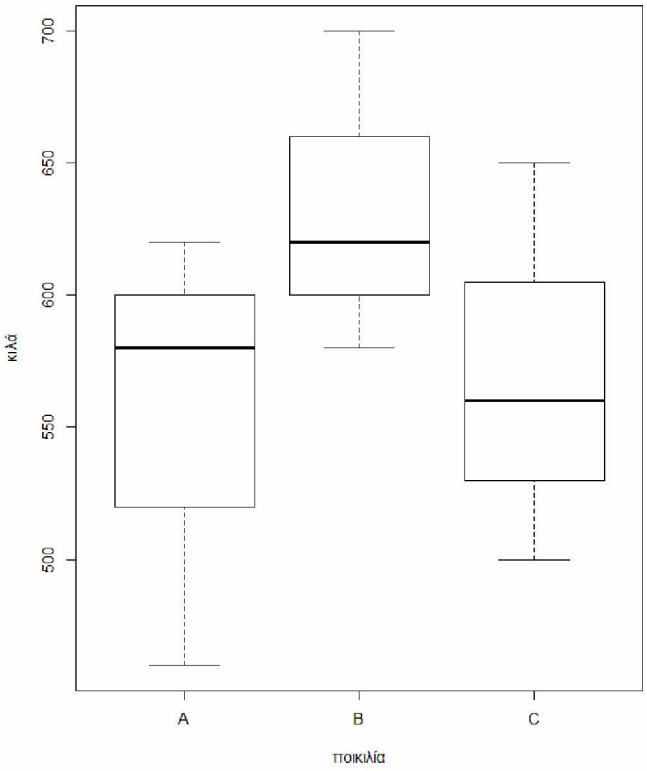
```
dfb = filter(df, ποικιλία == "B")
dfc = filter(df, ποικιλία == "C")
CVB = sd(dfb$`κιλά`) / mean(dfb$`κιλά`)
CVC = sd(dfc$`κιλά`) / mean(dfc$`κιλά`)

CVB
CVC
```

```
> CVB
[1] 0.09647528
> CVC
[1] 0.1324532
```

(Εικόνα 5.11) Υπολογισμός συντελεστών διακύμανσης

Για την δημιουργία ενός πίνακα boxplot έτσι ώστε να είναι εμφανής η κατανομή δεδομένων ανά ποικιλία μπορεί να γίνει χρήση της εντολής:

```
boxplot(κιλά~ποικιλία, data = df)
```



(Γράφημα 5.2) Box-plot Split By ποικιλία

Σύνοψη κώδικα

```
library(readxl)
library(stats)
df=read_excel('rcbd.xlsx')
Sys.setlocale(category = "LC_ALL", locale = "Greek")
two.way = aov(κιλά ~ ποικιλία + block, data = df)
summary(two.way)
TukeyHSD(two.way)
library(dplyr)
dfa = filter(df, ποικιλία == "A")
dfb = filter(df, ποικιλία == "B")
dfc = filter(df, ποικιλία == "C")
CVB = sd(dfb$`κιλά`) / mean(dfb$`κιλά`)
CVC = sd(dfc$`κιλά`) / mean(dfc$`κιλά`)
CVB
CVC
boxplot(κιλά~ποικιλία, data = df)
```

5.3 RCBD Python:

Η ανάλυση των δεδομένων που προκύπτουν από ένα RCBD πείραμα γίνεται με 2-way Anova. Για τον σκοπό αυτό στην python μπορούν να χρησιμοποιηθούν οι εξής βιβλιοθήκες:

```
import pandas as pd
import doex
```

Στην συνέχεια γίνεται δημιουργία μεταβλητής που αποτελεί το σύνολο των δεδομένων

```
data = pd.read_excel('filename.xlsx')
```

Ακολουθεί διαμόρφωση στο πλαίσιο δεδομένων έτσι ώστε να είναι κατάλληλο για τα στατιστικά μοντέλα .

```
a = data.query('treatment == "A"')[' μεταβλητή απόκρισης ']
```

Για το παράδειγμα της εικόνας 5.1 αυτό θα αντιστοιχούσε σε:

```
a = data.query('ποικιλία == "A"')['κιλά']
b = data.query('ποικιλία == "B"')['κιλά']
c = data.query('ποικιλία == "C"')['κιλά']
```

Ο σχηματισμός του πίνακα Anova γίνεται με την εντολή:

```
doex.RandomizedCompleteBlockDesign([a,b,c,])
```

Source of Variation	DOF	Sum of Squares	Mean Sum of Squares	F statistic	p value
Treatments	2	10688.8889	5344.4444	11.1860	0.0230
Blocks	2	30822.2222	15411.1111	32.2558	0.0034
Error	4	1911.1111	477.7778		
Total	8	43422.2222			

(Εικόνα 5.12) Πίνακας Two-Way Anova.

Από τον πίνακα Anova φαίνεται πως υπάρχει σημαντική διαφορά για τις ποικιλίες και τα block.

Post-hoc testing

Για την διάκριση των σημαντικά διαφορετικών μεταβλητών γίνεται χρήση της μεθόδου Tukey(HSD) [4], για τις ποικιλίες.

```
pip install bioinfokit
from bioinfokit.analys import stat
res = stat()
res.tukey_hsd(data,'κιλά','ποικιλία', anova_model='κιλά~C(ποικιλία)+C(block)')
print(res.tukey_summary)
```

	group1	group2	Diff	Lower	Upper	q-value	p-value
0	C	A	16.666667	-46.850751	80.184084	1.320676	0.649098
1	C	B	63.333333	-0.184084	126.850751	5.018570	0.050451
2	A	B	80.000000	16.482583	143.517417	6.339247	0.023637

(Εικόνα 5.13) Σύγκριση μέσων όρων με την μέθοδο Tukey.

Είναι εμφανές ότι η ποικιλία B έχει μεγαλύτερη απόδοση από την ποικιλία A , ενώ η ποικιλία C δεν διαφέρει στατιστικά σημαντικά από τις ποικιλίες A και B ($p > 0.05$). Στην περίπτωση αυτή γίνεται υπολογισμός του συντελεστή διακύμανσης των ποικιλιών με σκοπό την μεταξύ τους σύγκριση. Για τον υπολογισμό του συντελεστή διακύμανσης χρησιμοποιείται η βιβλιοθήκη statistics.

```
import statistics
```

Για την ποικιλία A ο υπολογισμός γίνεται ως εξής:

```
print(statistics.stdev(a)/statistics.mean(a))
```

Αντίστοιχα για την ποικιλία B:

```
print(statistics.stdev(b)/statistics.mean(b))
```

Τέλος για την ποικιλία C:

```
print(statistics.stdev(c)/statistics.mean(c))
```

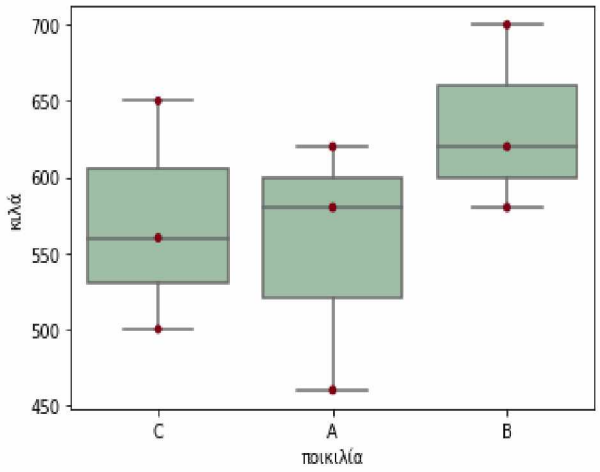
```
In [28]: print(statistics.stdev(a)/statistics.mean(a))
0.15048187947947947

In [29]: print(statistics.stdev(b)/statistics.mean(b))
0.09647527778854399

In [30]: print(statistics.stdev(c)/statistics.mean(c))
0.13245323570650436
```

(Εικόνα 5.14) Υπολογισμός συντελεστών διακύμανσης

Από την μεταξύ τους σύγκριση φαίνεται πως η ποικιλία B είναι πιο παραγωγική την C ενώ η ποικιλία A είναι η λιγότερο παραγωγική ποικιλία.

Για την δημιουργία ενός πίνακα boxplot έτσι ώστε να είναι εμφανής η κατανομή δεδομένων ανά ποικιλία και block χρησιμοποιούνται οι βιβλιοθήκες metaplotlib και seaborn με τον εξής τρόπο:

```
import matplotlib.pyplot as plt
import seaborn as sns
ax = sns.boxplot(x='ποικιλία', y='κιλά', data=data, color='#99c2a2')
ax = sns.swarmplot(x="ποικιλία", y="κιλά", data=data, color='#7d0013')
plt.show()
```



(Γράφημα 5.3) Box-plot Split By ποικιλία

Σύνοψη κώδικα

```
pip install bioinfokit
import pandas as pd
import doex
data = pd.read_excel('rcbd.xlsx')
a = data.query('ποικιλία == "A"')['κιλά']
b = data.query('ποικιλία == "B"')['κιλά']
c = data.query('ποικιλία == "C"')['κιλά']
doex.RandomizedCompleteBlockDesign([a,b,c,])
from bioinfokit.analys import stat
res = stat()
res.tukey_hsd(data,'κιλά','ποικιλία', anova_model='κιλά~C(ποικιλία)+C(block)')
print(res.tukey_summary)
import statistics
print(statistics.stdev(b)/statistics.mean(b))
print(statistics.stdev(c)/statistics.mean(c))
import matplotlib.pyplot as plt
import seaborn as sns
ax = sns.boxplot(x='ποικιλία', y='κιλά', data=data, color='#99c2a2')
ax = sns.swarmplot(x="ποικιλία", y="κιλά", data=data, color='#7d0013')
plt.show()
```

6. Λατινικό τετράγωνο

Ένα λατινικό τετράγωνο της τάξης n είναι ένας nxn πίνακας κελιών στον οποίο τοποθετούνται n σύμβολα(πειραματική μονάδα),ένα ανά κελί, έτσι ώστε το κάθε σύμβολο επίπεδο του παράγοντα ενδιαφέροντος να εμφανίζεται μόνο μια φορά σε κάθε γραμμή και στήλη[9]. Με την ομαδοποίηση αυτή των πειραματικών μονάδων σε σειρές και στήλες επιτυγχάνεται ο ταυτόχρονος έλεγχος δύο γνωστών συγχυτικών παραγόντων (nuisance factors)[10].

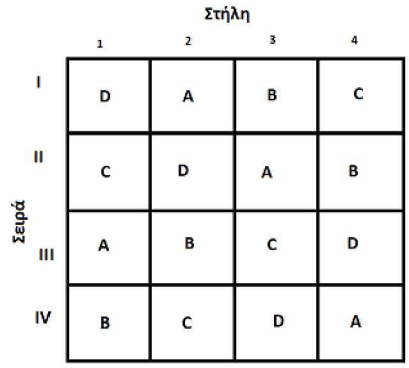
6.1 Τυχαιοποίηση του Λατινικού τετραγώνου

Για την ελαχιστοποίηση του στατιστικού σφάλματος οι στήλες και οι σειρές του πίνακα θα πρέπει να τοποθετηθούν με τυχαιοποιημένη διάταξη. Αρχικά έστω ότι έχουμε ένα 4x4 λατινικό τετράγωνο. Δημιουργείται ο πίνακας με τα σύμβολα των κελιών τοποθετημένα σε αλφαβητική σειρά. Ξεκινάμε με την πρώτη γραμμή (A,B,C,D) και για να δημιουργήσουμε τη δεύτερη μετακινούμε στα αριστερά τα γράμματα κυκλικά με το A να πηγαίνει στο τέλος. Αντίστοιχα κατασκευάζουμε τις υπόλοιπες γραμμές.

Σειρά \ Στήλη	1	2	3	4
I	A	B	C	D
II	B	C	D	A
III	C	D	A	B
IV	D	A	B	A

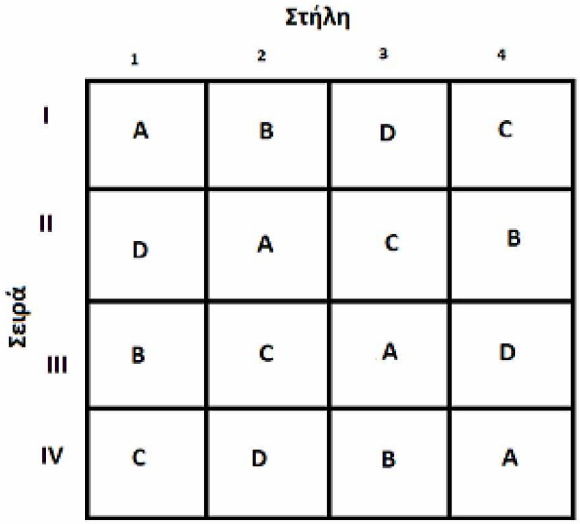
(Εικόνα 6.1)Συμμετρική διάταξη.

Στην συνέχεια γίνεται η τυχαιοποίηση των σειρών του πίνακα. Η τυχαιοποίηση μπορεί να γίνει αντιστοιχώντας κάθε σειρά με έναν τυχαίο αριθμό. Η σειρά με τον μεγαλύτερο τυχαίο αριθμό θα τοποθετηθεί πρώτη, η σειρά με τον δεύτερο μεγαλύτερο αριθμό θα τοποθετηθεί δεύτερη και ούτω καθεξής.[11]



(Εικόνα 6.2)Τυχαιοποιημένη διάταξη σειρών.

Με τον ίδιο τρόπο ακολουθεί η τυχαιοποίηση της διάταξης της κάθε στήλης η οποία θα σχηματίσει τον τελικό πίνακα.



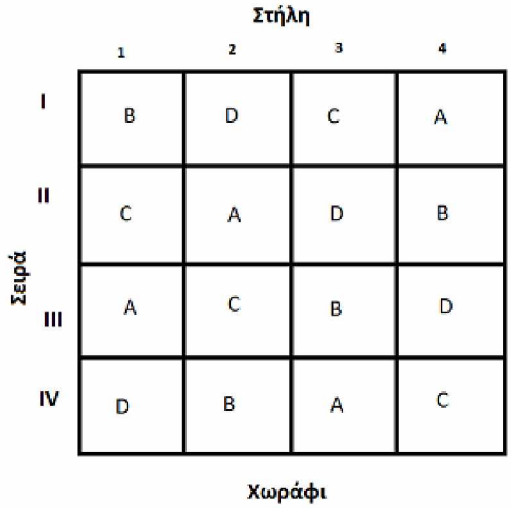
(Εικόνα 6.3)Τυχαιοποιημένη διάταξη σειρών και στηλών.

Τέλος τυχαιοποιούμε τις μεταχειρίσεις.

6.2 Παράδειγμα εφαρμογής Λατινικού τετραγώνου

Το λατινικό τετράγωνο είναι καλύτερο να χρησιμοποιείται όταν γίνονται τέσσερις έως οχτώ επεμβάσεις.[12] Για παράδειγμα, σε ένα χωράφι η γονιμότητα του εδάφους μπορεί να μεταβάλλεται προς όλες τις κατευθύνσεις, από το βόρειο μέρος προς το νότιο και από το ανατολικό προς το δυτικό. Στο συγκεκριμένο χωράφι διεξάγεται πείραμα στο οποίο μελετάται η απόδοση τεσσάρων διαφορετικών ποικιλιών καλαμποκιού(A,B,C,D). Η διαφορά γονιμότητας που προκύπτει από την ανομοιομορφία του χωραφιού μπορεί να μεταφραστεί σε δύο διαφορετικούς συγχυτικούς παράγοντες (nuisance factors).Στην εικόνα 6.4, το τετράγωνο 1Ι

αντιστοιχεί στην βορειοδυτική γωνία του χωραφιού ενώ το τετράγωνο 4IV στην νοτιοανατολική.



(Εικόνα 6.4) Τελική διάταξη πειράματος.

Μετά τον σχηματισμό του λατινικού μοντέλου ακολουθεί η λήψη των δεδομένων στα οποία θα γίνει στατιστική ανάλυση. Τα δεδομένα αυτά φαίνονται στην εικόνα 6.5. Παρακάτω θα αναλυθεί πως μπορεί να γίνει ο υπολογισμός του πίνακα Anova για το συγκεκριμένο παράδειγμα χρησιμοποιώντας το στατιστικό λογισμικό Jamovi, την γλώσσα προγραμματισμού R αλλά και την γλώσσα Python.

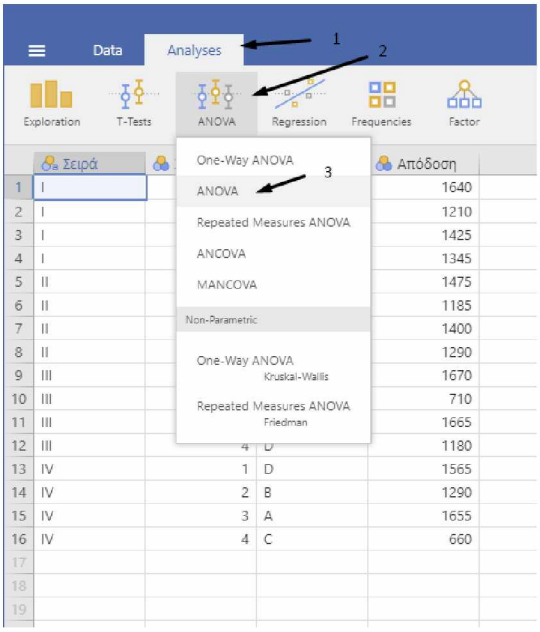
Σειρά	Στήλη	Ποικιλία	Απόδοση
I	1	B	1640
I	2	D	1210
I	3	C	1425
I	4	A	1345
II	1	C	1475
II	2	A	1185
II	3	D	1400
II	4	B	1290
III	1	A	1670
III	2	C	710
III	3	B	1665
III	4	D	1180
IV	1	D	1565
IV	2	B	1290
IV	3	A	1655
IV	4	C	660

(Εικόνα 6.5)Πλαίσιο δεδομένων πειράματος προς ανάλυση.

.

6.3 Λατινικό τετράγωνο Jamovi

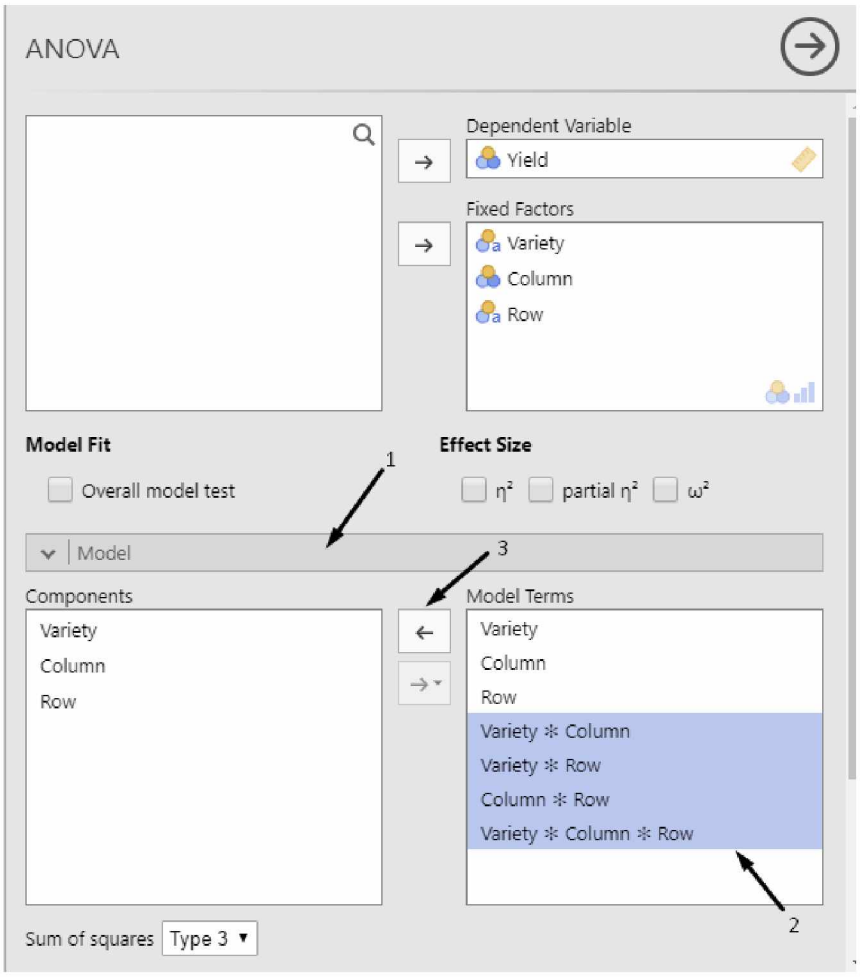
Με την εισαγωγή του αρχείου excel στο jamovi μπορεί να εκτελεστεί το Two-Way Anova test



(Εικόνα 6.6) Επιλογή Two-Way Anova.

Ως dependent variable τίθεται η μεταβλητή απόκρισης δηλαδή η απόδοση ενώ έως fixed factors η κατηγορική μεταβλητή , δηλαδή η ποικιλία και τα διαφορετικά block (Σειρά-Στήλη) .

Λόγο του ότι η μόνο η επίδραση της διαφορετικής ποικιλίας δεν θα πρέπει να απομονωθεί, το block*Treatment θα πρέπει να αφαιρεθεί από τα Model Terms καθώς υποθέτουμε ότι δεν υπάρχουν αλληλεπιδράσεις μεταξύ των παραγόντων.



(Εικόνα 6.7) Αφαίρεση όρων από τα Model terms.

Στην δεξιά μεριά του παραθύρου σχηματίζεται ο πίνακας Anova από τον οποίο φαίνεται πως υπάρχει σημαντική διαφοροποίηση της απόδοσης σε κάθε στήλη αλλά και εξαιτίας των διαφορετικών ποικιλιών.

ANOVA

ANOVA - Yield

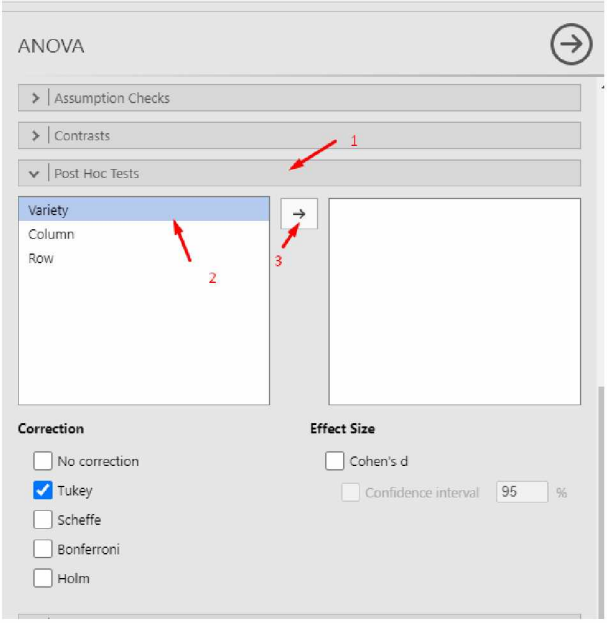
	Sum of Squares	df	Mean Square	F	p
Variety	426842	3	142281	6.588	0.025
Column	827342	3	275781	12.769	0.005
Row	30155	3	10052	0.465	0.717
Residuals	129584	6	21597		

[3]

(Εικόνα 6.8) Πίνακας Two-Way Anova

Post-hoc testing

Τέλος για να γίνει διάκριση της σημαντικά διαφορετικής μεταβλητής(ποικιλίας) μέσω χρήσης της μεθόδου Tukey επιλέγονται οι μεταβλητές από το Post Hoc Tests βάση των οποίων θα εφαρμοστεί η μέθοδος Tukey[4].



(Εικόνα 6.9) Διεξαγωγή Tukey test σε Jamovi.

Στην δεξιά μεριά του παραθύρου κάτω από τον προσχηματισμένο πίνακα Anova , σχηματίζεται ο πίνακας αποτελεσμάτων του test σύγκρισης μέσων όρων.

Post Hoc Comparisons - Variety

Comparison							
Variety		Variety	Mean Difference	SE	df	t	p_{tukey}
A	-	B	−7.50	104	6.00	−0.0722	1.000
	-	C	396.25	104	6.00	3.8131	0.034
	-	D	125.00	104	6.00	1.2029	0.647
B	-	C	403.75	104	6.00	3.8853	0.031
	-	D	132.50	104	6.00	1.2751	0.608
C	-	D	−271.25	104	6.00	−2.6103	0.138

Note. Comparisons are based on estimated marginal means

(Εικόνα 6.10) Σύγκριση μέσων όρων με την μέθοδο Tukey.

6.4 Λατινικό τετράγωνο R

Η εισαγωγή δεδομένων από ένα αρχείο excel μπορεί να γίνει με την βιβλιοθήκη readxl
library(readxl)
μέσα από την εντολή :
df=read_excel('filename.xlsx')
Με αυτόν τον τρόπο η μεταβλητή df αποτελεί το σύνολο των δεδομένων .
Σε περίπτωση που το αρχείο excel εμπεριέχει ελληνικούς χαρακτήρες θα πρέπει να χρησιμοποιηθεί η εντολή:
Sys.setlocale(category = "LC_ALL", locale = "Greek")
Διαφορετικά οι ελληνικοί χαρακτήρες δεν αναγνωρίζονται από την R.
Μετά την εισαγωγή του πλαισίου δεδομένων όπως αυτό της εικόνας 6.5 μπορεί να γίνει χρήση της εντολής str(df) για τον έλεγχο της μορφής κάθε στήλης του πλαισίου δεδομένων.
str(df)

```
> str(df)
tibble [24 x 7] (S3: tbl_df/tbl/data.frame)
 $ Σειρά   : chr [1:24] "I" "I" "I" "I" ...
 $ Στήλη   : num [1:24] 1 2 3 4 1 2 3 4 1 2 ...
 $ Ποικιλία: chr [1:24] "B" "D" "C" "A" ...
 $ Απόδοση : num [1:24] 1640 1210 1425 1345 1475 ...
```

(Εικόνα 6.11) Έλεγχος μορφής εισακτέων δεδομένων.

Σε αυτό το παράδειγμα η Σειρά, Στήλη και Ποικιλία διαβάζονται ως χαρακτήρες, αριθμοί και χαρακτήρες αντίστοιχα. Θα πρέπει αρχικά να γίνει η μετατροπή τους σε παράγοντες (factors) με την εντολή as.factor(filename$variabel) ή πιο συγκεκριμένα για το παρών παράδειγμα:

df$Σειρά = as.factor(df$Σειρά)
df$Στήλη = as.factor(df$Στήλη)
df$Ποικιλία= as.factor(df$Ποικιλία)
Μετά την σωστή διαμόρφωση των δεδομένων μπορεί να γίνει η ανάλυση της διακύμανσης με την εντολή:

analysis = aov(ΜεταβλητήΑπόκρισης ~ Σειρά+Στήλη+Θεραπεία,filename)
anova(analysis)

Πιο συγκεκριμένα για το παραπάνω παράδειγμα η εντολή αυτή θα ήταν:

analysis = aov(Απόδοση ~ Σειρά+Στήλη+Ποικιλία,df)
anova(analysis)

```
Analysis of Variance Table

Response: Απόδοση
          Df Sum Sq Mean Sq F value   Pr(>F)
Σειρά      3  30155   10052  0.4654 0.716972
Στήλη      3 827342  275781 12.7692 0.005148 **
Ποικιλία   3 426842  142281  6.5879 0.025092 *
Residuals  6 129584   21597
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

(Εικόνα 6.12) Πίνακας Two-Way Anova.

Από τον πίνακα Anova φαίνεται ότι υπάρχει σημαντική διαφοροποίηση στην απόδοση σε κάθε στήλη αλλά και εξαιτίας των διαφορετικών ποικιλιών.

Post-hoc testing

Για την διάκριση των σημαντικά διαφορετικών μεταβλητών γίνεται test σύγκρισης μέσων όρων με την βιβλιοθήκη agricolae

```
library(agricolae)
```

Για least significant difference test(LSD) χρησιμοποιείται η εντολή LSD.test(model, 'Treatment ',p.adj= 'bonferroni ') η οποία με βάση των δεδομένων του παραδείγματος γίνεται :

```
out1= LSD.test(analysis, 'Ποικιλία ', p.adj= 'bonferroni ')
out1
```

```
> out1 = LSD.test(analysis,"Ποικιλία", p.adj="bonferroni")
> out1
$statistics
  MSerror Df     Mean      CV  t.value      MSD
  21597.4  6 1335.312 11.0057 3.862991 401.4296

$parameters
        test  p.ajusted   name.t ntr alpha
  Fisher-LSD bonferroni Ποικιλία   4  0.05

$means
  Απόδοση      std r       LCL     UCL  Min  Max    Q25    Q50     Q75
A 1463.75 238.6900 4 1283.9503 1643.55 1185 1670 1305.0 1500.0 1658.75
B 1471.25 209.5382 4 1291.4503 1651.05 1290 1665 1290.0 1465.0 1646.25
C 1067.50 442.6153 4  887.7003 1247.30  660 1475  697.5 1067.5 1437.50
D 1338.75 179.5538 4 1158.9503 1518.55 1180 1565 1202.5 1305.0 1441.25

$comparison
NULL

$groups
  Απόδοση groups
B 1471.25      a
A 1463.75     ab
D 1338.75     ab
C 1067.50      b
```

(Εικόνα 6.13) Αποτελέσματα least significant difference test.

Από το αποτέλεσμα φαίνεται πως υπάρχει στατιστικά σημαντική διαφορά μόνο μεταξύ των ποικιλιών B και C με την ποικιλία B να είναι πιο παραγωγική. Δεν υπάρχει στατιστικά σημαντική διαφορά μεταξύ όλων των άλλων ποικιλιών , αφού στη στήλη groups του LSD test μοιράζονται τα ίδια γράμματα. Έτσι οι ποικιλίες A,D δεν διαφέρουν σημαντικά από τις ποικιλίες B,C παρά το γεγονός ότι οι ποικιλίες B,C διαφέρουν σημαντικά μεταξύ τους.

Για την διεξαγωγή HSD test μπορεί να χρησιμοποιηθεί η βιβλιοθήκη stats.

```
library(stats)
```

Με την βοήθεια της βιβλιοθήκης stats η διεξαγωγή HSD test γίνεται με την εντολή:

```
TukeyHSD(model)
```

Στην συγκεκριμένη περίπτωση ως model έχει τεθεί η μεταβλητή analysis, συνεπώς η εντολή γράφεται ως:

```
TukeyHSD(analysis)
```

```
> TukeyHSD(aov(Απόδοση ~ Ποικιλία+Στήλη+Σειρά,df))
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = Απόδοση ~ Ποικιλία + Στήλη + Σειρά, data = df)

$Ποικιλία
       diff        lwr       upr     p adj
B-A    7.50 -352.22991 367.22991 0.9998493
C-A -396.25 -755.97991 -36.52009 0.0335594
D-A -125.00 -484.72991 234.72991 0.6471198
C-B -403.75 -763.47991 -44.02009 0.0309755
D-B -132.50 -492.22991 227.22991 0.6082691
D-C  271.25  -88.47991 630.97991 0.1375327
```

(Εικόνα 6.14) Σύγκριση μέσων όρων με την μέθοδο Tukey.

Από τα αποτελέσματα που προκύπτουν εξετάζεται μόνο η σύγκριση μεταξύ ποικιλιών. Στατιστικά σημαντική διαφορά φαίνεται να υπάρχει μεταξύ των ποικιλιών C και A αλλά και μεταξύ C και B ($p < 0.05$) όπου και στις δύο περιπτώσεις η ποικιλία C είναι η λιγότερο παραγωγική.Δεν εμφανίζεται στατιστικά σημαντική διαφορά μεταξύ των υπόλοιπων συγκρίσεων. Στην περίπτωση αυτή γίνεται υπολογισμός του συντελεστή διακύμανσης των ποικιλιών με σκοπό την μεταξύ τους σύγκριση. Για τον υπολογισμό του συντελεστή διακύμανσης χρησιμοποιείται η βιβλιοθήκη dplyr.

```
library(dplyr)
```

Ο υπολογισμός των συντελεστών διακύμανσης γίνεται με τον εξής τρόπο:

```
for (i in unique(df$Ποικιλία)){
  x = paste0("CV",i,"=", sd(filter(df, Ποικιλία == i)$`Απόδοση`) / mean(filter(df, Ποικιλία ==
i)$`Απόδοση`))
  print(x)
}
```

```
> for (i in unique(df$Ποικιλία)){
+   x = paste0("CV",i,"=", sd(filter(df, Ποικιλία == i)$`Απόδοση`) / mean(filter(df, Ποικιλία == i)$`Απόδοση`))
+   print(x)
+ }
[1] "CVB=0.142421874377572"
[1] "CVD=0.134120519305417"
[1] "CVC=0.414627946949864"
[1] "CVA=0.163067464517189"
```

(Εικόνα 6.15) Υπολογισμός συντελεστών διακύμανσης

Όπου CVA, CVB, CVC και CVD είναι οι συντελεστές διακύμανσης των ποικιλιών A, B, C και D αντίστοιχα. Από την μεταξύ τους σύγκριση φαίνεται πως η ποικιλία D έχει μικρότερο συντελεστή διακύμανσης. Έτσι επιλέγουμε την ποικιλία D.

Σύνοψη κώδικα

```
library(readxl)
library(stats)
library(agricolae)
df=read_excel('latinsq.xlsx')
Sys.setlocale(category = "LC_ALL", locale = "Greek")
df$Σειρά = as.factor(df$Σειρά)
df$Στήλη = as.factor(df$Στήλη)
df$Ποικιλία= as.factor(df$Ποικιλία)
analysis = aov(Απόδοση ~ Σειρά+Στήλη+Ποικιλία,df)
anova(analysis)
out1 = LSD.test(analysis,"Ποικιλία", p.adj="bonferroni")
out1
TukeyHSD(analysis)
library(dplyr)
for (i in unique(df$Ποικιλία)){
  x = paste0("CV",i,"=", sd(filter(df, Ποικιλία == i)$`Απόδοση`) / mean(filter(df, Ποικιλία ==
i)$`Απόδοση`))
  print(x)
}
```

6.5 Λατινικό τετράγωνο Python

Η ανάλυση των δεδομένων που προκύπτουν από ένα λατινικό τετράγωνο γίνεται με την βοήθεια των βιβλιοθηκών pandas και statsmodels

```
import pandas as pd
from  statsmodels.formula.api import ols
from statsmodels.stats.anova import anova_lm
```

Αρχικά γίνεται εισαγωγή των δεδομένων μέσα από την εντολή :

```
data = pd.read_excel('filename.xlsx')
```

Με αυτόν τον τρόπο η μεταβλητή data αποτελεί το πλαίσιο δεδομένων. Μετά την εισαγωγή του πλαισίου δεδομένων όπως αυτό της εικόνας 6.5 ακολουθεί διαμόρφωση των δεδομένων έτσι ώστε να είναι κατάλληλα για τα στατιστικά μοντέλα.

```
data = pd.DataFrame(data, columns=[όνομα πρώτης στήλης,όνομα δεύτερης στήλης,όνομα τρίτης στήλης,όνομα τέταρτης στήλης])
```

Για το παράδειγμα της εικόνας 6.5 αυτό θα αντιστοιχούσε σε:

```
data = pd.DataFrame(data, columns=['Στήλη','Σειρά','Ποικιλία','Απόδοση'])
```

Ο σχηματισμός του πίνακα Anova γίνεται με την εντολή

```
formula = 'Απόδοση ~ C(Στήλη)+C(Σειρά)+C(Ποικιλία)'
lm = ols (formula,data).fit()
print(anova_lm(lm))
```

```
Name: Ποικιλία, dtype: object
               df       sum_sq        mean_sq          F    PR(>F)
C(Στήλη)      3.0  827342.1875  275780.729167  12.769166  0.005148
C(Σειρά)      3.0   30154.6875   10051.562500   0.465406  0.716972
C(Ποικιλία)   3.0  426842.1875  142280.729167   6.587865  0.025092
Residual      6.0  129584.3750   21597.395833        NaN       NaN
```

(Εικόνα 6.16) Πίνακας Two-Way Anova.

Από τον πίνακα Anova φαίνεται ότι υπάρχει σημαντική διαφοροποίηση στην απόδοση σε κάθε στήλη άλλα και εξαιτίας των διαφορετικών ποικιλιών.

Post-hoc testing

Για την διάκριση των σημαντικά διαφορετικών ποικιλιών γίνεται test σύγκρισης μέσων όρων HSD με την βιβλιοθήκη bioinfokit.

```
from bioinfokit.analys import stat
res = stat()
res.tukey_hsd(data,'Απόδοση','Ποικιλία', anova_model=formula)
print(res.tukey_summary)
```

	group1	group2	Diff	Lower	Upper	q-value	p-value
0	B	D	132.50	-227.278755	492.278755	1.803205	0.603394
1	B	C	403.75	43.971245	763.528755	5.494673	0.030986
2	B	A	7.50	-352.278755	367.278755	0.102068	0.900000
3	D	C	271.25	-88.528755	631.028755	3.691467	0.137556
4	D	A	125.00	-234.778755	484.778755	1.701137	0.639197
5	C	A	396.25	36.471245	756.028755	5.392605	0.033569

(Εικόνα 6.17) Σύγκριση μέσων όρων με την μέθοδο Tukey.

Στατιστικά σημαντική διαφορά φαίνεται να υπάρχει μεταξύ των ποικιλιών C και A αλλά και μεταξύ C και B ($p < 0.05$) όπου και στις δύο περιπτώσεις η ποικιλία C είναι η λιγότερο παραγωγική. Δεν εμφανίζεται στατιστικά σημαντική διαφορά μεταξύ των υπόλοιπων συγκρίσεων. Στην περίπτωση αυτή γίνεται υπολογισμός του συντελεστή διακύμανσης των ποικιλιών με σκοπό την μεταξύ τους σύγκριση.Για τον υπολογισμό του συντελεστή διακύμανσης χρησιμοποιείται η βιβλιοθήκη statistics και numpy.

```
import statistics
import numpy as np
```

Ο υπολογισμός των συντελεστών διακύμανσης γίνεται με τον εξής τρόπο:

```
for i in np.unique(data.Ποικιλία):
    x = data.query('Ποικιλία == "{}"'.format(i))['Απόδοση']
    print("CV{}={}".format(i,statistics.stdev(x)/statistics.mean(x)))
```

```
CVA=0.16306746451718857
CVB=0.14242187437757187
CVC=0.4146279469498642
CVD=0.13412051930541702
```

(Εικόνα 6.18) Υπολογισμός συντελεστών διακύμανσης

Όπου CVA, CVB, CVC και CVD είναι οι συντελεστές διακύμανσης των ποικιλιών A, B, C και D αντίστοιχα.

Σύνοψη κώδικα

```
import pandas as pd
from  statsmodels.formula.api import ols
from statsmodels.stats.anova import anova_lm
data = pd.read_excel('latinsq.xlsx')
data = pd.DataFrame(data, columns=['Στήλη', 'Σειρά', 'Ποικιλία', 'Απόδοση'])
formula = 'Απόδοση ~ C(Στήλη)+C(Σειρά)+C(Ποικιλία)'
lm = ols (formula,data).fit()
print(anova_lm(lm))
from bioinfokit.analys import stat
res = stat()
res.tukey_hsd(data,'Απόδοση','Ποικιλία', anova_model=formula)
print(res.tukey_summary)
import statistics
import numpy as np
for i in np.unique(data.Ποικιλία):
    x = data.query('Ποικιλία == "{}"'.format(i))['Απόδοση']
    print("CV{}={}".format(i,statistics.stdev(x)/statistics.mean(x)))
```

7 Split-plot design

Ο σχεδιασμός split-plot είναι ένας πειραματικός σχεδιασμός ο οποίος αποτελείται από δύο πειράματα με διαφορετικές πειραματικές μονάδες διαφορετικού μεγέθους[13]. Κυρίως χρησιμοποιείται όταν ένας υπό εξέταση παράγοντας είναι πιο δύσκολο να τυχαιοποιηθεί σε σχέση με άλλους παράγοντες.[14] Μπορεί ακόμα να χρησιμοποιηθεί όταν είναι αναγκαία η μεγαλύτερης ακρίβειας σύγκριση ενός παράγοντα από΄ ό,τι η σύγκριση άλλων παραγόντων ή εάν είναι αναγκαία η εισαγωγή νέων παραγόντων σε ένα υπό εξέλιξη πείραμα[15]. Σε ένα πείραμα με 4 επίπεδα εφαρμογής αζώτου(N1,N2,N3,N4), 3 διαφορετικές ποικιλίες καλαμποκιού(V1,V2,V3) και 3 διαφορετικά χωράφια, εξετάζεται η απόδοση της κάθε ποικιλίας. Ο σχεδιασμός σε split-plot του παραπάνω πειράματος θα γινόταν ως εξής:

Χωράφι 1	Χωράφι 2	Χωράφι 3

Κάθε ένα από τα τρία χωράφια αποτελεί ένα block (whole-plot) του πειράματος. Αρχικά λόγο του ότι η λίπανση μπορεί να εφαρμοστεί σε μικρές πειραματικές μονάδες, το κάθε χωράφι θα χωριστεί σε 3 ίσα μέρη λόγο του ό,τι εξετάζονται 3 διαφορετικές ποικιλίες .Οι ποικιλίες τοποθετούνται τυχαία σε κάθε whole-plot όπως φαίνεται παρακάτω και αποτελούν τον split-plot παράγοντα.

V2	V3	V1	V2	V1	V3	V3	V2	V1
(Χωράφι 1)			(Χωράφι 2)			(Χωράφι 3)		

Στην συνέχεια τα split-plots διαιρούνται σε τόσα μέρη (sub-plots)όσα και τα επίπεδα αζώτου. Το άζωτο αποτελεί τον sub-plot παράγοντα μιας και ο διαχωρισμός σε sub-plots έγινε βάση των επιπέδων του. Τα επίπεδα αζώτου ανατίθενται τυχαία σε κάθε sub-plot όπως φαίνεται παρακάτω.

V2 N1	V3 N4	V1 N2	V2 N4	V1 N2	V3 N2	V3 N2	V2 N1	V1 N3
V2 N4	V3 N1	V1 N1	V2 N2	V1 N1	V3 N1	V3 N1	V2 N2	V1 N2
V2 N2	V3 N3	V1 N4	V2 N3	V1 N3	V3 N4	V3 N3	V2 N4	V1 N1
V2 N3	V3 N2	V1 N3	V2 N1	V1 N4	V3 N3	V3 N4	V2 N3	V1 N4
(Χωράφι 1)			(Χωράφι 2)			(Χωράφι 3)		

Ουσιαστικά πραγματοποιούνται δύο διαφορετικά πειράματα με το κάθε πείραμα να έχει την δικιά του τυχαιοποίηση και πειραματικές μονάδες. Για τον λόγο αυτό αναμένονται δύο ξεχωριστά πειραματικά σφάλματα[16] .Τα δεδομένα του παραδείγματος φαίνονται στην παρακάτω εικόνα. Ο δύσκολος παράγοντας (ποικιλία) έχει 3 επαναλήψεις, ενώ ο εύκολος (επίπεδο αζώτου) 9. Με τον τρόπο αυτό διασφαλίζεται καλύτερη ακρίβεια στον παράγοντα άζωτο.

variety	Nrate	block	Yield
V1	N2	1	92
V1	N1	1	93
V1	N4	1	72
V1	N3	1	89
V1	N2	2	88
V1	N1	2	91
V1	N3	2	96
V1	N4	2	76
V1	N3	3	90
V1	N2	3	85
V1	N1	3	86
V1	N4	3	72
V2	N1	1	70
V2	N4	1	61
V2	N2	1	75
V2	N3	1	86
V2	N4	2	60
V2	N2	2	61
V2	N3	2	70
V2	N1	2	72
V2	N1	3	62
V2	N2	3	63
V2	N4	3	59
V2	N3	3	70
V3	N4	1	58
V3	N1	1	69
V3	N3	1	69
V3	N2	1	82
V3	N2	2	65
V3	N1	2	68
V3	N4	2	58
V3	N3	2	86
V3	N2	3	75
V3	N1	3	77
V3	N3	3	80
V3	N4	3	60

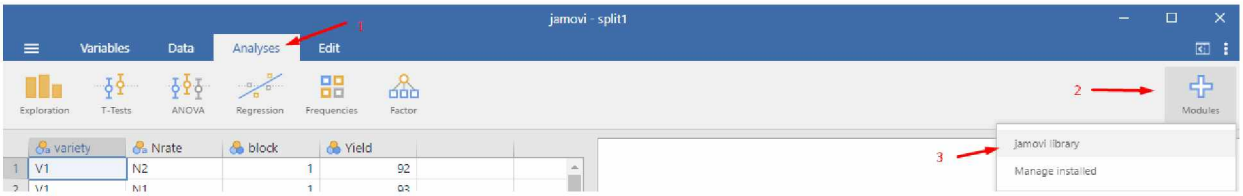
(Εικόνα 7.1) Πλαίσιο δεδομένων split-plot πειράματος.

7.1 Ανάλυση Δεδομένων split-plot design

Το σχέδιο split-plot αναλύεται με τη χρήση Ανάλυσης Διακύμανσης (ANOVA).Παρακάτω περιγράφεται ο τρόπος με τον οποίο αυτή η ανάλυση μπορεί να γίνει μέσω του στατιστικού λογισμικού Jamovi, της γλώσσας προγραμματισμού R αλλά και της γλώσσας Python. Ως παράδειγμα θα χρησιμοποιηθεί το πλαίσιο δεδομένων της εικόνας 7.1.

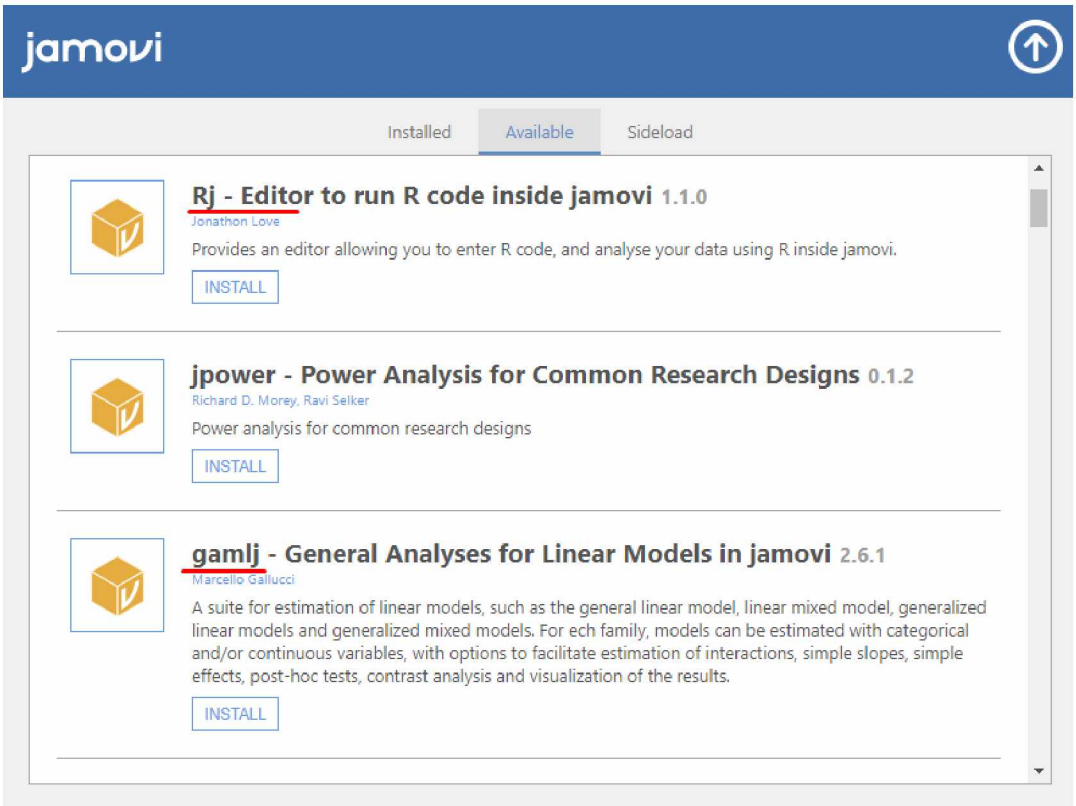
7.2 Split-plot Jamovi:

Με την εισαγωγή του αρχείου excel στο jamovi η ανάλυση split-plot σχεδιασμού απαιτεί την χρήση δύο πρόσθετων πακέτων από την βιβλιοθήκη του jamovi. Η προσθήκη των πακέτων γίνεται από την καρτέλα "Analyses" κάνοντας click στο "Modules" στο πάνω μέρος του παραθύρου και στην συνέχεια "jamovi library".



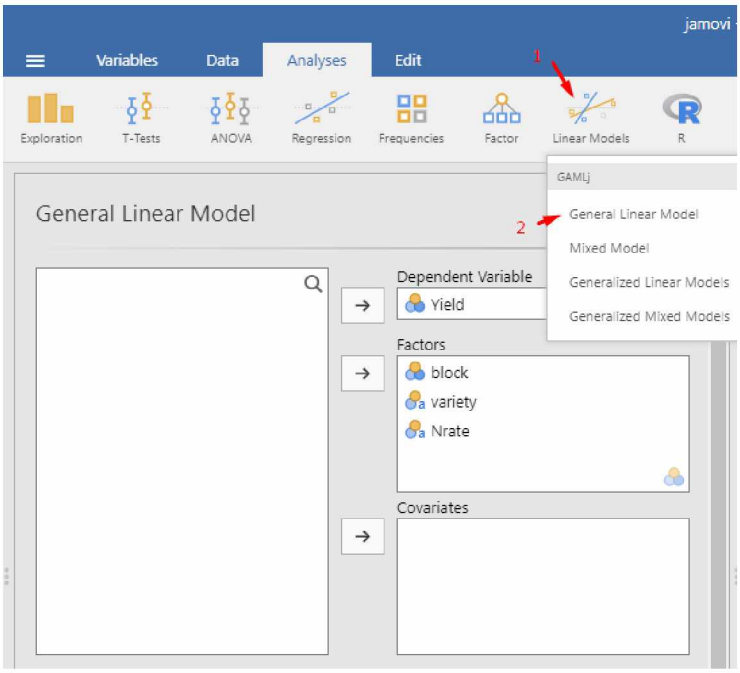
(Εικόνα 7.2)Εισαγωγή στην βιβλιοθήκη Jamovi.

Τα πακέτα που θα εγκατασταθούν είναι το "Rj Editor" και το "gamlj". Η εγκατάσταση τους γίνεται πατώντας στο κουμπί "INSTALL" κάτω από κάθε πακέτο.



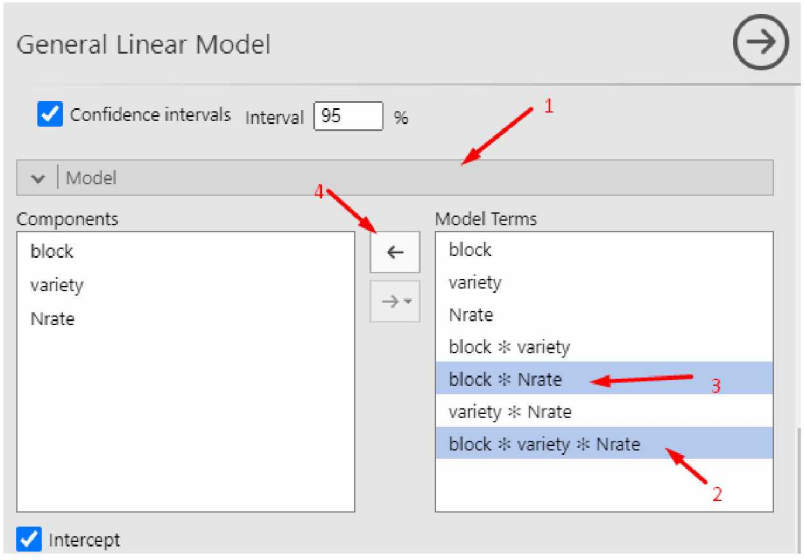
(Εικόνα 7.3)Εγκατάσταση πακέτων.

Μετά την εγκατάσταση των δύο πακέτων, για την ανάλυση split-plot, από την καρτέλα "Analyses" επιλέγεται το "Linear models" και στην συνέχεια το "General Linear Model". Ως dependent Variable χρησιμοποιείται η μεταβλητή απόκρισης ενώ ως factors η μεταβλητή block, η μεταβλητή του whole-plot παράγοντα, και η μεταβλητή του split-plot παράγοντα με αυτήν την σειρά.



(Εικόνα 7.4) Επιλογή όρων για ανάλυση.

Στην συνέχεια αφαιρούνται από τα model terms τα block* sub-plot factor*split-plot factor, block*sub-plot factor και η τριπλή αλληλεπίδραση block*variety*Nrateόπως φαίνεται παρακάτω.



(Εικόνα 7.5) Αφαίρεση όρων από τα Model terms.

Στα δεξιά σχηματίζεται ο πίνακας Anova. Στον πίνακα Anova ο όρος “block * variety” αποτελεί το Error a της ανάλυσης ενώ ο όρος “Residuals” αποτελεί το Error b της ανάλυσης.

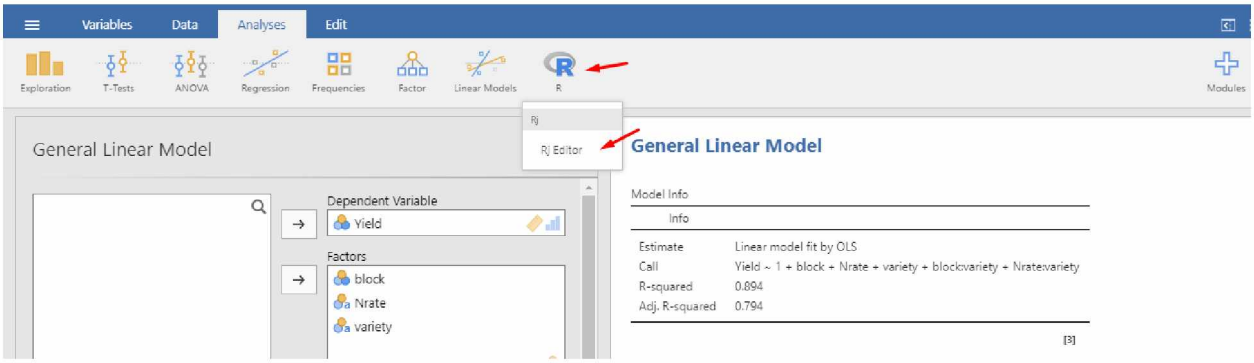
Model Results

ANOVA Omnibus tests

	SS	df	F	p
Model	4244.7	17	8.920	< .001
block	59.4	2	1.061	0.367
Nrate	1529.2	3	18.211	< .001
variety	2327.1	2	41.568	< .001
block ✻ variety	216.1	4	1.930	0.149
Nrate ✻ variety	112.9	6	0.673	0.673
Residuals	503.8	18		
Total	4748.6	35		

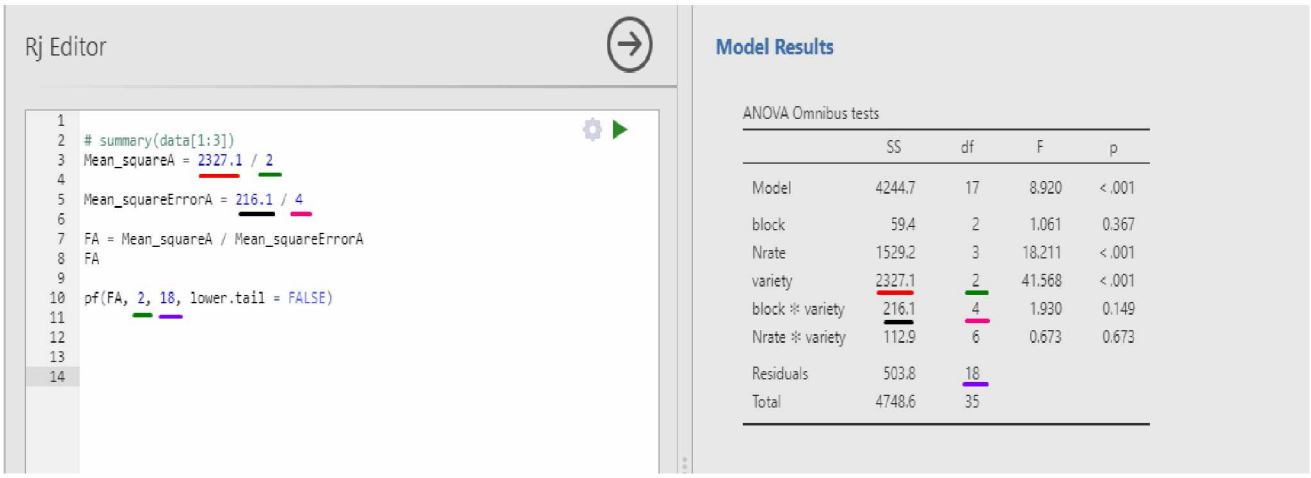
(Εικόνα 7.6) Πίνακας Anova.

Ακόμα λόγο περιορισμών του jamovi ο υπολογισμός της τιμής F και p για τον split-plot factor(variety) είναι λάθος και δεν πρέπει να ληφθούν υπόψιν. Για τον σωστό υπολογισμό των δύο αυτών τιμών, γίνεται χρήση του Rj Editor.



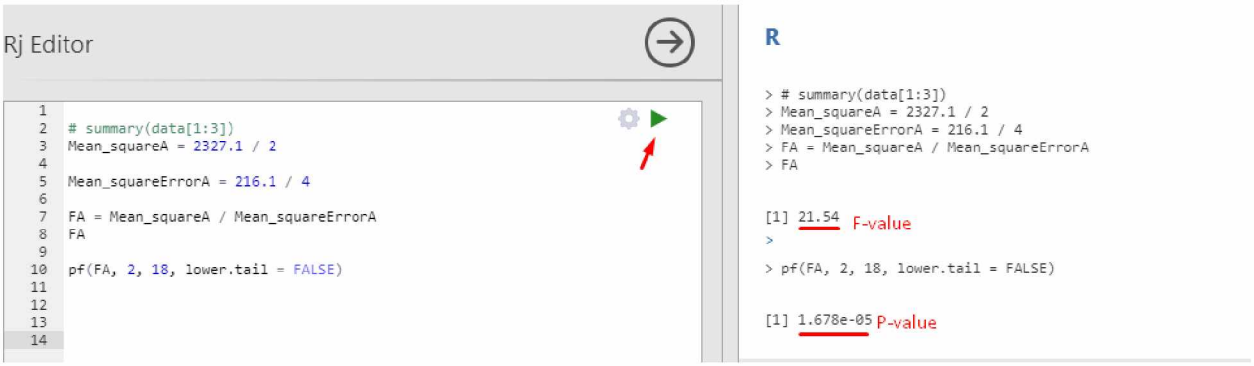
(Εικόνα 7.7)Επιλογή Rj editor.

Για τον υπολογισμό της τιμής F αρχικά υπολογίζεται το Mean square για τον split-plot παράγοντα και για το Error A.Το mean square υπολογίζεται διαιρώντας το άθροισμα τετραγώνων με τον βαθμό ελευθερίας.Η τιμή F υπολογίζεται μέσα από την διαίρεση του Mean square του split-plot παράγοντα με το Mean square του Error A[17].Τέλος ο υπολογισμός της τιμής p γίνεται μέσω του Rj Editor γράφοντας “pf(τιμή F, βαθμός ελευθερίας του παράγοντα, βαθμός ελευθερίας του Error, lower.tail = FALSE)”.



(Εικόνα 7.8)Επεξεργαστής κειμένου Rj Editor.

Για την εμφάνιση των αποτελεσμάτων αρκεί να γίνει κλικ στο πράσινο βελάκι.

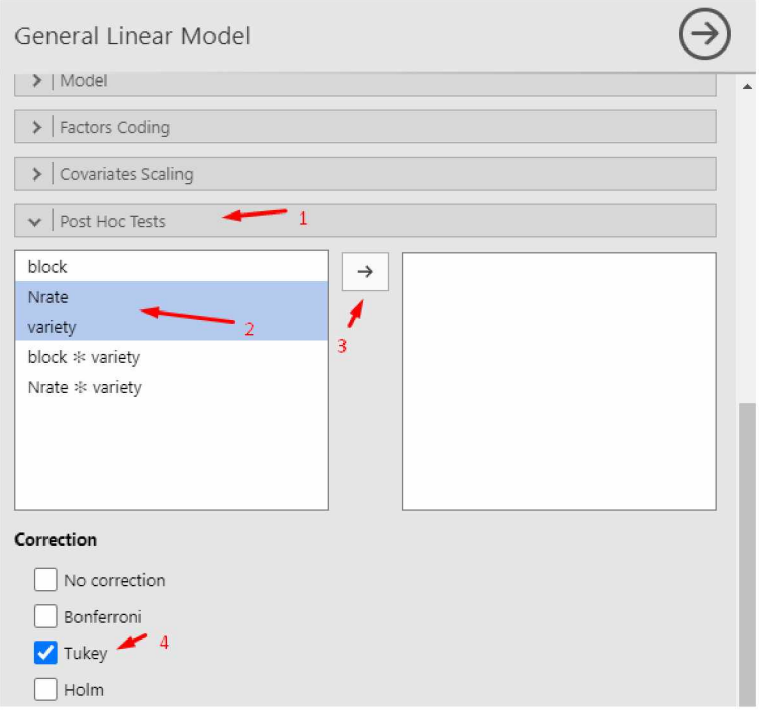


(Εικόνα 7.9)Υπολογισμός F και p value.

Από τα αποτελέσματα παρατηρείται ότι υπάρχει σημαντική διαφορά για την ποικιλία και τον βαθμό αζώτου ενώ οι αλληλεπίδραση των δύο δεν είναι σημαντική.

Post-hoc testing

Για την διάκριση των σημαντικά διαφορετικών μεταβλητών γίνεται test σύγκρισης μέσων όρων με την μέθοδο Tukey[4].Από τα Post Hoc Tests επιλέγονται οι μεταβλητές οι οποίες έχουν σημαντική διαφορά και επιλέγεται η μέθοδος Tukey.



(Εικόνα 7.10) Διεξαγωγή Tukey test σε Jamovi.

Post Hoc Tests

Post Hoc Comparisons - variety

Comparison			Mean Difference	SE	df	t	p_{tukey}
variety		variety					
V1	-	V2	18.42	2.34	22.0	7.89	< .001
	-	V3	15.25	2.34	22.0	6.53	< .001
V2	-	V3	−3.17	2.34	22.0	−1.36	0.381

Note. Comparisons are based on estimated marginal means

Post Hoc Comparisons - Nrate

Comparison			Mean Difference	SE	df	t	p_{tukey}
Nrate		Nrate					
N1	-	N2	0.222	2.70	22.0	0.0824	1.000
	-	N3	−5.333	2.70	22.0	−1.9777	0.227
	-	N4	12.444	2.70	22.0	4.6147	< .001
N2	-	N3	−5.556	2.70	22.0	−2.0601	0.197
	-	N4	12.222	2.70	22.0	4.5323	< .001
N3	-	N4	17.778	2.70	22.0	6.5924	< .001

Note. Comparisons are based on estimated marginal means

(Εικόνα 7.11) Σύγκριση μέσων όρων με την μέθοδο Tukey.

Από το αποτέλεσμα φαίνεται πως η ποικιλία V1 είναι πιο παραγωγική από τις άλλες 2(V1 > V3>V2) ενώ με τον βαθμό εφαρμογής αζώτου N3 επιτυγχάνονται τα καλύτερα αποτελέσματα.

7.3 Split-plot R:

Η ανάλυση split-plot σχεδιασμού στην R μπορεί να γίνει με χρήση της βιβλιοθήκης agricolae.

```
library(agricolae)
```

Αρχικά γίνεται η εισαγωγή δεδομένων μέσα από ένα αρχείο excel.Αυτό προϋποθέτει την χρήση της βιβλιοθήκης readxl.

```
library(readxl)
```

μέσα από την εντολή :

```
df=read_excel('filename.xlsx')
```

Με αυτόν τον τρόπο η μεταβλητή df αποτελεί το σύνολο των δεδομένων στο οποίο θα γίνει η στατιστική ανάλυση. Με χρήση της εντολής head(df) προβάλλονται οι πρώτες 6 γραμμές των δεδομένων.

```
> head(df)
# A tibble: 6 x 4
  variety Nrate block Yield
  <fct>   <fct> <dbl> <dbl>
1 V1      N2        1    92
2 V1      N1        1    93
3 V1      N4        1    72
4 V1      N3        1    89
5 V1      N2        2    88
6 V1      N1        2    91
>
```

(Εικόνα 7.12) Έλεγχος σωστής εισαγωγής δεδομένων.

Μετά την εισαγωγή του πλαισίου δεδομένων όπως αυτό της εικόνας 7.1 μπορεί να γίνει χρήση της εντολής str(df) για τον έλεγχο της μορφής κάθε στήλης του πλαισίου δεδομένων.

```
str(df)
```

```
> str(df)
tibble [36 x 4] (S3: tbl_df/tbl/data.frame)
 $ variety: chr [1:36] "V1" "V1" "V1" "V1" ...
 $ Nrate  : chr [1:36] "N2" "N1" "N4" "N3" ...
 $ block  : num [1:36] 1 1 1 1 2 2 2 2 3 3 ...
 $ Yield  : num [1:36] 92 93 72 89 88 91 96 76 90 85 ...
```

(Εικόνα 7.13) Έλεγχος μορφής εισακτέων δεδομένων.

Σε αυτό το παράδειγμα οι ποικιλίες και το Nrate διαβάζονται ως χαρακτήρες(chr).Θα πρέπει αρχικά να γίνει η μετατροπή τους σε factors με την εντολή as.factor(filename$variabel) ή πιο συγκεκριμένα για το παρών παράδειγμα:

df$variety = as.factor(df$variety)
df$Nrate = as.factor(df$Nrate)
Για την ανάλυση διακύμανσης ενός split-plot σχεδιασμού χρησιμοποιείται το εξής μοντέλο.

model = with(df, sp.plot(block, split-plot factor, sub-plot factor, μεταβλητή απόκρισης))
Όπως προαναφέρθηκε για το συγκεκριμένο παράδειγμα , whole-plot factor αποτελούν οι ποικιλίες, split-plot factor αποτελεί το επίπεδο εφαρμογής αζώτου ενώ η απόδοση είναι η μεταβλητή απόκρισης. Συνεπώς το παραπάνω μοντέλο γράφεται ώς:

model = with(df, sp.plot(block, variety, Nrate, Yield))

Με αυτόν τον τρόπο δημιουργείται η ανάλυση split-plot.

```
ANALYSIS SPLIT PLOT:  Yield
Class level information

variety         :  V1 V2 V3
Nrate   :  N2 N1 N4 N3
block   :  1 2 3

Number of observations:  36

Analysis of Variance Table

Response: Yield
              Df  Sum Sq Mean Sq F value    Pr(>F)
block          2   59.39   29.69     NaN       NaN
variety        2 2327.06 1163.53 21.5357  0.007221 **
Ea             4  216.11   54.03     NaN       NaN
Nrate          3 1529.22  509.74 18.2110 1.095e-05 ***
variety:Nrate  6  112.94   18.82  0.6725  0.673303
Eb            18  503.83   27.99     NaN       NaN
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

cv(a) = 9.9 %, cv(b) = 7.1 %, Mean = 74.61111
```

(Εικόνα 7.14)Πίνακας Anova.

Από τον πίνακα Anova παρατηρείται ότι υπάρχει σημαντική διαφορά για την ποικιλία και τον βαθμό αζώτου ενώ οι αλληλεπίδραση των δύο δεν είναι σημαντική.

Post-hoc testing

Για την διάκριση των σημαντικά διαφορετικών μεταβλητών γίνεται test σύγκρισης μέσων όρων. Για την σύγκριση μέσων όρων του split-plot παράγοντα(variety) θα χρησιμοποιηθεί ο βαθμός ελευθερίας και το τετράγωνο του μέσου όρου του Ea.Για τον sub-plot παράγοντα και για την αλληλεπίδραση του split-plot παράγοντα με τον sub-plot παράγοντα θα χρησιμοποιηθούν ο βαθμός ελευθερίας και το τετράγωνο του μέσου όρου του Eb[18].Αυτό μπορεί να διευκρινιστεί με τις παρακάτω εντολές:

```
gla = model$gl.a
glb = model$gl.b
Ea = model$Ea
Eb = model$Eb
```

Για την διεξαγωγή του τεστ σύγκρισης μέσων όρων του whole-plot παράγοντα γίνεται χρήση της εντολής:

```
out1 = with(df,LSD.test(μεταβλητή απόκρισης,whole-plot παράγοντας,gla, Ea, console = TRUE))
```

Ή για το συγκεκριμένο παράδειγμα

```
out1 = with(df,LSD.test(Yield,variety,gla, Ea, console = TRUE))
```

```
Study: Yield ~ variety

LSD t Test for Yield

Mean Square Error:  54.02778

variety,  means and individual ( 95 %) CI

      Yield      std  r      LCL      UCL Min Max
V1 85.83333 8.155682 12 79.94209 91.72458  72  96
V2 67.41667 7.982462 12 61.52542 73.30791  59  86
V3 70.58333 9.481641 12 64.69209 76.47458  58  86

Alpha: 0.05 ; DF Error: 4
Critical Value of t: 2.776445

least Significant Difference: 8.331477

Treatments with the same letter are not significantly different.

      Yield groups
V1 85.83333      a
V3 70.58333      b
V2 67.41667      b
```

(Εικόνα 7.15) least significant difference test για ποικιλίες.

Από το αποτέλεσμα φαίνεται πως η ποικιλία V1 είναι πιο παραγωγική από τις άλλες 2. Δεν υπάρχει στατιστικά σημαντική διαφορά μεταξύ των V2,V3, αφού στη στήλη groups του LSD test μοιράζονται το ίδιο γράμμα (b).

Για την διεξαγωγή του τεστ σύγκρισης μέσων όρων του split-plot παράγοντα γίνεται χρήση της εντολής:

```
out2 = with(df,LSD.test(μεταβλητή απόκρισης,sub-plot παράγοντας,glb, Eb, console = TRUE))
```

Ή για το συγκεκριμένο παράδειγμα

```
out2 = with(df,LSD.test(Yield,Nrate,glb, Eb, console = TRUE))
```

```
      Yield       std r      LCL      UCL Min Max
N1 76.44444 11.035297 9 72.73938 80.14951  62  93
N2 76.22222 11.366373 9 72.51716 79.92729  61  92
N3 81.77778 10.009718 9 78.07271 85.48284  69  96
N4 64.00000  7.158911 9 60.29493 67.70507  58  76

Alpha: 0.05 ; DF Error: 18
Critical Value of t: 2.100922

least Significant Difference: 5.239754

Treatments with the same letter are not significantly different.

      Yield groups
N3 81.77778      a
N1 76.44444      b
N2 76.22222      b
N4 64.00000      c
```

(Εικόνα 7.16) least significant difference test για τα επίπεδα αζώτου.

Από το αποτέλεσμα φαίνεται πως η μεγαλύτερη απόδοση επιτυγχάνεται με τον βαθμό εφαρμογής αζώτου N3.

Σύνοψη κώδικα

```
library(agricolae)
library(readxl)
df=read_excel('splitplot.xlsx')
head(df)
str(df)
df$variety = as.factor(df$variety)
df$Nrate = as.factor(df$Nrate)
model = with(df, sp.plot(block, variety, Nrate, Yield))
gla = model$gl.a
glb = model$gl.b
Ea = model$Ea
Eb = model$Eb
out1 = with(df,LSD.test(Yield,variety,gla, Ea, console = TRUE))
out2 = with(df,LSD.test(Yield,Nrate,glb, Eb, console = TRUE))
```

7.4 Split-plot Python:

Η ανάλυση των δεδομένων που προκύπτουν από έναν πειραματικό σχεδιασμό split-plot γίνεται με την βοήθεια των βιβλιοθηκών pandas,numpy,math και scipy.

```
import pandas as pd
import numpy as np
import math
import scipy.stats
from scipy.stats import studentized_range
```

Αρχικά γίνεται εισαγωγή των δεδομένων μέσα από την εντολή :

```
data = pd.read_excel('filename.xlsx')
```

Με αυτόν τον τρόπο η μεταβλητή data αποτελεί το πλαίσιο δεδομένων. Μετά την εισαγωγή του πλαισίου δεδομένων όπως αυτό της εικόνας 7.1 ακολουθεί διαμόρφωση των δεδομένων έτσι ώστε να είναι κατάλληλα για τα στατιστικά μοντέλα.

```
data = pd.DataFrame(data, columns=[όνομα πρώτης στήλης,όνομα δεύτερης στήλης,όνομα τρίτης στήλης])
```

Για το παράδειγμα της εικόνας 7.1 αυτό θα αντιστοιχούσε σε:

```
data = pd.DataFrame(data, columns=['variety','Nrate','block','Yield'])
```

Δυστυχώς στην Python δεν προσφέρεται κάποια βιβλιοθήκη για τον σχηματισμό του πίνακα Anova.

Αντ'αυτού οι υπολογισμοί των δεδομένων για τον σχηματισμό του πίνακα Anova μπορούν να γίνουν με τον εξής τρόπο:

Υπολογισμός του Grand Total της μεταβλητής απόκρισης μπορεί να γίνει με την παρακάτω εντολή:

```
GT = sum(data.Yield)
```

Όπου "Yield" η μεταβλητή απόκρισης.

Στην συνέχεια είναι απαραίτητο να σχηματιστούν μεταβλητές με τα επίπεδα κάθε μεταβλητής.

```
#Levels of Factor A
```

```
v = len(np.unique(data.variety))
#Levels of Factor B
N = len(np.unique(data.Nrate))
#Levels of Blocks
b = len(np.unique(data.block))
```

Με αυτόν τον τρόπο η μεταβλητή v εμπεριέχει τον αριθμό 3 δηλαδή όσα και τα επίπεδα των ποικιλιών. Παρομοίως οι μεταβλητές N και b αντιπροσωπεύουν των αριθμό των επιπέδων του αζώτου και των block.

Με αυτά τα δεδομένα μπορεί να γίνει ο υπολογισμός του Correction Factor:

```
CF = (GT**2)/(v*N*b)
```

```
In [26]: CF
Out[26]: 200405.44444444444
```

(Εικόνα 7.17) Correction Factor.

ενώ στην συνέχεια υπολογίζεται το συνολικό άθροισμα των τετραγώνων.

```
TSS = sum(data.Yield**2) - CF
```

```
In [24]: TSS
Out[24]: 4748.555555555562
```

(Εικόνα 7.18) Συνολικό άθροισμα τετραγώνων.

Το επόμενο βήμα είναι ο υπολογισμός του αθροίσματος των τετραγώνων κάθε μεταβλητής ξεχωριστά.

Για τον υπολογισμό του αθροίσματος των τετραγώνων των block αρχικά σχηματίζεται μια κενή λίστα με τρία στοιχεία, όσα δηλαδή και τα επίπεδα των block.Εάν τα block είχαν 4 επίπεδα τότε η λίστα που θα χρειαζόταν θα είχε 4 στοιχεία.

```
rss = [0,0,0]
```

Τα στοιχεία της λίστας θα αντικατασταθούν με το άθροισμα της απόδοσης που αντιστοιχεί σε κάθε block.Για το συγκεκριμένο παράδειγμα η λίστα θα είναι rss = [916, 891, 879] διότι στο block 1 η συνολική απόδοση ήταν 916.Αυτό μπορεί να γίνει αυτόματα με τον παρακάτω κώδικα.

```
rss = [0,0,0]
for y in range(b):
    for x in range(len(data.block)):
                if data.block[x] == y+1:
                    rss[y] = rss[y] + data.Yield[x]
```

Μετά τον σχηματισμό της λίστας με τον παραπάνω τρόπο το άθροισμα τετραγώνων των block υπολογίζεται ως εξής

```
RSS = 0
for i in rss:
    RSS = RSS + i**2
RSS = (RSS /(v*N)) - CF
```

Όπου η μεταβλητή RSS είναι το άθροισμα τετραγώνων των block.

```
In [23]: RSS
Out[23]: 59.38888888890506
```

(Εικόνα 7.19) Άθροισμα τετραγώνων block.

Με παρόμοιο τρόπο μπορούν να υπολογιστούν τα αθροίσματα τετραγώνων για το άζωτο και την ποικιλία.

Άθροισμα τετραγώνων για ποικιλίες:

```
vb= [0,0,0]
for y in range(v):
    for x in range(len(data.Yield)):
                    if data.variety[x] == f'V{y+1}':
                        vb[y] = vb[y] + data.Yield[x]

Variety_SS = 0
for i in vb:
    Variety_SS = Variety_SS + i**2

Variety_SS = (Variety_SS/(N*b)) - CF
```

```
In [27]: Variety_SS
Out[27]: 2327.055555555562
```

(Εικόνα 7.20) Άθροισμα τετραγώνων ποικιλίας.

Επί της ουσίας διαβάζονται όλες οι τιμές της μεταβλητής απόκρισης και εάν η τιμή αυτή αντιστοιχεί σε ποικιλία με το όνομα "V1" τότε προστίθεται στο πρώτο στοιχείο της λίστας vb.Ο διαχωρισμός των τιμών γίνεται με την εντολή `if data.variety[x] == f'V{y+1}':` .Εάν ο κάθε whole-plot παράγοντας είχε διαφορετικό όνομα, για παράδειγμα 1,2,3 αντί για V1,V2,V3, τότε η παραπάνω εντολή θα γινόταν `if data.variety[x] == y+1` και αυτό γιατί η μεταβλητή y θα πάρει τιμές 0,1,2. Η ίδια μεθοδολογία ακολουθείται και για το άθροισμα τετραγώνων για το άζωτο:

```
Nv = [0,0,0,0]
for y in range(N):
    for x in range(len(data.Yield)):
        if data.Nrate[x] == f'N{y+1}':
            Nv[y] = Nv[y] + data.Yield[x]

NRate_SS = 0
for i in Nv:
```

```
    NRate_SS = NRate_SS + i**2
NRate_SS = (NRate_SS/(v*b)) - CF
```

```
In [28]: NRate_SS
Out[28]: 1529.222222222219
```

(Εικόνα 7.21)Άθροισμα τετραγώνων αζώτου.

Στη συνέχεια γίνεται ο υπολογισμός του αθροίσματος τετραγώνων του Error a των ποικιλιών.Εδώ δεν χρειάζεται ο σχηματισμός λίστας αλλά δύο διαφορετικών μεταβλητών "vb" και "SSVB".

```
SSVB = 0
vb = 0
for u in range(v):
    for y in range(v+1):
        SSVB = SSVB + vb**2
        vb = 0
        for x in range(len(data.Yield)):
            if data.variety[x] == f'V{u+1}':
                if data.block[x] == y+1:
                    vb = vb + data.Yield[x]


Ea = (SSVB/N)-CF-Variety_SS-RSS
```

```
In [33]: Ea
Out[33]: 216.11111111109494
```

(Εικόνα 7.22)Άθροισμα τετραγώνων Σφάλματος A

Με παρόμοιο τρόπο μπορεί να υπολογιστεί και το άθροισμα τετραγώνων για την αλληλεπίδραση μεταξύ των δύο παραγόντων(άζωτο και ποικιλία).

```
SSVN = 0
vb = 0

for y in range(N+1):
    for u in range(v):
        SSVN = SSVN + vb**2
        vb = 0
        for x in range(len(data.Yield)):
            if data.Nrate[x] == f'N{y+1}':
                if data.variety[x] == f'V{u+1}':
                    vb = vb + data.Yield[x]

Interaction = (SSVN/b) - CF - Variety_SS - NRate_SS
```

```
In [34]: Interaction
Out[34]: 112.94444444443798
```
(Εικόνα 7.23)Άθροισμα τετραγώνων Αλληλεπίδρασης

Τέλος για το Error B χρησιμοποιείται η εντολή:

```
Eb = TSS - RSS - Variety_SS - Ea - NRate_SS - Interaction
```

```
In [35]: Eb
Out[35]: 503.83333333334303
```
(Εικόνα 7.24)Άθροισμα τετραγώνων Σφάλματος B

Οι βαθμοί ελευθερίας μπορούν εύκολα να υπολογιστούν μιας και είδη έχουν σχηματιστεί μεταβλητές με τα επίπεδα κάθε παράγοντα.

```
# Degrees of Freedom
#Variety
DFv = v - 1
#Nrate
DFN = N - 1
#Block
DFb = b - 1
#Error a
DFEa = DFv*DFb
#Interaction
DFInt = DFv*DFN
#Error b
DFEb = v*DFb*DFN
```

Μετά τον υπολογισμό των βαθμών ελευθερίας, τα μέσα τετράγωνα (Mean_sq) υπολογίζονται με την εξής απλό τρόπο.

```
#Mean_Sq
#Blcok
MSBlock = RSS / DFb
#Variety
MSVar  = Variety_SS / DFv
#Nrate
MSNrate = NRate_SS / DFN
#Error A
MSErrora = Ea / DFEa
#Interaction
```

```
MSInteraction = Interaction / DFInt
#Error B
MSErrorb = Eb / DFEb
```

Τέλος για τον σχηματισμό του πίνακα Anova είναι αναγκαίος ο υπολογισμός των F-values και η εύρεση της κρίσιμης τιμής F. Η F-value υπολογίζεται με τη διαίρεση των μέσων τετραγώνων του παράγοντα με μέσο τετράγωνο του αντίστοιχου λάθους του παράγοντα όπως φαίνεται παρακάτω.

```
#Fvalue
Fvar = MSVar / MSErrora
FNrate = MSNrate / MSErrorb
FInteraction = MSInteraction / MSErrorb
```

Η κρίσιμη τιμή F υπολογίζεται με την εντολή scipy.stats.f.ppf(1-α, df1, df2) όπου df1 είναι οι βαθμοί ελευθερίας του βασικού παράγοντα και df2 οι βαθμοί ελευθερίας του λάθους που αντιστοιχεί στον κάθε παράγοντα.

```
#Fcrit
FAcrit =  scipy.stats.f.ppf(0.95, DFv, DFEa)
FBcrit = scipy.stats.f.ppf(0.95, DFN, DFEb)
Fintcrit = scipy.stats.f.ppf(0.95, DFInt, DFEb)
```

Μετά από όλα τα παραπάνω βήματα έχουν σχηματιστεί οι μεταβλητές που αποτελούν τον πίνακα Anova όπως φαίνεται παρακάτω.

SV	Df	Sum squares	Mean Squares	F value	F value 5%
Block	DFb	RSS	MSBlock		
Variety	DFv	Variety_SS	MSVar	Fvar	FAcrit
Ea	DFEa	Ea	MSNrate		
Nrate	DFN	NRate_SS	MSErrora	FNrate	FBcrit
Variety:Nrate	DFInt	Interaction	MSInteraction	FInteraction	Fintcrit
Eb	DFEb	Eb	MSErrorb		

(Πίνακας 7.1)Πίνακας Anova πριν την αντικατάσταση μεταβλητών.

Αρκεί να γίνει η αντικατάσταση τους.

```
In [41]: DFb,DFv,DFEa,DFN,DFInt,DFEb
Out[41]: (2, 2, 4, 3, 6, 18)

In [42]: RSS,Variety_SS,Ea,NRate_SS,Interaction,Eb
Out[42]:
(59.38888888890506,
 2327.055555555562,
 216.11111111109494,
 1529.222222222219,
 112.94444444443798,
 503.83333333334303)

In [43]: MSBlock,MSVar,MSNrate,MSErrora,MSInteraction,MSErrorb
Out[43]:
(29.69444444445253,
 1163.527777777781,
 509.7407407407397,
 54.027777777773736,
 18.824074074072996,
 27.99074074074128)

In [44]: Fvar,FNrate,FInteraction
Out[44]: (21.53573264781658, 18.211048627191143, 0.6725107509096409)
```

(Εικόνα 7.25)Στοιχεία πίνακα Anova.

SV	Df	Sum squares	Mean Squares	F value	F value 5%
Block	2	59.39	29.69		
Variety	2	2327.06	1163.53	21.5357	6.9443**
Ea	4	216.11	54.03		
Nrate	3	1529.22	509.74	18.211	3.1599**
Variety:Nrate	6	112.94	18.82	0.6725	2.6613
Eb	18	503.83	27.99		

(Πίνακας 7.2)Πίνακας Anova μετά από αντικατάσταση μεταβλητών.

Από τον πίνακα Anova παρατηρείται ότι υπάρχει σημαντική διαφορά για την ποικιλία και τον βαθμό αζώτου ενώ οι αλληλεπίδραση των δύο δεν είναι σημαντική.

Post-hoc testing

Για την διάκριση των σημαντικά διαφορετικών μεταβλητών γίνεται test σύγκρισης μέσων όρων.

Για την σύγκριση μέσων όρων για τις ποικιλίες αρχικά γίνεται ο υπολογισμός της κρίσιμης τιμής Tukey.

$T = q\sqrt{(MSE/n)}$[19]

Η κρίσιμη τιμή q υπολογίζεται με την εντολή studentized_range.ppf(1-α, k, df) όπου k είναι ο αριθμός των μέσων όρων και df είναι οι βαθμοί ελευθερίας του σφάλματος.Για το συγκεκριμένο παράδειγμα η εντολή αυτή γράφεται ως:

```
q1 = studentized_range.ppf(0.95, v, DFEa)
```

Η τιμή του "n" για τον τύπο "$T = q\sqrt{(MSE/n)}$" προκύπτει ως εξής:

```
nv1 = 0
for x in range(len(data.Yield)):
    if data.variety[x] == 'V1':
        nv1 = nv1 + 1
```

Μιας και οι παρατηρήσεις για την ποικιλία V1 είναι όσες και οι παρατηρήσεις για τις άλλες δύο ποικιλίες, αρκεί ο υπολογισμός του πλήθους παρατηρήσεων μόνο μιας ποικιλίας.

Επομένως το T:

```
Tcrit1 = q1*math.sqrt(MSErrora/nv1)
```

```
In [32]: Tcrit1
Out[32]: 10.694715948418873
```

(Εικόνα 7.26)Κρίσιμη τιμή Tukey για ποικιλίες.

Ακολουθεί ο υπολογισμός των μέσων όρων κάθε ποικιλίας.Αρχικά δημιουργείται μία λίστα με τρία στοιχεία, δηλαδή όσες και οι διαφορετικές ποικιλίες.Σε κάθε στοιχείο της λίστας αποθηκεύεται το σύνολο των αποδόσεων που αντιστοιχεί σε κάθε ποικιλία.

```
vmean=[0,0,0]
for y in range(v):
    for x in range(len(data.Yield)):
                    if data.variety[x] == f'V{y+1}':
                        vmean[y] = vmean[y] + data.Yield[x]

vmean = np.array(vmean)
vmean = vmean/nv1
```

Με αυτόν τον τρόπο η λίστα vmean εμπεριέχει στην πρώτη θέση (vmean[0]) τον μέσο όρο της ποικιλίας V1 , στην δεύτερη θέση (vmean[1]) τον μέσο όρο της ποικιλίας V2 και στην τρίτη θέση (vmean[2]) τον μέσο όρο της ποικιλίας V3.

```
In [144]: vmean
Out[144]: array([85.83333333, 67.41666667, 70.58333333])
```

(Εικόνα 7.27)Μέσοι όροι ποικιλιών.

Τέλος η σύγκριση μέσων όρων μπορεί να γίνει ως εξής:

```
res = np.unique(data.variety)
z = 0
for y in range(len(np.unique(data.variety))):
    for x in range(y+1,len(np.unique(data.variety))):
        print(res[y],"-",res[x], "=",abs(vmean[y] - vmean[x]), "|Critical value =", Tcrit1,
"|Significant at α = 5%",abs(vmean[y] - vmean[x])>Tcrit1 )
        z = z + 1
```

```
    ...: res = np.unique(data.variety)
    ...: z = 0
    ...: for y in range(len(np.unique(data.variety))):
    ...:     for x in range(y+1,len(np.unique(data.variety))):
    ...:         print(res[y],"-",res[x],"=",abs(vmean[y] - vmean[x]), "|Critical value =", Tcrit1, "|Significant at α =
5%",abs(vmean[y] - vmean[x])>Tcrit1 )
    ...:         z = z + 1
V1 - V2 = 18.416666666666657 |Critical value = 10.694715948418873 |Significant at α = 5% True
V1 - V3 = 15.25 |Critical value = 10.694715948418873 |Significant at α = 5% True
V2 - V3 = 3.166666666666657 |Critical value = 10.694715948418873 |Significant at α = 5% False
```

(Εικόνα 7.28)Σύγκριση μέσων όρων ποικιλιών.

Μέσω σύγκρισης με την μεταβλητή Tcrit1 το συμπέρασμα είναι πως τα ζευγάρια V1V2 και V1V3 έχουν σημαντική διαφορά.
Με τον ίδιο τρόπο γίνεται και η σύγκριση μέσων όρων για τα επίπεδα αζώτου. Η μόνη διαφορά είναι πως εδώ το MSE είναι η μεταβλητή MSErrorb

```
q2 = studentized_range.ppf(0.95, N, DFEb)
nN1 = 0
for x in range(len(data.Yield)):
    if data.Nrate[x] == 'N1':
        nN1 = nN1 + 1

Tcrit2 = q2*math.sqrt(MSErrorb/nN1)
nmean=[0,0,0,0]
for y in range(N):
    for x in range(len(data.Yield)):
                    if data.Nrate[x] == f'N{y+1}':
                        nmean[y] = nmean[y] + data.Yield[x]
```

```
nmean = np.array(nmean)
nmean = nmean/nN1
#Mean Comparison
res = np.unique(data.Nrate)
z = 0
for y in range(len(np.unique(data.Nrate))):
   for x in range(y+1,len(np.unique(data.Nrate))):
      print(res[y],"-",res[x],"=",abs(nmean[y] - nmean[x]), "|Critical value =", Tcrit2,"|Significant
at α = 5%",abs(nmean[y] - nmean[x])>Tcrit2)
      z = z + 1
```

```
    ...: res = np.unique(data.Nrate)
    ...: z = 0
    ...: for y in range(len(np.unique(data.Nrate))):
    ...:     for x in range(y+1,len(np.unique(data.Nrate))):
    ...:         print(res[y],"-",res[x],"=",abs(nmean[y] - nmean[x]), "|Critical value =", Tcrit2,"|Significant at α =
5%",abs(nmean[y] - nmean[x])>Tcrit2)
    ...:         z = z + 1
    ...:
N1 - N2 = 0.22222222222221433 |Critical value = 7.048840264514708 |Significant at α = 5% False
N1 - N3 = 5.333333333333329 |Critical value = 7.048840264514708 |Significant at α = 5% False
N1 - N4 = 12.444444444444443 |Critical value = 7.048840264514708 |Significant at α = 5% True
N2 - N3 = 5.555555555555543 |Critical value = 7.048840264514708 |Significant at α = 5% False
N2 - N4 = 12.222222222222229 |Critical value = 7.048840264514708 |Significant at α = 5% True
N3 - N4 = 17.77777777777777 |Critical value = 7.048840264514708 |Significant at α = 5% True
```

(Εικόνα 7.29) Σύγκριση μέσων όρων αζώτου.

Μέσω σύγκρισης με την μεταβλητή Tcrit2 το συμπέρασμα είναι πως τα ζευγάρια N1N4, N2N4 και N3N4 έχουν σημαντική διαφορά.

Σύνοψη κώδικα

```
import pandas as pd
import numpy as np
import math
import scipy.stats
from scipy.stats import studentized_range


data = pd.read_excel('splitplot.xlsx')
data = pd.DataFrame(data, columns=['variety','Nrate','block','Yield'])

#Grant Total
GT = sum(data.Yield)
#Levels of Factor A
v = len(np.unique(data.variety))
#Levels of Factor B
N = len(np.unique(data.Nrate))
#Levels of Blocks
b = len(np.unique(data.block))
#Correction Factor
CF = (GT**2)/(v*N*b)
#Tota Sum Of Square
TSS = sum(data.Yield**2) - CF

#Replication sum of square

rss = [0,0,0]
for y in range(v):
   for x in range(len(data.block)):
                if data.block[x] == y+1:
                   rss[y] = rss[y] + data.Yield[x]

RSS = 0
for i in rss:
   RSS = RSS + i**2

RSS = (RSS /(v*N)) - CF
```

```
#Sum of Square due to Factor A/Whole-plot Factor (Variety)
vb= [0,0,0]
for y in range(v):
    for x in range(len(data.Yield)):
                    if data.variety[x] == f'V{y+1}':
                        vb[y] = vb[y] + data.Yield[x]


Variety_SS = 0
for i in vb:
    Variety_SS = Variety_SS + i**2

Variety_SS = (Variety_SS/(N*b)) - CF

#Error A Sum of square
SSVB = 0
vb = 0
for u in range(v):
    for y in range(v+1):
        SSVB = SSVB + vb**2
        vb = 0
        for x in range(len(data.Yield)):
            if data.variety[x] == f'V{u+1}':
                if data.block[x] == y+1:
                    vb = vb + data.Yield[x]


Ea = (SSVB/N)-CF-Variety_SS-RSS


#Sum of Square due to Factor B/Split-Plot Factor (Nrate)

Nv = [0,0,0,0]
for y in range(v+1):
    for x in range(len(data.Yield)):
        if data.Nrate[x] == f'N{y+1}':
            Nv[y] = Nv[y] + data.Yield[x]
```

```
NRate_SS = 0
for i in Nv:
   NRate_SS = NRate_SS + i**2

NRate_SS = (NRate_SS/(v*b)) - CF

#Sum of Square due to Interaction
SSVN = 0
vb = 0

for y in range(N+1):
   for u in range(v):
      SSVN = SSVN + vb**2
      vb = 0
      for x in range(len(data.Yield)):
         if data.Nrate[x] == f'N{y+1}':
            if data.variety[x] == f'V{u+1}':
               vb = vb + data.Yield[x]

Interaction = (SSVN/b) - CF - Variety_SS - NRate_SS


#Error B Sum of Square

Eb = TSS - RSS - Variety_SS - Ea - NRate_SS - Interaction



# Degrees of Freedom
#Variety
DFv = v - 1
#Nrate
DFN = N - 1
#Block
DFb = b - 1
#Error a
DFEa = DFv*DFb

#Interaction
DFInt = DFv*DFN
```

```
#Error b
DFEb = v*DFb*DFN


#Mean_Sq
#Blcok
MSBlock = RSS / DFb
#Variety
MSVar  = Variety_SS / DFv
#Nrate
MSNrate = NRate_SS / DFN
#Error A
MSErrora = Ea / DFEa
#Interaction
MSInteraction = Interaction / DFInt
#Error B
MSErrorb = Eb / DFEb
#Fvalue
Fvar = MSVar / MSErrora
FNrate = MSNrate / MSErrorb
FInteraction = MSInteraction / MSErrorb
#Fcrit
FAcrit =  scipy.stats.f.ppf(0.95, DFv, DFEa)
FBcrit = scipy.stats.f.ppf(0.95, DFN, DFEb)
Fintcrit = scipy.stats.f.ppf(0.95, DFInt, DFEb)
#Tukey
#variety
q1 = studentized_range.ppf(0.95, v, DFEa)
nv1 = 0
for x in range(len(data.Yield)):
   if data.variety[x] == 'V1':
      nv1 = nv1 + 1
Tcrit1 = q1*math.sqrt(MSErrora/nv1)

vmean= vmean=[0,0,0]
for y in range(v):
   for x in range(len(data.Yield)):
                  if data.variety[x] == f'V{y+1}':
                     vmean[y] = vmean[y] + data.Yield[x]
```

```
vmean = np.array(vmean)
vmean = vmean/nv1
#Mean Comparison
res = np.unique(data.variety)
z = 0
for y in range(len(np.unique(data.variety))):
   for x in range(y+1,len(np.unique(data.variety))):
      print(res[y],"-",res[x],"=",abs(vmean[y] - vmean[x]), "|Critical value =", Tcrit1,
"|Significant at α = 5%",abs(vmean[y] - vmean[x])>Tcrit1 )
      z = z + 1

#Nrate
q2 = studentized_range.ppf(0.95, N, DFEb)
nN1 = 0
for x in range(len(data.Yield)):
   if data.Nrate[x] == 'N1':
     nN1 = nN1 + 1

Tcrit2 = q2*math.sqrt(MSErrorb/nN1)
nmean=[0,0,0,0]
for y in range(N):
   for x in range(len(data.Yield)):
                  if data.Nrate[x] == f'N{y+1}':
                    nmean[y] = nmean[y] + data.Yield[x]

nmean = np.array(nmean)
nmean = nmean/nN1
#Mean Comparison

res = np.unique(data.Nrate)
z = 0
for y in range(len(np.unique(data.Nrate))):
   for x in range(y+1,len(np.unique(data.Nrate))):
      print(res[y],"-",res[x],"=",abs(nmean[y] - nmean[x]), "|Critical value =", Tcrit2,
"|Significant at α = 5%",abs(nmean[y] - nmean[x])>Tcrit2)
      z = z + 1
```

8. Strip-plot design

Ο σχεδιασμός strip-plot είναι ένας πειραματικός σχεδιασμός δύο παραγοντικών πειραμάτων. Είναι παρόμοιος με τον split-plot σχεδιασμό αλλά κυρίως χρησιμοποιείται όταν και οι δύο υπό εξέταση παράγοντες(A,B) είναι δύσκολο να διαφοροποιηθούν. Πρωταρχικός στόχος αυτού του σχεδιασμού είναι ο ακριβής υπολογισμός της επίδρασης που έχει η αλληλεπίδραση μεταξύ των δύο παραγόντων και όχι τόσο η επίδραση του κάθε παράγοντα ξεχωριστά. Η ακρίβεια υπολογισμού της επίδρασης των δύο κύριων παραγόντων θυσιάζεται για να ενισχυθεί η ακρίβεια υπολογισμού της επίδρασης που έχει η αλληλεπίδρασή τους[21]. .Σε ένα πείραμα με 4 επίπεδα εφαρμογής λιπάσματος(F1,F2,F3,F4), 3 διαφορετικές ποικιλίες καλαμποκιού(V1,V2,V3) σε 4 διαφορετικά χωράφια, εξετάζεται η απόδοση της κάθε ποικιλίας σε σχέση με το διαφορετικό λίπασμα που εφαρμόστηκε. Ο σχεδιασμός σε strip-plot του παραπάνω πειράματος θα γινόταν ως εξής:

Χωράφι 1	Χωράφι 2	Χωράφι 3	Χωράφι 4

Κάθε ένα από τα τέσσερα χωράφια αποτελεί ένα block του πειράματος. Αρχικά το κάθε χωράφι διαιρείται οριζόντια, ανάλογα με τον αριθμό των επιπέδων εφαρμογής λιπάσματος δηλαδή σε τέσσερα μέρη. Σε αυτήν την περίπτωση το κάθε επίπεδο εφαρμογής λιπάσματος αποτελεί έναν οριζόντιο παράγοντα. Στην συνέχεια ο κάθε οριζόντιος παράγοντας τοποθετείται τυχαία σε ένα από τα τέσσερα μέρη του χωραφιού όπως φαίνεται παρακάτω.

F1	F4	F3	F2
F4	F3	F2	F1
F3	F2	F1	F3
F2	F1	F4	F4
Χωράφι 1	Χωράφι 2	Χωράφι 3	Χωράφι 4

Μετά την τυχαία εναπόθεση του οριζόντιου παράγοντα γίνεται κάθετος διαχωρισμός των χωραφιών ανάλογα με τον αριθμό των ποικιλιών δηλαδή σε τρία μέρη. Σε αυτήν την περίπτωση η κάθε ποικιλία αποτελεί έναν κάθετο παράγοντα ο οποίος τοποθετείται τυχαία σε μία από τις τρεις κάθετες λωρίδες του χωραφιού όπως παρακάτω.

F1V3	F1V1	F1V2	F4V1	F4V2	F4V3	F3V2	F3V3	F3V1	F2V2	F2V1	F2V3
F4V3	F4V1	F4V2	F3V1	F3V2	F3V3	F2V2	F2V3	F2V1	F1V2	F1V1	F1V3
F3V3	F3V1	F3V2	F2V1	F2V2	F2V3	F1V2	F1V3	F1V1	F3V2	F3V1	F3V3
F2V3	F2V1	F2V2	F1V1	F1V2	F1V3	F4V2	F4V3	F4V1	F4V2	F4V1	F4V3
Χωράφι 1			Χωράφι 2			Χωράφι 3			Χωράφι 4		

Ουσιαστικά πραγματοποιείται μια ξεχωριστή τυχαιοποίηση σε κάθε χωράφι, για κάθε παράγοντα αυξάνοντας έτσι την ακρίβεια υπολογισμού της επίδρασης που έχει η αλληλεπίδραση μεταξύ των παραγόντων[22].

Αντιθέτως στον split-plot σχεδιασμό η τυχαιοποίηση σε split-plots του split-plot παράγοντα γίνεται βάση της τυχαιοποίησης του whole-plot παράγοντα εστιάζοντας περισσότερο στην επίδραση του κάθε παράγοντα ξεχωριστά. Τα δεδομένα του παραδείγματος φαίνονται στην παρακάτω εικόνα.

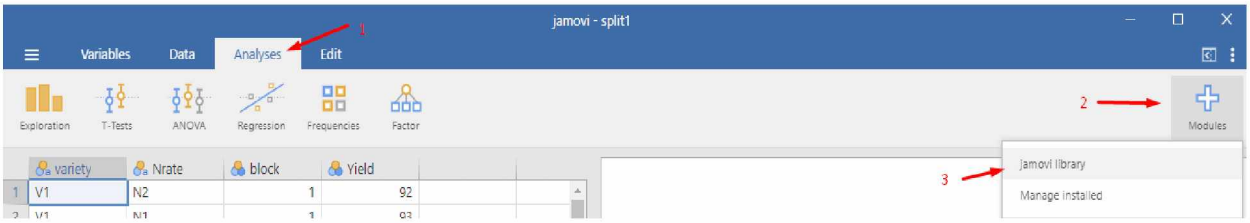
block	varieties	fertilizer	Yield
1	V1	F1	10.2
1	V1	F2	11.1
1	V1	F3	6.8
1	V1	F4	5.3
1	V2	F1	8
1	V2	F2	9.7
1	V2	F3	8.6
1	V2	F4	3.4
1	V3	F1	2
1	V3	F2	10.9
1	V3	F3	2.2
1	V3	F4	2.1
2	V1	F1	10.1
2	V1	F2	9.8
2	V1	F3	9.5
2	V1	F4	7.5
2	V2	F1	9.7
2	V2	F2	7.9
2	V2	F3	9.6
2	V2	F4	4.2
2	V3	F1	6.1
2	V3	F2	8.4
2	V3	F3	4.9
2	V3	F4	0.9
3	V1	F1	12.1
3	V1	F2	8.6
3	V1	F3	9.5
3	V1	F4	4.6
3	V2	F1	12
3	V2	F2	10.3
3	V2	F3	9.5
3	V2	F4	7.3
3	V3	F1	4.8
3	V3	F2	6.5
3	V3	F3	4.4
3	V3	F4	3.4
4	V1	F1	12.3
4	V1	F2	9.4
4	V1	F3	10.3
4	V1	F4	7.3
4	V2	F1	7.8
4	V2	F2	11.2
4	V2	F3	10
4	V2	F4	7.6
4	V3	F1	6.7
4	V3	F2	9.2
4	V3	F3	3.6
4	V3	F4	2.3

(Εικόνα 8.1)Πλαίσιο δεδομένων strip-plot πειράματος.

Παρακάτω περιγράφεται ο τρόπος ανάλυσης ενός strip-plot σχεδιασμού με παράδειγμα το παραπάνω πείραμα. Η ανάλυση αυτή θα γίνει μέσω του στατιστικού λογισμικού Jamovi, της γλώσσας προγραμματισμού R αλλά και της γλώσσας Python.

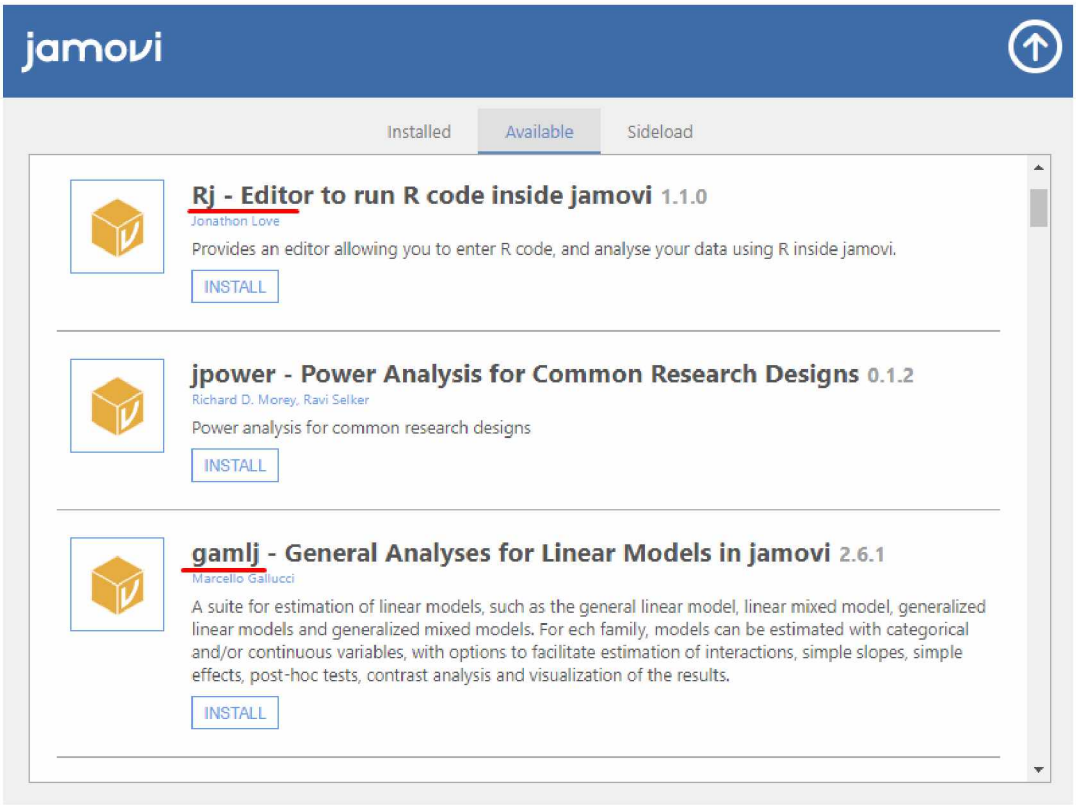
8.1 Strip plot Jamovi:

Με την εισαγωγή του αρχείου excel στο jamovi η ανάλυση strip-plot σχεδιασμού απαιτεί την χρήση δύο πρόσθετων πακέτων από την βιβλιοθήκη του jamovi. Η προσθήκη των πακέτων γίνεται από την καρτέλα "Analyses" κάνοντας click στο "Modules" στο πάνω μέρος του παραθύρου και στην συνέχεια "jamovi library".



(Εικόνα 8.2)Εισαγωγή στην βιβλιοθήκη Jamovi.

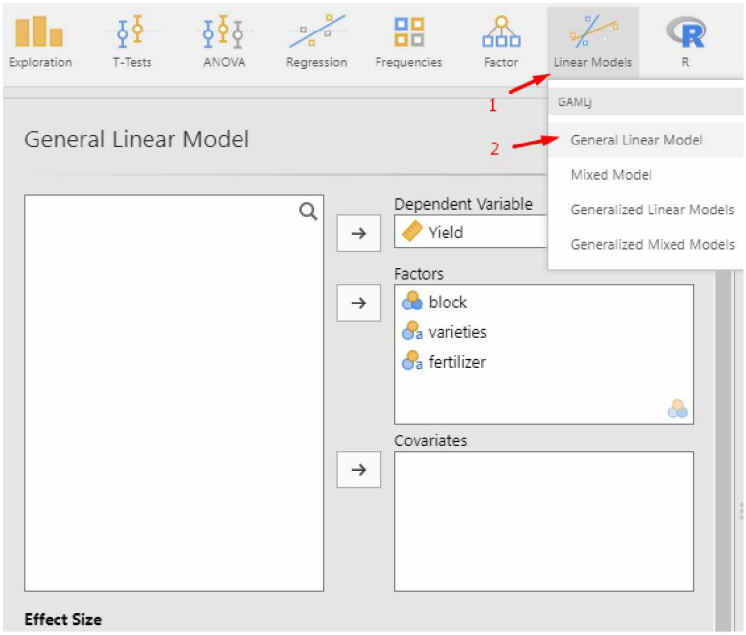
Τα πακέτα που θα εγκατασταθούν είναι το "Rj Editor" και το "gamlj". Η εγκατάστασή τους γίνεται πατώντας στο κουμπί "INSTALL" κάτω από κάθε πακέτο.



(Εικόνα 8.3)Εγκατάσταση πακέτων.

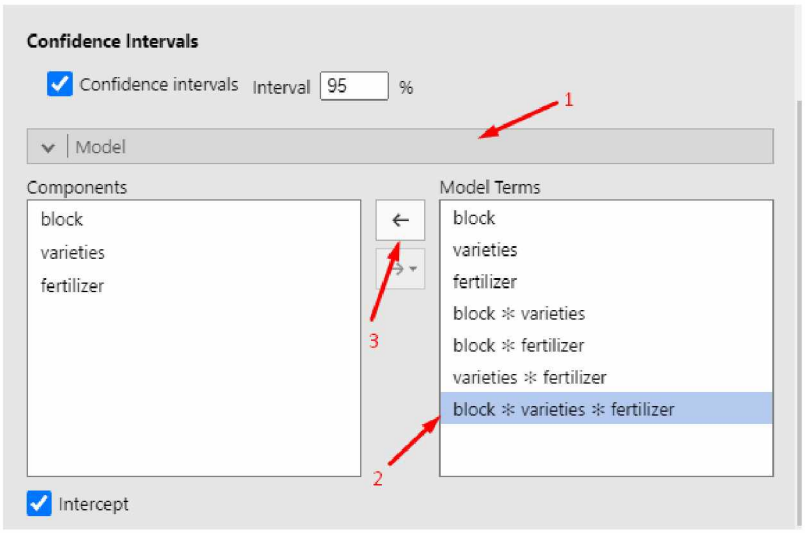
Μετά την εγκατάσταση των δύο πακέτων, για την ανάλυση strip-plot, από την καρτέλα "Analyses" επιλέγεται το "Linear models" και στην συνέχεια το "General Linear Model".

Ως dependent Variable χρησιμοποιείται η μεταβλητή απόκρισης ενώ ως factors η μεταβλητή block, η μεταβλητή των vertical factors και η μεταβλητή των horizontal factors με αυτήν την σειρά.



(Εικόνα 8.4) Επιλογή όρων για ανάλυση.

Στην συνέχεια αφαιρούνται από τα model terms ο όρος block*horizontal factor*verticalfactor .



(Εικόνα 8.5) Αφαίρεση όρων από τα Model terms.

Στα δεξιά σχηματίζεται ο πίνακας Anova. Στον πίνακα Anova ο όρος "block * variety" αποτελεί το Error a της ανάλυσης, ο όρος "block * fertilizer" αποτελεί το Error b της ανάλυσης, ενώ ο όρος "Residuals" αποτελεί το Error c της ανάλυσης.

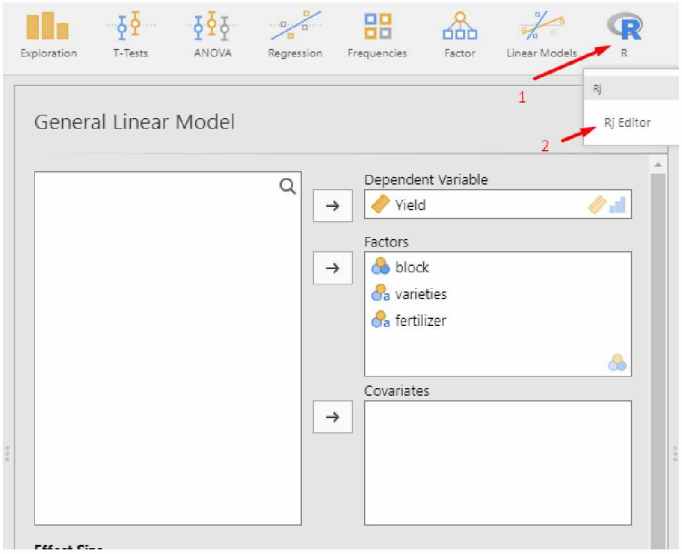
Model Results

ANOVA Omnibus tests

	SS	df	F	p
Model	405.05	29	7.371	< .001
block	13.69	3	2.409	0.101
varieties	163.01	2	43.014	< .001
fertilizer	152.68	3	26.860	< .001
block ✻ varieties	8.57	6	0.754	0.614
block ✻ fertilizer	26.77	9	1.570	0.198
varieties ✻ fertilizer	40.32	6	3.547	0.017
Residuals	34.11	18		
Total	439.16	47		

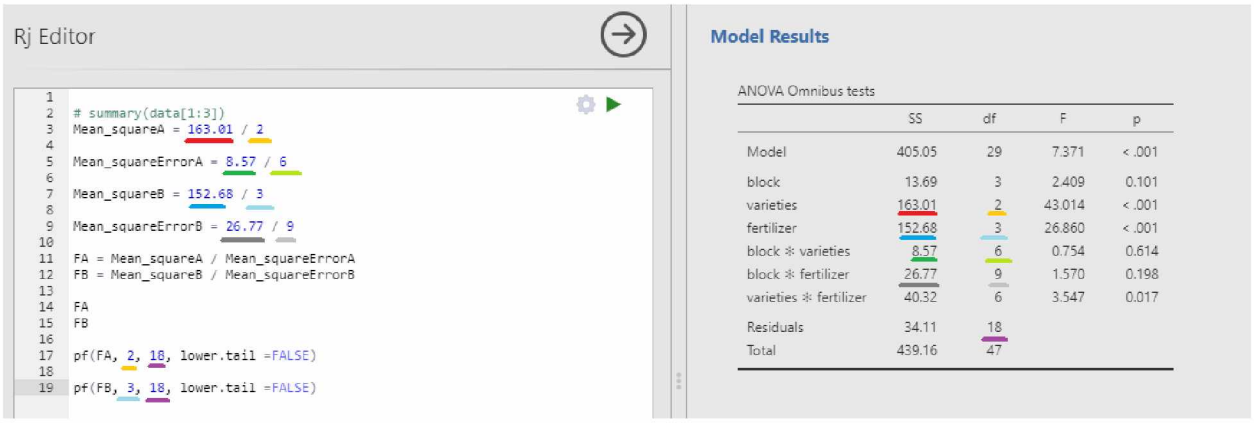
(Εικόνα 8.6) Πίνακας Anova

Λόγο περιορισμών του jamovi ο υπολογισμός της τιμής F και p για των vertical factor και horizontal factor είναι λάθος και δεν πρέπει να ληφθούν υπόψιν. Για τον σωστό υπολογισμό των δύο αυτών τιμών, γίνεται χρήση του Rj Editor.



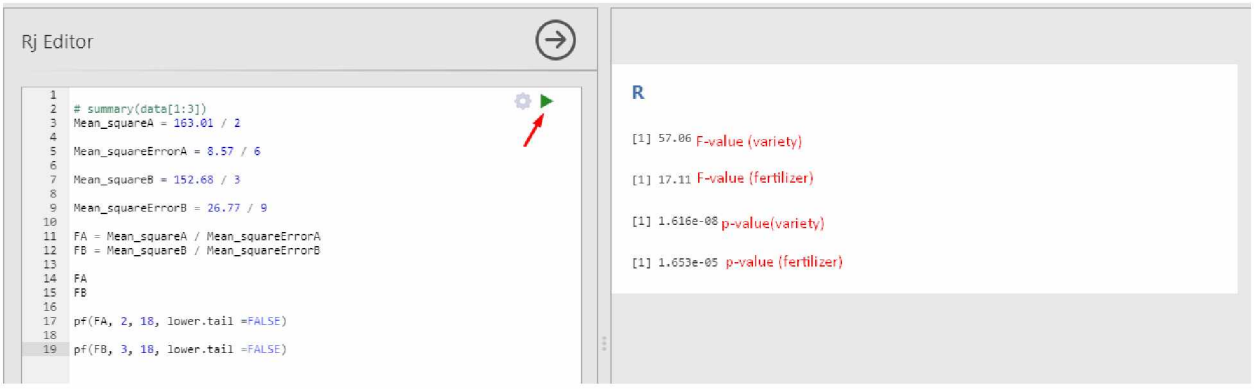
(Εικόνα 8.7)Επιλογή Rj editor.

Για τον υπολογισμό της τιμής F αρχικά υπολογίζεται το Mean square για τον κάθε παράγοντα και για το Error που αντιστοιχεί σε κάθε παράγοντα.Το mean square υπολογίζεται διαιρώντας το άθροισμα τετραγώνων με τον βαθμό ελευθερίας. Η τιμή F υπολογίζεται μέσα από την διαίρεση του Mean square του παράγοντα με το Mean square του Error.Τέλος ο υπολογισμός της τιμής p γίνεται μέσω του Rj Editor γράφοντας "pf(τιμή F, βαθμός ελευθερίας του παράγοντα, βαθμός ελευθερίας του Error c, lower.tail = FALSE)".



(Εικόνα 8.8)Επεξεργαστής κειμένου Rj Editor.

Για την εμφάνιση των αποτελεσμάτων αρκεί να γίνει κλικ στο πράσινο βελάκι.

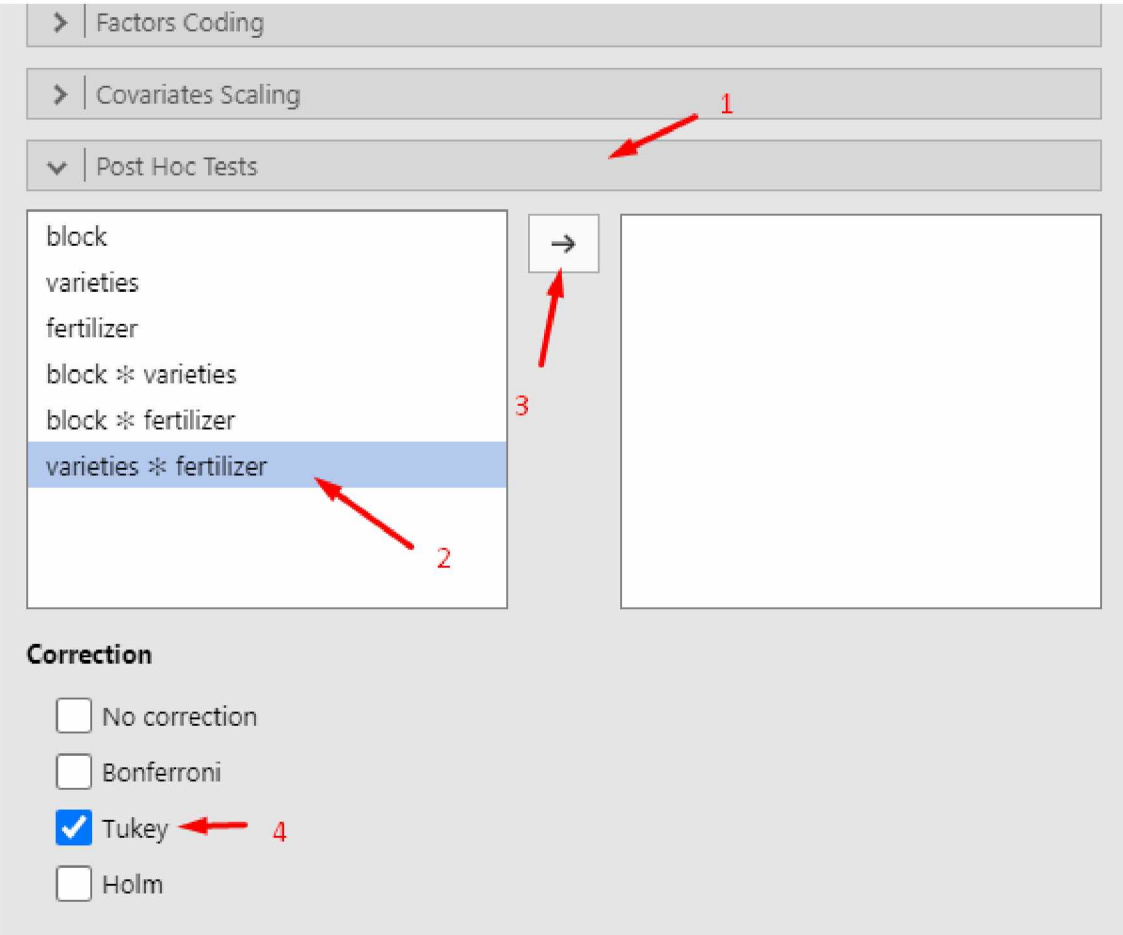


(Εικόνα 8.9)Υπολογισμός F και p value.

Από τα αποτελέσματα παρατηρείται ότι υπάρχει σημαντική διαφορά για την αλληλεπίδραση των δύο βασικών παραγόντων.

Post-hoc testing

Στην συνέχεια πραγματοποιείται tukey test για την όρο της αλληλεπίδρασης των δύο κύριων μεταβλητών. Από τα Post Hoc Tests επιλέγεται ο όρος αλληλεπίδρασης και επιλέγεται η μέθοδος Tukey.



(Εικόνα 8.10) Διεξαγωγή Tukey test σε Jamovi.

Με αυτόν τον τρόπο κάτω από τον προσχηματισμένο πίνακα Anova εμφανίζονται τα αποτελέσματα του τεστ Tukey.

Post Hoc Tests

Post Hoc Comparisons - varieties ✻ fertilizer

Comparison								
varieties	fertilizer		varieties	fertilizer	Difference	SE	t	df	p_{tukey}
V1	F1	-	V1	F2	1.4500	0.973	1.4897	18.0	0.925
V1	F1	-	V1	F3	2.1500	0.973	2.2089	18.0	0.563
V1	F1	-	V1	F4	5.0000	0.973	5.1369	18.0	0.003
V1	F1	-	V2	F1	1.8000	0.973	1.8493	18.0	0.773
V1	F1	-	V2	F2	1.4000	0.973	1.4383	18.0	0.940
V1	F1	-	V2	F3	1.7500	0.973	1.7979	18.0	0.800
V1	F1	-	V2	F4	5.5500	0.973	5.7020	18.0	< .001
V1	F1	-	V3	F1	6.2750	0.973	6.4468	18.0	< .001
V1	F1	-	V3	F2	2.4250	0.973	2.4914	18.0	0.401
V1	F1	-	V3	F3	7.4000	0.973	7.6026	18.0	< .001
V1	F1	-	V3	F4	9.0000	0.973	9.2464	18.0	< .001
V1	F2	-	V1	F3	0.7000	0.973	0.7192	18.0	1.000
V1	F2	-	V1	F4	3.5500	0.973	3.6472	18.0	0.058
V1	F2	-	V2	F2	−0.0500	0.973	−0.0514	18.0	1.000
V1	F2	-	V2	F3	0.3000	0.973	0.3082	18.0	1.000
V1	F2	-	V2	F4	4.1000	0.973	4.2123	18.0	0.019
V1	F2	-	V3	F2	0.9750	0.973	1.0017	18.0	0.996
V1	F2	-	V3	F3	5.9500	0.973	6.1129	18.0	< .001
V1	F2	-	V3	F4	7.5500	0.973	7.7567	18.0	< .001
V1	F3	-	V1	F4	2.8500	0.973	2.9280	18.0	0.210
V1	F3	-	V2	F3	−0.4000	0.973	−0.4110	18.0	1.000
V1	F3	-	V2	F4	3.4000	0.973	3.4931	18.0	0.078
V1	F3	-	V3	F3	5.2500	0.973	5.3938	18.0	0.002
V1	F3	-	V3	F4	6.8500	0.973	7.0376	18.0	< .001
V1	F4	-	V2	F4	0.5500	0.973	0.5651	18.0	1.000
V1	F4	-	V3	F4	4.0000	0.973	4.1095	18.0	0.023
V2	F1	-	V1	F2	−0.3500	0.973	−0.3596	18.0	1.000
V2	F1	-	V1	F3	0.3500	0.973	0.3596	18.0	1.000
V2	F1	-	V1	F4	3.2000	0.973	3.2876	18.0	0.113
V2	F1	-	V2	F2	−0.4000	0.973	−0.4110	18.0	1.000
V2	F1	-	V2	F3	−0.0500	0.973	−0.0514	18.0	1.000
V2	F1	-	V2	F4	3.7500	0.973	3.8527	18.0	0.039
V2	F1	-	V3	F1	4.4750	0.973	4.5975	18.0	0.009
V2	F1	-	V3	F2	0.6250	0.973	0.6421	18.0	1.000
V2	F1	-	V3	F3	5.6000	0.973	5.7533	18.0	< .001
V2	F1	-	V3	F4	7.2000	0.973	7.3971	18.0	< .001
V2	F2	-	V1	F3	0.7500	0.973	0.7705	18.0	1.000
V2	F2	-	V1	F4	3.6000	0.973	3.6986	18.0	0.053
V2	F2	-	V2	F3	0.3500	0.973	0.3596	18.0	1.000
V2	F2	-	V2	F4	4.1500	0.973	4.2636	18.0	0.017
V2	F2	-	V3	F2	1.0250	0.973	1.0531	18.0	0.993
V2	F2	-	V3	F3	6.0000	0.973	6.1643	18.0	< .001

(Εικόνα 8.11.Α) Σύγκριση μέσων όρων αλληλεπίδρασης.

V2	F2	-	V3	F4	7.6000	0.973	7.8081	18.0	< .001
V2	F3	-	V1	F4	3.2500	0.973	3.3390	18.0	0.103
V2	F3	-	V2	F4	3.8000	0.973	3.9040	18.0	0.035
V2	F3	-	V3	F3	5.6500	0.973	5.8047	18.0	< .001
V2	F3	-	V3	F4	7.2500	0.973	7.4485	18.0	< .001
V2	F4	-	V3	F4	3.4500	0.973	3.5445	18.0	0.070
V3	F1	-	V1	F2	−4.8250	0.973	−4.9571	18.0	0.004
V3	F1	-	V1	F3	−4.1250	0.973	−4.2379	18.0	0.018
V3	F1	-	V1	F4	−1.2750	0.973	−1.3099	18.0	0.967
V3	F1	-	V2	F2	−4.8750	0.973	−5.0085	18.0	0.004
V3	F1	-	V2	F3	−4.5250	0.973	−4.6489	18.0	0.008
V3	F1	-	V2	F4	−0.7250	0.973	−0.7449	18.0	1.000
V3	F1	-	V3	F2	−3.8500	0.973	−3.9554	18.0	0.032
V3	F1	-	V3	F3	1.1250	0.973	1.1558	18.0	0.986
V3	F1	-	V3	F4	2.7250	0.973	2.7996	18.0	0.257
V3	F2	-	V1	F3	−0.2750	0.973	−0.2825	18.0	1.000
V3	F2	-	V1	F4	2.5750	0.973	2.6455	18.0	0.324
V3	F2	-	V2	F3	−0.6750	0.973	−0.6935	18.0	1.000
V3	F2	-	V2	F4	3.1250	0.973	3.2106	18.0	0.130
V3	F2	-	V3	F3	4.9750	0.973	5.1112	18.0	0.003
V3	F2	-	V3	F4	6.5750	0.973	6.7550	18.0	< .001
V3	F3	-	V1	F4	−2.4000	0.973	−2.4657	18.0	0.415
V3	F3	-	V2	F4	−1.8500	0.973	−1.9007	18.0	0.745
V3	F3	-	V3	F4	1.6000	0.973	1.6438	18.0	0.871

(Εικόνα 8.11.Β) Σύγκριση μέσων όρων αλληλεπίδρασης.

8.2 Strip-plot R:

Η ανάλυση strip-plot σχεδιασμού στην R μπορεί να γίνει με χρήση της βιβλιοθήκης agricolae.

```
library(agricolae)
```

Αρχικά γίνεται η εισαγωγή δεδομένων μέσα από ένα αρχείο excel.Αυτό προϋποθέτει την χρήση της βιβλιοθήκης readxl.

```
library(readxl)
```

μέσα από την εντολή :

```
df=read_excel('filename.xlsx')
```

Με αυτόν τον τρόπο η μεταβλητή df αποτελεί το σύνολο των δεδομένων στο οποίο θα γίνει η στατιστική ανάλυση. Με χρήση της εντολής head(df) προβάλλονται οι πρώτες 6 γραμμές των δεδομένων.

```
> head(df)
# A tibble: 6 x 4
  block varieties fertilizer Yield
  <dbl> <fct>     <fct>      <dbl>
1     1 V1        F1          10.2
2     1 V1        F2          11.1
3     1 V1        F3           6.8
4     1 V1        F4           5.3
5     1 V2        F1           8
6     1 V2        F2           9.7
>
```

(Εικόνα 8.12) Έλεγχος σωστής εισαγωγής δεδομένων.

Μετά την εισαγωγή του πλαισίου δεδομένων όπως αυτό της εικόνας 8.1 μπορεί να γίνει χρήση της εντολής str(df) για τον έλεγχο της μορφής κάθε στήλης του πλαισίου δεδομένων.

```
str(df)
```

```
> str(df)
tibble [48 x 4] (S3: tbl_df/tbl/data.frame)
 $ block     : num [1:48] 1 1 1 1 1 1 1 1 1 1 ...
 $ varieties : chr [1:48] "V1" "V1" "V1" "V1" ...
 $ fertilizer: chr [1:48] "F1" "F2" "F3" "F4" ...
 $ Yield     : num [1:48] 10.2 11.1 6.8 5.3 8 9.7 8.6 3.4 2 10.9 ...
```

(Εικόνα 8.13) Έλεγχος μορφής εισακτέων δεδομένων.

Σε αυτό το παράδειγμα οι ποικιλίες και το λίπασμα διαβάζονται ως χαρακτήρες(chr).Θα πρέπει αρχικά να γίνει η μετατροπή τους σε factors με την εντολή as.factor(filename$variabel) ή πιο συγκεκριμένα για το παρών παράδειγμα:

df$varieties = as.factor(df$varieties)
df$fertilizer = as.factor(df$fertilizer)
Για την ανάλυση διακύμανσης ενός strip-plot σχεδιασμού χρησιμοποιείται το εξής μοντέλο.

model = with(df, strip.plot(block, μεταβλητή κάθετου παράγοντα, μεταβλητή οριζόντιου παράγοντα, μεταβλητή απόκρισης))

Όπως προαναφέρθηκε για το συγκεκριμένο παράδειγμα ,vertical factors αποτελούν οι ποικιλίες, horizontal factor αποτελεί το επίπεδο εφαρμογής λιπάσματος ενώ η απόδοση είναι η μεταβλητή απόκρισης. Συνεπώς το παραπάνω μοντέλο γράφεται ώς:

model = with(df, strip.plot(block, varieties, fertilizer, Yield))

Με αυτόν τον τρόπο δημιουργείται η ανάλυση strip-plot.

```
ANALYSIS STRIP PLOT:  Yield
Class level information

varieties       :  V1 V2 V3
fertilizer      :  F1 F2 F3 F4
block   :  1 2 3 4

Number of observations:  48

model Y: Yield ~ block + varieties + Ea + fertilizer + Eb + fertilizer:varieties + Ec

Analysis of Variance Table

Response: Yield
                   Df  Sum Sq Mean Sq F value    Pr(>F)
block               3  13.692   4.564
varieties           2 163.007  81.503 57.0397 0.0001248 ***
Ea                  6   8.573   1.429
fertilizer          3 152.685  50.895 17.1086 0.0004638 ***
Eb                  9  26.773   2.975
fertilizer:varieties  6  40.320   6.720  3.5465 0.0170071 *
Ec                 18  34.107   1.895
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

cv(a) = 16 %, cv(b) = 23 %, cv(c) = 18.4 %, Mean = 7.491667
```

(Εικόνα 8.14) Πίνακας Anova.

Από τον πίνακα Anova παρατηρείται ότι ο όρος αλληλεπίδρασης (fertilizer:varieties) είναι σημαντικός οπότε δεν θα εξεταστεί η ξεχωριστή επίδραση των δύο κύριων μεταβλητών , δηλαδή των ποικιλιών και του λιπάσματος .

Post-hoc testing

Για την διάκριση των σημαντικά διαφορετικών μεταβλητών γίνεται test σύγκρισης μέσων όρων LSD. Για την σύγκριση μέσων όρων του vertical factor (varieties) θα χρησιμοποιηθεί ο βαθμός ελευθερίας και το τετράγωνο του μέσου όρου του Ea. Για τον horizontal factor θα χρησιμοποιηθεί ο βαθμός ελευθερίας και το τετράγωνο του μέσου όρου του Eb.Τέλος για τον παράγοντα αλληλεπίδρασης θα χρησιμοποιηθεί ο βαθμός ελευθερίας και το τετράγωνο του μέσου όρου του Ec[23]. Αυτό μπορεί να διευκρινιστεί με τις παρακάτω εντολές:

```
gla = model$gl.a
glb = model$gl.b
glc = model$gl.c
Ea = model$Ea
Eb = model$Eb
Ec = model$Ec
```

Για την διεξαγωγή του τεστ σύγκρισης μέσων όρων του vertical factor γίνεται χρήση της εντολής:

```
out1 = with(df,LSD.test(μεταβλητή απόκρισης, μεταβλητή κάθετου παράγοντα ,gla, Ea, console =TRUE))
```

Ή για το συγκεκριμένο παράδειγμα

```
out1 = with(df,LSD.test(Yield,varieties,gla, Ea, console = TRUE))
```

```
   Yield groups
V1 9.025      a
V2 8.550      a
V3 4.900      b
```

(Εικόνα 8.15) Αποτελέσματα least significant difference test ποικιλιών.

Από το αποτέλεσμα φαίνεται πως η ποικιλία V3 είναι η λιγότερο παραγωγική. Δεν υπάρχει στατιστικά σημαντική διαφορά μεταξύ των V1,V2, αφού στη στήλη groups του LSD test μοιράζονται το ίδιο γράμμα (a).

Για την διεξαγωγή του τεστ σύγκρισης μέσων όρων του horizontal factor γίνεται χρήση της εντολής:

out2 = with(df,LSD.test(μεταβλητή απόκρισης, μεταβλητή οριζόντιου παράγοντα ,glb, Eb, console =TRUE))

Ή για το συγκεκριμένο παράδειγμα

out2 = with(df,LSD.test(Yield,fertilizer,glb, Eb, console = TRUE))

```
      Yield groups
F2 9.416667      a
F1 8.483333     ab
F3 7.408333      b
F4 4.658333      c
```

(Εικόνα 8.16) Αποτελέσματα least significant difference test επιπέδων λιπάσματος.

Από το αποτέλεσμα φαίνεται πως το επίπεδο λιπάσματος F2 αποφέρει την μεγαλύτερη απόδοση αλλά δεν έχει στατιστικά σημαντική διαφορά από το επίπεδο λιπάσματος F1.

Τέλος για την διεξαγωγή του τεστ σύγκρισης μέσων όρων της αλληλεπίδρασης γίνεται χρήση της εντολής:

out3 = with(df,LSD.test(μεταβλητήαπόκρισης,vertical factor:horizontal factor, glc, Ec, console = TRUE))

ή για το συγκεκριμένο παράδειγμα

out3 = with(df,LSD.test(Yield,fertilizer:varieties, glc, Ec, console = TRUE))

```
       Yield groups
F1:V1 11.175      a
F2:V2  9.775     ab
F2:V1  9.725     ab
F3:V2  9.425     ab
F1:V2  9.375     ab
F3:V1  9.025      b
F2:V3  8.750      b
F4:V1  6.175      c
F4:V2  5.625     cd
F1:V3  4.900     cd
F3:V3  3.775     de
F4:V3  2.175      e
> |
```

(Εικόνα 8.17) Αποτελέσματα least significant difference test αλληλεπίδρασης.

Από το αποτέλεσμα φαίνεται πως ο συνδυασμός F1:V1 αποφέρει την μεγαλύτερη απόδοση χωρίς να υπάρχει στατιστικά σημαντική διαφορά με τα επίπεδα F2:V2, F2:V1, F3:V2, F1:V2.

Σύνοψη κώδικα

```
library(agricolae)
library(readxl)
df=read_excel('stripplot.xlsx')
head(df)
str(df)
df$varieties = as.factor(df$varieties)
df$fertilizer = as.factor(df$fertilizer)
model = with(df, strip.plot(block, varieties, fertilizer, Yield))
gla = model$gl.a
glb = model$gl.b
glc = model$gl.c
Ea = model$Ea
Eb = model$Eb
Ec = model$Ec
out1 = with(df,LSD.test(Yield,varieties,gla, Ea, console = TRUE))
out2 = with(df,LSD.test(Yield,fertilizer,glb, Eb, console = TRUE))
out3 = with(df,LSD.test(Yield,fertilizer:varieties, glc, Ec, console = TRUE))
```

8.3 Strip-plot Python:

Η ανάλυση των δεδομένων που προκύπτουν από έναν πειραματικό σχεδιασμό strip-plot γίνεται με την βοήθεια των βιβλιοθηκών pandas , numpy ,math και scipy.

```
import pandas as pd
import numpy as np
import math
import scipy.stats
from scipy.stats import studentized_range
```

Αρχικά γίνεται εισαγωγή των δεδομένων μέσα από την εντολή :

```
data = pd.read_excel('filename.xlsx')
```

Δυστυχώς στην Python δεν προσφέρεται κάποια βιβλιοθήκη για τον σχηματισμό του πίνακα Anova.

Αντ' αυτού οι υπολογισμοί των δεδομένων για τον σχηματισμό του πίνακα Anova μπορούν να γίνουν με τον εξής τρόπο:

Υπολογισμός του Grand Total της μεταβλητής απόκρισης μπορεί να γίνει με την παρακάτω εντολή:

```
GT = sum(data.Yield)
```

Όπου "Yield" η μεταβλητή απόκρισης.

Στην συνέχεια είναι απαραίτητο να σχηματιστούν μεταβλητές με τα επίπεδα κάθε μεταβλητής.

```
#Levels of Factor A
v = len(np.unique(data.varieties))
#Levels of Factor B
f = len(np.unique(data.fertilizer))
#Levels of Blocks
b = len(np.unique(data.block))
```

Με αυτόν τον τρόπο η μεταβλητή v εμπεριέχει τον αριθμό 3 δηλαδή όσα και τα επίπεδα των ποικιλιών.Παρομοίως οι μεταβλητές f και b αντιπροσωπεύουν των αριθμό των επιπέδων του λιπάσματος και των block.
Με αυτά τα δεδομένα μπορεί να γίνει ο υπολογισμός του Correction Factor αλλά και το συνολικό άθροισμα τετραγώνων.

```
#Correction Factor
CF = (GT**2)/(v*f*b)

#Tota Sum Of Square
TSS = sum(data.Yield**2) -CF
```

Το επόμενο βήμα είναι ο υπολογισμός του αθροίσματος των τετραγώνων κάθε μεταβλητής ξεχωριστά.
Για τον υπολογισμό του αθροίσματος των τετραγώνων των block αρχικά σχηματίζεται μια κενή λίστα με τέσσερα στοιχεία, όσα δηλαδή και τα επίπεδα των block.Εάν τα block είχαν 5 επίπεδα τότε η λίστα που θα χρειαζόταν θα είχε 5 στοιχεία.

```
rss = [0,0,0,0]
```

Τα στοιχεία της λίστας θα αντικατασταθούν με το άθροισμα της απόδοσης που αντιστοιχεί σε κάθε block.Για το συγκεκριμένο παράδειγμα η λίστα θα είναι rss = [80.3, 88.6, 93.0, 97.7] διότι στο block 1 η συνολική απόδοση ήταν 80.3.Αυτό μπορεί να γίνει αυτόματα με τον παρακάτω κώδικα.

```
rss = [0,0,0,0]
for y in range(b):
    for x in range(len(data.block)):
        if data.block[x] == y+1:
            rss[y] = rss[y] + data.Yield[x]
```

Μετά τον σχηματισμό της λίστας με τον παραπάνω τρόπο το άθροισμα τετραγώνων των block υπολογίζεται ως εξής

```
SSR = 0
for i in rss:
    SSR = SSR + i**2

SSR = (SSR /(v*f)) - CF
```

Όπου η μεταβλητή SSR είναι το άθροισμα τετραγώνων των block.
Με παρόμοιο τρόπο μπορούν να υπολογιστούν τα αθροίσματα τετραγώνων για το λίπασμα και την ποικιλία.

Άθροισμα τετραγώνων για ποικιλίες

```
vb = [0,0,0]
for y in range(v):
    for x in range(len(data.varieties)):
                    if data.varieties[x] == f'V{y+1}':
                        vb[y] = vb[y] + data.Yield[x]
```

```
SSA = 0
for i in vb:
   SSA = SSA + i**2

SSA = (SSA/(f*b))-CF
```

Επί της ουσίας διαβάζονται όλες οι τιμές της μεταβλητής απόκρισης και εάν η τιμή αυτή αντιστοιχεί σε ποικιλία με το όνομα "V1" τότε προστίθεται στο πρώτο στοιχείο της λίστας vb.Ο διαχωρισμός των τιμών γίνεται με την εντολή `if data.variety[x] == f'V{y+1}':` .Εάν οι ποικιλίες είχαν διαφορετικό όνομα, για παράδειγμα 1,2,3 αντί για V1,V2,V3, τότε η παραπάνω εντολή θα γινόταν `if data.variety[x] == y+1` και αυτό γιατί η μεταβλητή y θα πάρει τιμές 0,1,2.Εν τέλη η μεταβλητή "SSA" αποτελεί το άθροισμα τετραγώνων των ποικιλιών.Η ίδια μεθοδολογία ακολουθείται και για το άθροισμα τετραγώνων για το λίπασμα:

```
ssb = [0,0,0,0]
for y in range(f):
   for x in range(len(data.Yield)):
      if data.fertilizer[x] == f'F{y+1}':
         ssb[y] = ssb[y] + data.Yield[x]

SSB = 0
for i in ssb:
   SSB = SSB + i**2

SSB = (SSB/(v*b)) -CF
```

Στη συνέχεια γίνεται ο υπολογισμός του αθροίσματος τετραγώνων του Error a των ποικιλιών.Εδώ δεν χρειάζεται ο σχηματισμός λίστας αλλά δύο διαφορετικών μεταβλητών "va" και "SSEa".

```
SSEa = 0
va = 0
for u in range(v):
   for y in range(b+1):
      SSEa = SSEa + va**2
      va = 0
      for x in range(len(data.Yield)):
         if data.varieties[x] == f'V{u+1}' and data.block[x] == y+1:
```

```
            va = va + data.Yield[x]

Ea = (SSEa/f)-CF-SSA-SSR
```

Η ίδια μεθοδολογία ακολουθείται και για το άθροισμα τετραγώνων του Error b για το λίπασμα αλλά και για την αλληλεπίδραση μεταξύ των δύο βασικών παραγόντων(λίπασμα και ποικιλίες).

Error b:

```
SSEb = 0
vbb = 0
for u in range(f):
    for y in range(b+1):
        SSEb = SSEb + vbb**2
        vbb = 0
        for x in range(len(data.Yield)):
            if data.fertilizer[x] == f'F{u+1}' and data.block[x] == y+1:
                vbb = vbb + data.Yield[x]

Eb = (SSEb/v)-CF-SSB-SSR
```

Αλληλεπίδραση βασικών παραγόντων:

```
SSVF = 0
vvf = 0
for y in range(f+1):
    for u in range(v):
        SSVF = SSVF + vvf**2
        vvf = 0
        for x in range(len(data.Yield)):
            if data.fertilizer[x] == f'F{y+1}':
                if data.varieties[x] == f'V{u+1}':
                    vvf = vvf + data.Yield[x]

SSVF = (SSVF/b) - CF - SSA - SSB
```

Τέλος για το Error c χρησιμοποιείται η εντολή:

```
Ec = TSS - SSR - SSA - SSB - SSVF - Ea - Eb
```

Οι βαθμοί ελευθερίας μπορούν εύκολα να υπολογιστούν μιας και είδη έχουν σχηματιστεί μεταβλητές με τα επίπεδα κάθε παράγοντα

```
#Factor A(varieties)
DFv = v - 1
#Factor B(Fertilizer)
DFf = f - 1
#Block
DFb = b - 1
#Interaction
DFvf = DFv*DFf
#Error a
DFEa = DFv*DFb
#Error b
DFEb = DFb*DFf
#Error c
DFEc = DFv*DFb*DFf
```

Μετά τον υπολογισμό των βαθμών ελευθερίας τα μέσα τετράγωνα (Mean_sq) υπολογίζονται με τον εξής απλό τρόπο.

```
#Blcok
MSB = SSR / DFb
#Factor A(varieties)
MSV  = SSA / DFv
#Error A
MSErrora = Ea / DFEa
#Factor B(fertilizer)
MSF = SSB / DFf
#Error B
MSErrorb = Eb / DFEb
#Interaction
MSvf = SSVF / DFvf
#Error c
MSErrorc = Ec / DFEc
```

Τέλος για τον σχηματισμό του πίνακα Anova είναι αναγκαίος ο υπολογισμός των F-values και η εύρεση της κρίσιμης τιμής F.H F value υπολογίζεται με την διαίρεση των μέσων τετραγώνων του παράγοντα με μέσο τετράγωνο του αντίστοιχου λάθους του παράγοντα όπως φαίνεται παρακάτω.

```
#Fvalue
```

```
FA = MSV / MSErrora
FB = MSF / MSErrorb
FAB = MSvf / MSErrorc
```

Η κρίσιμη τιμή F υπολογίζεται με την εντολή scipy.stats.f.ppf(1-α, df1, df2) όπου df1 είναι οι βαθμοί ελευθερίας του βασικού παράγοντα και df2 οι βαθμοί ελευθερίας του λάθους που αντιστοιχεί στον κάθε παράγοντα.

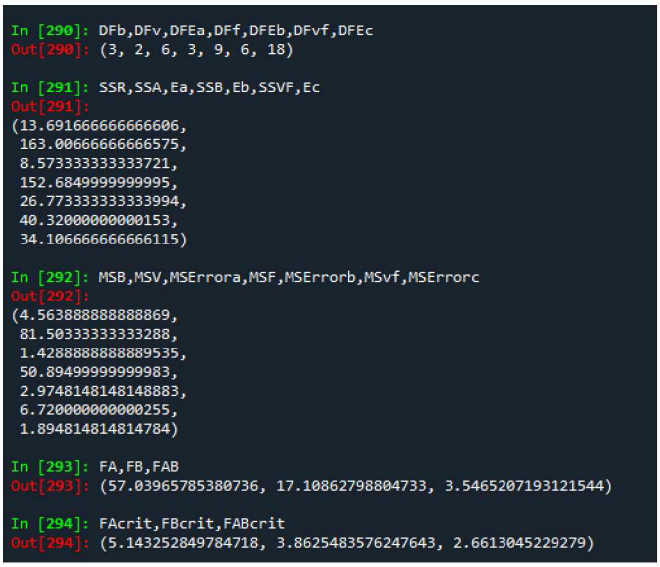
```
#Fcrit
FAcrit =  scipy.stats.f.ppf(0.95, DFv, DFEa)
FBcrit = scipy.stats.f.ppf(0.95, DFf, DFEb)
Fvfcrit = scipy.stats.f.ppf(0.95, DFvf, DFEc)
```

Μετά από όλα τα παραπάνω βήματα έχουν σχηματιστεί οι μεταβλητές που αποτελούν τον πίνακα Anova όπως φαίνεται παρακάτω.

SV	Df	Sum squares	Mean Squares	F value	F value 5%
Block	DFb	SSR	MSB		
varieties	DFv	SSA	MSV	FA	FAcrit
Ea	DFEa	Ea	MSErrora		
fertilizer	DFf	SSB	MSF	FB	FBcrit
Eb	DFEb	Eb	MSErrorb		
varieties:fertilizer	DFvf	SSVF	MSvf	FAB	FABcrit
Ec	DFEc	Ec	MSErrorc		

(Πίνακας 8.2)Πίνακας Anova πριν την αντικατάσταση μεταβλητών.

Αρκεί να γίνει η αντικατάσταση τους.



(Εικόνα 8.18)Στοιχεία πίνακα Anova.

SV	Df	Sum squares	Mean Squares	F value	F value 5%
Block	3	13.69	4.564		
varieties	2	163.007	81.503	57.0397	5.1432
Ea	6	8.573	1.429		
fertilizer	3	152.685	50.895	17.1086	3.8625
Eb	9	26.773	2.975		
varieties:fertilizer	6	40.32	6.72	3.5465	2.6613
Ec	18	34.107	1.895		

(Πίνακας 8.1)Πίνακας Anova μετά από αντικατάσταση μεταβλητών.

Από τον πίνακα Anova παρατηρείται ότι υπάρχει σημαντική διαφορά για την αλληλεπίδραση των δύο βασικών παραγόντων.

Post-hoc testing

Για την διάκριση των σημαντικά διαφορετικών μεταβλητών γίνεται test σύγκρισης μέσων όρων με την μέθοδο Tukey HSD.

Από τον πίνακα Anova παρατηρείται ότι ο όρος αλληλεπίδρασης (varieties :fertilizer) είναι σημαντικός και συνεπώς δεν υπάρχει λόγος εξέτασης της ξεχωριστής επίδραση των δύο κύριων μεταβλητών , δηλαδή των ποικιλιών και του λιπάσματος.Παρόλα αυτά ο τρόπος εφαρμογής της μεθόδου Tukey για τους δύο βασικούς παράγοντες αναλύεται παρακάτω. Για την σύγκριση μέσων όρων του vertical factor (varieties) θα χρησιμοποιηθεί ο βαθμός ελευθερίας και το τετράγωνο του μέσου όρου του Ea.

Για τον horizontal factor(fertilizer) θα χρησιμοποιηθεί ο βαθμός ελευθερίας και το τετράγωνο του μέσου όρου του Eb.Τέλος για τον παράγοντα αλληλεπίδρασης θα χρησιμοποιηθεί ο βαθμός ελευθερίας και το τετράγωνο του μέσου όρου του Ec[24].

Για την σύγκριση μέσων όρων για τις ποικιλίες αρχικά γίνεται ο υπολογισμός της κρίσιμης τιμής T.

$T = q \sqrt{(MSE/n)}$[19]

Η κρίσιμη τιμή q υπολογίζεται με την εντολή studentized_range.ppf(1-α, k, df) όπου k είναι ο αριθμός των μέσων όρων και df είναι οι βαθμοί ελευθερίας του σφάλματος.Για το συγκεκριμένο παράδειγμα η εντολή αυτή γράφεται ως:

```
q1 = studentized_range.ppf(0.95, v, DFEa)
```

Η τιμή του "n" για τον τύπο "T = q √(MSE/n)" προκύπτει ως εξής:

```
nv1 = 0
for x in range(len(data.Yield)):
   if data.varieties[x] == 'V1':
      nv1 = nv1 + 1
```

Τέλος η κρίσιμη τιμή T υπολογίζεται με την εντολή:

```
Tcrit1 = q1*math.sqrt(MSErrora/nv1)
```

Ακολουθεί ο υπολογισμός των μέσων όρων για κάθε ποικιλία:

```
vmean=[0,0,0]
for y in range(v):
   for x in range(len(data.Yield)):
                  if data.varieties[x] == f'V{y+1}':
                     vmean[y] = vmean[y] + data.Yield[x]

vmean = np.array(vmean)
vmean = vmean/nv1
```

Εδώ η λίστα "vmean" αποτελείται από τρία στοιχεία.Το κάθε στοιχείο αποτελεί τον μέσο όρο μιας ποικιλίας.Το πρώτο στοιχείο αποτελεί τον μέσο όρο της ποικιλίας V1 , το δεύτερο της ποικιλίας V2 και το τρίτο της ποικιλίας V3.Απομένει η μεταξύ τους σύγκριση.

```
res = np.unique(data.varieties)
z = 0
for y in range(len(np.unique(data.varieties))):
   for x in range(y+1,len(np.unique(data.varieties))):
      print(res[y],"-",res[x],"=",abs(vmean[y] - vmean[x]), "|Critical value =", Tcrit1,
"|Significant at α = 5%",abs(vmean[y] - vmean[x])>Tcrit1 )
      z = z + 1
```

```
In [41]: res = np.unique(data.varieties)
    ...:
    ...:
    ...: z = 0
    ...: for y in range(len(np.unique(data.varieties))):
    ...:     for x in range(y+1,len(np.unique(data.varieties))):
    ...:         print(res[y],"-",res[x],"=",abs(vmean[y] - vmean[x]), "|Critical value =", Tcrit1, "|Significant at α =
5%",abs(vmean[y] - vmean[x])>Tcrit1 )
    ...:         z = z + 1
    ...:
V1 - V2 = 0.4750000000000014 |Critical value = 1.2967267004933718 |Significant at α = 5% False
V1 - V3 = 4.125000000000001 |Critical value = 1.2967267004933718 |Significant at α = 5% True
V2 - V3 = 3.6499999999999995 |Critical value = 1.2967267004933718 |Significant at α = 5% True
```

(Εικόνα 8.19) Σύγκριση μέσων όρων ποικιλιών.

Από το αποτέλεσμα φαίνεται πως η ποικιλία V3 είναι η λιγότερο παραγωγική ενώ δεν υπάρχει στατιστικά σημαντική διαφορά μεταξύ των V1,V2.

.

Με τα ίδια βήματα πραγματοποιείται και η σύγκριση μέσων όρων των λιπασμάτων.

```
q2 = studentized_range.ppf(0.95, f, DFEb)
nf1 = 0
for x in range(len(data.Yield)):
    if data.fertilizer[x] == 'F1':
        nf1 = nf1 + 1


Tcrit2 = q2*math.sqrt(MSErrorb/nf1)
fmean=[0,0,0,0]
for y in range(f):
    for x in range(len(data.Yield)):
                    if data.fertilizer[x] == f'F{y+1}':
                        fmean[y] = fmean[y] + data.Yield[x]


fmean = np.array(fmean)
fmean = fmean/nf1
res = np.unique(data.fertilizer)
z = 0
```

```
for y in range(len(np.unique(data.fertilizer))):
   for x in range(y+1,len(np.unique(data.fertilizer))):
      print(res[y],"-",res[x],"=",abs(fmean[y] - fmean[x]), "|Critical value =", Tcrit2,"|Significant
at α = 5%",abs(fmean[y] - fmean[x])>Tcrit2)
      z = z + 1
```

```
In [42]: res = np.unique(data.fertilizer)
    ...: z = 0
    ...: for y in range(len(np.unique(data.fertilizer))):
    ...:     for x in range(y+1,len(np.unique(data.fertilizer))):
    ...:         print(res[y],"-",res[x],"=",abs(fmean[y] - fmean[x]), "|Critical value =", Tcrit2,"|Significant at α =
5%",abs(fmean[y] - fmean[x])>Tcrit2)
    ...:         z = z + 1
F1 - F2 = 0.9333333333333336 |Critical value = 2.1981596664909726 |Significant at α = 5% False
F1 - F3 = 1.0750000000000002 |Critical value = 2.1981596664909726 |Significant at α = 5% False
F1 - F4 = 3.825 |Critical value = 2.1981596664909726 |Significant at α = 5% True
F2 - F3 = 2.0083333333333337 |Critical value = 2.1981596664909726 |Significant at α = 5% False
F2 - F4 = 4.758333333333334 |Critical value = 2.1981596664909726 |Significant at α = 5% True
F3 - F4 = 2.75 |Critical value = 2.1981596664909726 |Significant at α = 5% True
```

(Εικόνα 8.20) Σύγκριση μέσων όρων λιπασμάτων.

Από το αποτέλεσμα φαίνεται πως το επίπεδο λιπάσματος F4 επιφέρει την μικρότερη απόδοση ενώ δεν υπάρχει στατιστικά σημαντική διαφορά μεταξύ των άλλων επιπέδων

(Εικόνα 8.23)Υπολογισμός συντελεστών διακύμανσης λιπασμάτων.

Επίσης με τα ίδια βήματα πραγματοποιείται και η σύγκριση μέσων όρων των του παράγοντα αλληλεπίδρασης. Η διαφορά εδώ είναι πως για την κρίσιμη τιμή T η μεταβλητή “n” είναι ίση με τον αριθμό που εμφανίζεται κάθε ζεύγος FV, για παράδειγμα F1V1 ή F2V3.Μιας και όλα τα ζεύγη εμφανίζονται ίσες φορές στο πείραμα, αρκεί ο υπολογισμός του αριθμού εμφάνισης ενός ζεύγους.

```
nvf1 = 0
for x in range(len(data.Yield)):
    if data.fertilizer[x] == 'F1' and data.varieties[x] == 'V1':
       nvf1 = nvf1 + 1
```

Τέλος πραγματοποιείται η σύγκριση μέσων όρων με τον παρακάτω κώδικα:

```
q3 = studentized_range.ppf(0.95, f*v, DFEc)
Tcrit3 = q3*math.sqrt(MSErrorc/nvf1)
vfmean = [0]*12
```

```
for y in range(f):
    for u in range(v):
        for x in range(len(data.Yield)):
            if data.fertilizer[x] == f'F{y+1}':
                if data.varieties[x] == f'V{u+1}':
                    vfmean[3*y+u] = vfmean[3*y+u] + data.Yield[x]
vfmean = np.array(vfmean)
vfmean = vfmean / nvf1
#Absolute Mean Comparison
res = np.unique(data.fertilizer + data.varieties)
z = 0
for y in range(len(res)):
    for x in range(y+1,len(res)):
            print(res[y],"-",res[x],"=", abs(vfmean[y] - vfmean[x]), "|Critical value =", Tcrit3,
"|Significant at α = 5%",abs(vfmean[y] - vfmean[x])>Tcrit3)
            z = z + 1
```

```
In [46]: res = np.unique(data.fertilizer + data.varieties)
    ...: z = 0
    ...: for y in range(len(res)):
    ...:     for x in range(y+1,len(res)):
    ...:             print(res[y],"-",res[x],"=", abs(vfmean[y] - vfmean[x]), "|Critical value =", Tcrit3, "|Significant at α
= 5%",abs(vfmean[y] - vfmean[x])>Tcrit3)
    ...:             z = z + 1
F1V1 - F1V2 = 1.8000000000000007 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V1 - F1V3 = 6.275000000000001 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V1 - F2V1 = 1.450000000000001 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V1 - F2V2 = 1.4000000000000004 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V1 - F2V3 = 2.4250000000000007 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V1 - F3V1 = 2.1500000000000004 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V1 - F3V2 = 1.75 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V1 - F3V3 = 7.4 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V1 - F4V1 = 5.000000000000001 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V1 - F4V2 = 5.550000000000001 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V1 - F4V3 = 9.0 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V2 - F1V3 = 4.4750000000000005 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V2 - F2V1 = 0.34999999999999964 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V2 - F2V2 = 0.40000000000000036 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V2 - F2V3 = 0.625 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V2 - F3V1 = 0.34999999999999964 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V2 - F3V2 = 0.05000000000000071 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V2 - F3V3 = 5.6 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V2 - F4V1 = 3.2 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V2 - F4V2 = 3.75 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V2 - F4V3 = 7.2 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V3 - F2V1 = 4.825 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V3 - F2V2 = 4.875000000000001 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V3 - F2V3 = 3.8500000000000005 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V3 - F3V1 = 4.125000000000001 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V3 - F3V2 = 4.525000000000001 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F1V3 - F3V3 = 1.1249999999999996 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V3 - F4V1 = 1.2750000000000004 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V3 - F4V2 = 0.7250000000000005 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F1V3 - F4V3 = 2.7249999999999996 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V1 - F2V2 = 0.05000000000000071 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V1 - F2V3 = 0.9749999999999996 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V1 - F3V1 = 0.6999999999999993 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V1 - F3V2 = 0.29999999999999893 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V1 - F3V3 = 5.949999999999999 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F2V1 - F4V1 = 3.55 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V1 - F4V2 = 4.1 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F2V1 - F4V3 = 7.55 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F2V2 - F2V3 = 1.0250000000000004 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V2 - F3V1 = 0.75 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V2 - F3V2 = 0.34999999999999964 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V2 - F3V3 = 6.0 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F2V2 - F4V1 = 3.6000000000000005 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V2 - F4V2 = 4.15 |Critical value = 3.6246668096273607 |Significant at α = 5% True
```

(Εικόνα 8.21.Α) Σύγκριση μέσων όρων αλληλεπίδρασης.

```
F2V2 - F4V3 = 7.6000000000000005 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F2V3 - F3V1 = 0.27500000000000036 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V3 - F3V2 = 0.6750000000000007 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V3 - F3V3 = 4.975 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F2V3 - F4V1 = 2.575 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V3 - F4V2 = 3.125 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F2V3 - F4V3 = 6.575 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F3V1 - F3V2 = 0.40000000000000036 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F3V1 - F3V3 = 5.25 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F3V1 - F4V1 = 2.8500000000000005 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F3V1 - F4V2 = 3.4000000000000004 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F3V1 - F4V3 = 6.8500000000000005 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F3V2 - F3V3 = 5.65 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F3V2 - F4V1 = 3.250000000000001 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F3V2 - F4V2 = 3.800000000000007 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F3V2 - F4V3 = 7.250000000000001 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F3V3 - F4V1 = 2.4 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F3V3 - F4V2 = 1.85 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F3V3 - F4V3 = 1.6 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F4V1 - F4V2 = 0.5499999999999998 |Critical value = 3.6246668096273607 |Significant at α = 5% False
F4V1 - F4V3 = 4.0 |Critical value = 3.6246668096273607 |Significant at α = 5% True
F4V2 - F4V3 = 3.45 |Critical value = 3.6246668096273607 |Significant at α = 5% False
```

(Εικόνα 8.22.Β) Σύγκριση μέσων όρων αλληλεπίδρασης.

Σύνοψη κώδικα

```
import pandas as pd
import numpy as np
import math
import scipy.stats
from scipy.stats import studentized_range
data = pd.read_excel('strip1.xlsx')

#Grant Total
GT = sum(data.Yield)
#Levels of Factor A
v = len(np.unique(data.varieties))
#Levels of Factor B
f = len(np.unique(data.fertilizer))
#Levels of Blocks
b = len(np.unique(data.block))
#Correction Factor
CF = (GT**2)/(v*f*b)
#Tota Sum Of Square
TSS = sum(data.Yield**2) - CF


#Replication sum of square
```

```
rss = [0,0,0,0]
for y in range(b):
    for x in range(len(data.block)):
        if data.block[x] == y+1:
            rss[y] = rss[y] + data.Yield[x]


SSR = 0
for i in rss:
    SSR = SSR + i**2

SSR = (SSR /(v*f)) - CF


#Sum of Square due to Factor A(varieties)
vb = [0,0,0]
for y in range(v):
    for x in range(len(data.varieties)):
                    if data.varieties[x] == f'V{y+1}':
                        vb[y] = vb[y] + data.Yield[x]

SSA = 0
for i in vb:
    SSA = SSA + i**2

SSA = (SSA/(f*b)) - CF

#Error A Sum of square
SSEa = 0
va = 0
for u in range(v):
    for y in range(b+1):
        SSEa = SSEa + va**2
        va = 0
        for x in range(len(data.Yield)):
            if data.varieties[x] == f'V{u+1}' and data.block[x] == y+1:
                va = va + data.Yield[x]
```

```
Ea = (SSEa/f)-CF-SSA-SSR

#Sum of Square due to Factor B (fertilizer)

ssb = [0,0,0,0]
for y in range(f):
    for x in range(len(data.Yield)):
        if data.fertilizer[x] == f'F{y+1}':
            ssb[y] = ssb[y] + data.Yield[x]


SSB = 0
for i in ssb:
    SSB = SSB + i**2

SSB = (SSB/(v*b)) - CF
#Error B Sum of square
SSEb = 0
vbb = 0
for u in range(f):
    for y in range(b+1):
        SSEb = SSEb + vbb**2
        vbb = 0
        for x in range(len(data.Yield)):
            if data.fertilizer[x] == f'F{u+1}' and data.block[x] == y+1:
                vbb = vbb + data.Yield[x]



Eb = (SSEb/v)-CF-SSB-SSR

#Sum of Square due to Interaction
SSVF = 0
vvf = 0

for y in range(f+1):
    for u in range(v):
        SSVF = SSVF + vvf**2
        vvf = 0
```

```
        for x in range(len(data.Yield)):
            if data.fertilizer[x] == f'F{y+1}':
                if data.varieties[x] == f'V{u+1}':
                    vvf = vvf + data.Yield[x]



SSVF = (SSVF/b) - CF - SSA - SSB




#Error C Sum of square

Ec = TSS - SSR - SSA - SSB - SSVF - Ea - Eb


# Degrees of Freedom
#Factor A(varieties)
DFv = v - 1
#Factor B(Fertilizer)
DFf = f - 1
#Block
DFb = b - 1
#Interaction
DFvf = DFv*DFf
#Error a
DFEa = DFv*DFb
#Error b
DFEb = DFb*DFf
#Error c
DFEc = DFv*DFb*DFf



#Mean_Sq
#Block
MSB = SSR / DFb
#Factor A(varieties)
```

```
MSV  = SSA / DFv
#Error A
MSErrora = Ea / DFEa
#Factor B(fertilizer)
MSF = SSB / DFf
#Error B
MSErrorb = Eb / DFEb
#Interaction
MSvf = SSVF / DFvf
#Error c
MSErrorc = Ec / DFEc


#Fvalue
FA = MSV / MSErrora
FB = MSF / MSErrorb
FAB = MSvf / MSErrorc


#Fcrit
FAcrit =  scipy.stats.f.ppf(0.95, DFv, DFEa)
FBcrit = scipy.stats.f.ppf(0.95, DFf, DFEb)
FABcrit = scipy.stats.f.ppf(0.95, DFvf, DFEc)

#########TUKEY#########
#Variety
q1 = studentized_range.ppf(0.95, v, DFEa)
nv1 = 0
for x in range(len(data.Yield)):
   if data.varieties[x] == 'V1':
      nv1 = nv1 + 1

Tcrit1 = q1*math.sqrt(MSErrora/nv1)

vmean=[0,0,0]
for y in range(v):
   for x in range(len(data.Yield)):
                  if data.varieties[x] == f'V{y+1}':
                     vmean[y] = vmean[y] + data.Yield[x]
```

```
vmean = np.array(vmean)
vmean = vmean/nv1
#Absolute Mean Comparison
res = np.unique(data.varieties)

z = 0
for y in range(len(np.unique(data.varieties))):
   for x in range(y+1,len(np.unique(data.varieties))):
      print(res[y],"-",res[x],"=",abs(vmean[y] - vmean[x]), "|Critical value =", Tcrit1,
"|Significant at α = 5%",abs(vmean[y] - vmean[x])>Tcrit1 )
      z = z + 1

import statistics
A = data.query('varieties == "V1"')['Yield']
B = data.query('varieties == "V2"')['Yield']
print("CV1={}".format(statistics.stdev(A)/statistics.mean(A)))
print("CV2={}".format(statistics.stdev(B)/statistics.mean(B)))

#Fertilizer
q2 = studentized_range.ppf(0.95, f, DFEb)
nf1 = 0
for x in range(len(data.Yield)):
   if data.fertilizer[x] == 'F1':
      nf1 = nf1 + 1


Tcrit2 = q2*math.sqrt(MSErrorb/nf1)
fmean=[0,0,0,0]
for y in range(f):
   for x in range(len(data.Yield)):
                  if data.fertilizer[x] == f'F{y+1}':
                     fmean[y] = fmean[y] + data.Yield[x]

fmean = np.array(fmean)
fmean = fmean/nf1

#Absolute Mean Comparison
res = np.unique(data.fertilizer)
z = 0
```

```
for y in range(len(np.unique(data.fertilizer))):
   for x in range(y+1,len(np.unique(data.fertilizer))):
      print(res[y],"-",res[x],"=",abs(fmean[y] - fmean[x]), "|Critical value =", Tcrit2,"|Significant
at α = 5%",abs(fmean[y] - fmean[x])>Tcrit2)
      z = z + 1



#Interaction

nvf1 = 0
for x in range(len(data.Yield)):
    if data.fertilizer[x] == 'F1' and data.varieties[x] == 'V1':
      nvf1 = nvf1 + 1

q3 = studentized_range.ppf(0.95, f*v, DFEc)
Tcrit3 = q3*math.sqrt(MSErrorc/nvf1)



vfmean = [0]*12
for y in range(f):
   for u in range(v):
      for x in range(len(data.Yield)):
         if data.fertilizer[x] == f'F{y+1}':
            if data.varieties[x] == f'V{u+1}':
               vfmean[3*y+u] = vfmean[3*y+u] + data.Yield[x]


vfmean = np.array(vfmean)
vfmean = vfmean / nvf1

#Absolute Mean Comparison
res = np.unique(data.fertilizer + data.varieties)
z = 0
for y in range(len(res)):
   for x in range(y+1,len(res)):
         print(res[y],"-",res[x],"=", abs(vfmean[y] - vfmean[x]), "|Critical value =", Tcrit3,
"|Significant at α = 5%",abs(vfmean[y] - vfmean[x])>Tcrit3)
         z = z + 1
```

9. Ανάλυση συνδιακύμανσης (Ancova)

Η ανάλυση συνδιακύμανσης ή συμμεταβλητή ανάλυση (Ancova) χρησιμοποιείται για την στατιστική ανάλυση πειραμάτων στα οποία η συνεχής μεταβλητή δεν μπορεί να ελεγχθεί αλλά μπορεί μόνο να παρατηρηθεί σε γραμμική συσχέτιση με την μεταβλητή απόκρισης[25]. Η Ancova είναι ένας συνδυασμός της απλής ανάλυσης διακύμανσης(Anova) και της ανάλυση παλινδρόμησης (Regression)[26]. Αποτελείται από δύο είδη ανεξάρτητων μεταβλητών, τις κατηγορικές μεταβλητές και τις συνεχείς μεταβλητές (covariates). Εάν δεν υπάρχουν συνεχείς μεταβλητές τότε γίνεται λόγος για Anova ενώ εάν δεν υπάρχουν κατηγορικές μεταβλητές τότε γίνεται λόγος για ανάλυση παλινδρόμησης [27]. Με την Ancova γίνεται ο υπολογισμός της διαφοροποίησης μέσων όρων της μεταβλητής απόκρισης μεταξύ των κατηγορικών μεταβλητών και δεδομένης της επίδρασης των συμμεταβλητών(covariates). Η διαδικασία αυτή γίνεται με σκοπό την μείωση της πηγής διασποράς (error variance) εντός των ομάδων. Οι διορθωμένοι μέσοι όροι που προκύπτουν ονομάζονται προσαρμοσμένοι μέσοι όροι (adjusted means) [28].Η Ancova χρησιμοποιείται συνήθως όταν υπάρχουν διαφορές μεταξύ των βασικών ομάδων[29], αν και μπορεί να εφαρμοστεί και σε αναλύσεις protest/posttest όταν η παλινδρόμηση στη μέση τιμή επηρεάζει τη post test μέτρηση [30].Η τεχνική Ancova εφαρμόζεται ακόμα και σε μη πειραματικές έρευνες (πχ. Επισκόπηση) αλλά και σε ψευδο-πειράματα στα οποία οι συμμετέχοντες δεν μπορούν να τυχαιοποιηθούν επαρκώς, αν και αυτή η εφαρμογή της Ancova δεν συνιστάται πάντα[31].

9.1 Παράδειγμα εφαρμογής Ancova

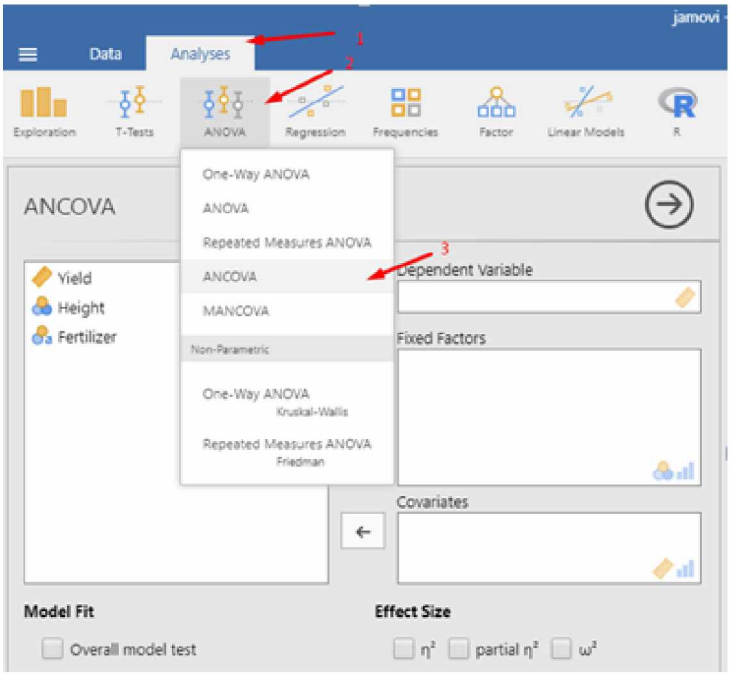
Σε ένα πείραμα αγρού εξετάζεται η επίδραση τριών λιπασμάτων (C,S,F) στην απόδοση σε κιλά 30 φυτών μουσμουλιάς. Η εφαρμογή των λιπασμάτων έγινε σύμφωνα με ένα πλήρως τυχαιοποιημένο σχέδιο. Μαζί με την αρχική εφαρμογή των λιπασμάτων έγινε μέτρηση και καταγραφή του αρχικού ύψους κάθε φυτού μίας και το αρχικό ύψος αποτελεί μια μεταβλητή που ίσως επηρεάζει την απόδοση. Στο συγκεκριμένο παράδειγμα η μεταβλητή απόκρισης είναι η απόδοση κάθε φυτού. Το ύψος αποτελεί έναν παράγοντα που δεν μπορεί να ελεγχθεί αλλά ίσως επηρεάσει την απόδοση, συνεπώς το ύψος αποτελεί την συμμεταβλητή του πειράματος (covariate).Τέλος κατηγορική μεταβλητή αποτελεί το κάθε λίπασμα(C,S,F).Τα δεδομένα του πειράματος φαίνονται την εικόνα 9.1. Παρακάτω περιγράφεται ο τρόπος εφαρμογής της τεχνικής Ancova με βάση το παραπάνω πείραμα. Η ανάλυση αυτή θα γίνει μέσω του στατιστικού λογισμικού Jamovi,της γλώσσας R, αλλά και μέσω Python.

Yield	Height	Fertilizer
12.2	45	C
12.4	52	C
11.9	42	C
11.3	35	C
11.8	40	C
12.1	48	C
13.1	60	C
12.7	61	C
12.4	50	C
11.4	33	C
16.6	63	S
15.8	50	S
16.5	63	S
15	33	S
15.4	38	S
15.6	45	S
15.8	50	S
15.8	48	S
16	50	S
15.8	49	S
9.5	52	F
9.5	54	F
9.6	58	F
8.8	45	F
9.5	57	F
9.8	62	F
9.1	52	F
10.3	67	F
9.5	55	F
8.5	40	F

(Εικόνα 9.1)Δεδομένα πειράματος προς Ancova ανάλυση.

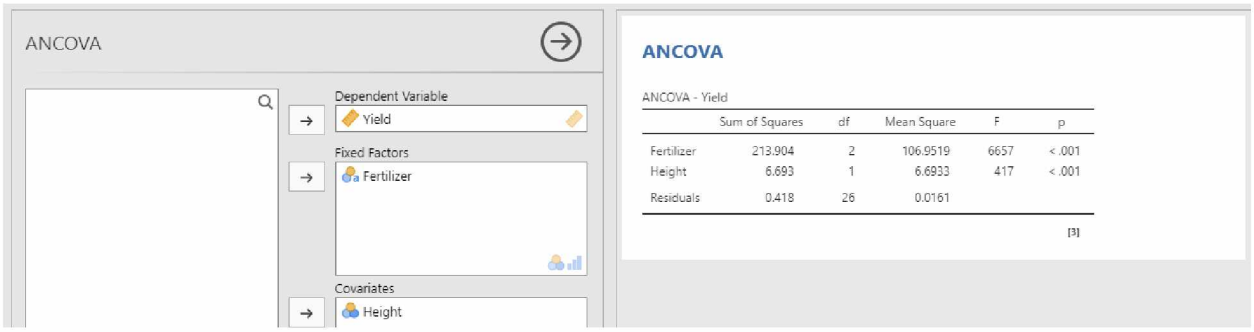
9.2 Ancova Jamovi:

Με την εισαγωγή του αρχείου excel όπως αυτό της εικόνας 9.1 στο jamovi μπορεί να εκτελεστεί το Ancova test.



(Εικόνα 9.2) Επιλογή Ancova.

Ως dependent variable τίθεται η μεταβλητή απόκρισης δηλαδή η απόδοση(Yield) ενώ ως Fixed Factors η κατηγορική μεταβλητή, δηλαδή το λίπασμα(Fertilizer). Τέλος συμμεταβλητή του πειράματος(Covariate) αποτελεί το ύψος των φυτών(Height).Μετά την κατηγοριοποίηση της κάθε μεταβλητής, γίνεται ο σχηματισμός του πίνακα Ancova.

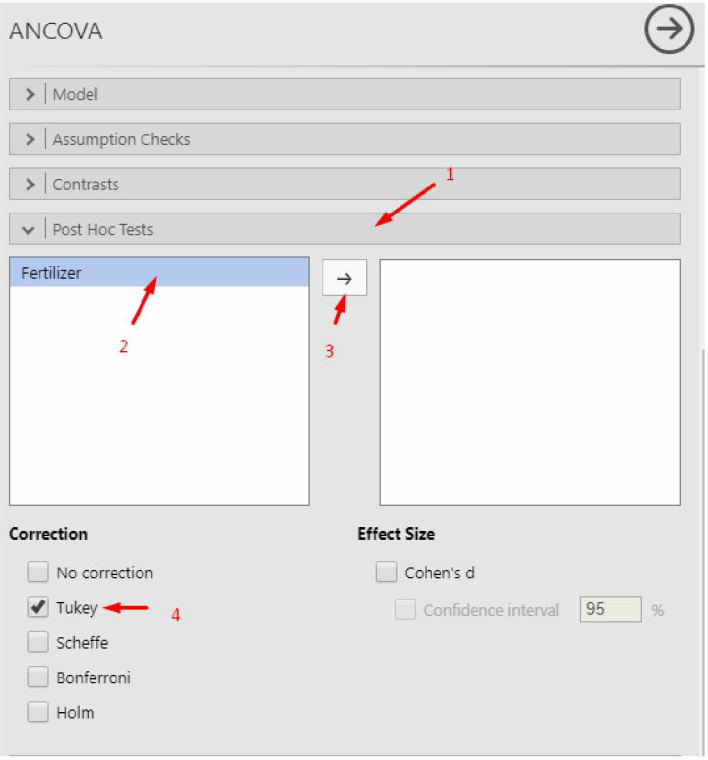


(Εικόνα 9.3) Σχηματισμός πίνακα Ancova σε Jamovi.

Από τον πίνακα Ancova φαίνεται πως ο τύπος λιπάσματος αλλά και το αρχικό ύψος των φυτών έχουν εξαιρετικά σημαντική επίδραση στην τελική απόδοση των φυτών($p < 0.01$).

Post-hoc testing

Για την διάκριση του πιο αποτελεσματικού λιπάσματος γίνεται test σύγκρισης μέσων όρων με την μέθοδο Tukey.Από τα Post Hoc Tests επιλέγεται η μεταβλητή απόκρισης και επιλέγεται η μέθοδος Tukey.



(Εικόνα 9.4) Tukey test σε Jamovi.

Στα δεξιά, κάτω από τον προσχηματισμένο πίνακα Ancova, σχηματίζεται ο πίνακας αποτελεσμάτων του Tukey test από τον οποίο διακρίνεται ότι το λίπασμα S είναι πιο αποτελεσματικό από τα άλλα δύο λιπάσματα.

Post Hoc Tests

Post Hoc Comparisons - Fertilizer

Comparison							
Fertilizer		Fertilizer	Mean Difference	SE	df	t	p_{tukey}
C	-	F	3.14	0.0604	26.0	52.1	< .001
	-	S	−3.57	0.0570	26.0	−62.6	< .001
F	-	S	−6.72	0.0585	26.0	−114.8	< .001

(Εικόνα 9.5) Πίνακας Tukey.

9.3 Ancova R:

Η εισαγωγή δεδομένων από ένα αρχείο excel μπορεί να γίνει με την βιβλιοθήκη readxl
library(readxl)
μέσα από την εντολή :
df=read_excel('filename.xlsx')
Με αυτόν τον τρόπο η μεταβλητή df αποτελεί το σύνολο των δεδομένων στο οποίο θα γίνει η στατιστική ανάλυση. Με χρήση της εντολής head(df) προβάλλονται οι πρώτες 6 γραμμές των δεδομένων.

```
> head(df)
# A tibble: 6 x 3
  Yield Height Fertilizer
  <dbl>  <dbl> <fct>
1  12.2     45 C
2  12.4     52 C
3  11.9     42 C
4  11.3     35 C
5  11.8     40 C
6  12.1     48 C
>
```

(Εικόνα 9.6) Έλεγχος σωστής εισαγωγής δεδομένων.

Μετά την εισαγωγή του πλαισίου δεδομένων όπως αυτό της εικόνας 9.1 μπορεί να γίνει χρήση της εντολής str(df) για τον έλεγχο της μορφής κάθε στήλης του πλαισίου δεδομένων.
str(df)

```
> str(df)
tibble [30 x 3] (S3: tbl_df/tbl/data.frame)
 $ Yield     : num [1:30] 12.2 12.4 11.9 11.3 11.8 12.1 13.1 12.7 12.4 11.4 ...
 $ Height    : num [1:30] 45 52 42 35 40 48 60 61 50 33 ...
 $ Fertilizer: chr [1:30] "C" "C" "C" "C" ...
```

(Εικόνα 9.7) Έλεγχος μορφής δεδομένων.

Σε αυτό το παράδειγμα οι τύποι των λιπασμάτων διαβάζονται ως χαρακτήρες. Θα πρέπει αρχικά να γίνει η μετατροπή τους σε factors με την εντολή as.factor(filename$variabel) ή πιο συγκεκριμένα για το παρών παράδειγμα:

df$Fertilizer = as.factor(df$Fertilizer)

Μετά την σωστή διαμόρφωση των δεδομένων μπορεί να γίνει η ανάλυση Ancova μέσω της εντολής:

```
model = lm(μεταβλητή απόκρισης~συμμεταβλητή* κατηγορική μεταβλητή, data=df)
ancova = aov(model)
summary(ancova)
```

Όπως προαναφέρθηκε για το συγκεκριμένο παράδειγμα ,μεταβλητή απόκρισης αποτελεί η απόδοση(Yield), κατηγορική μεταβλητή αποτελεί το επίπεδο εφαρμογής λιπάσματος(Fertilizer) ενώ συμμεταβλητή αποτελεί το ύψος των φυτών(Height).

```
model = lm(Yield~Height*Fertilizer, data=df)
ancova = aov(model)
summary(ancova)
```

Με τον τρόπο αυτό σχηματίζεται ο πίνακας Ancova όπως φαίνεται στην εικόνα 9.6.

```
> model = lm(Yield~Height*Fertilizer, data=df)
> ancova = aov(model)
> summary(ancova)
                 Df Sum Sq Mean Sq  F value   Pr(>F)
Height            1   0.47    0.47   31.791 8.34e-06 ***
Fertilizer        2 213.90  106.95 7201.300  < 2e-16 ***
Height:Fertilizer 2   0.06    0.03    2.063    0.149
Residuals        24   0.36    0.01
```

(Εικόνα 9.8) Πίνακας Ancova dependent~independent*factor

Στην περίπτωση που η αλληλεπίδραση μεταξύ της συμμεταβλητής και της κατηγορικής μεταβλητής δεν είναι σημαντική, όπως και στο παραπάνω παράδειγμα, τότε η εντολή που χρησιμοποιείται είναι:

```
model = lm(Yield~Height+Fertilizer, data=df)
ancova = aov(model)
summary(ancova)
```

```
> model = lm(Yield~Height+Fertilizer, data=df)
> ancova = aov(model)
> summary(ancova)
            Df Sum Sq Mean Sq F value   Pr(>F)
Height       1   0.47    0.47   29.39 1.11e-05 ***
Fertilizer   2 213.90  106.95 6657.09  < 2e-16 ***
Residuals   26   0.42    0.02
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

(Εικόνα 9.9) Πίνακας Ancova dependent~independent+factor

Από τον πίνακα Ancova φαίνεται πως ο τύπος λιπάσματος αλλά και το αρχικό ύψος των φυτών έχουν εξαιρετικά σημαντική επίδραση στην τελική απόδοση των φυτών($p < 0.01$).

Post-hoc testing

Για την διάκριση του πιο αποτελεσματικού λιπάσματος γίνεται test σύγκρισης μέσων όρων με την βιβλιοθήκη agricolae

```
library(agricolae)
```

Για least significant difference test(LSD) χρησιμοποιείται η εντολή
LSD.test(model,"κατηγορική μεταβλητή",p.adj="bonferroni") η οποία με βάση των δεδομένων του παραδείγματος γίνεται :

```
out1 = LSD.test(model,"Fertilizer", p.adj = "bonferroni")
out1
```

```
> out1 = LSD.test(model,"Fertilizer", p.adj = "bonferroni")
> out1
$statistics
     MSerror Df     Mean       CV  t.value       MSD
  0.01606588 26 12.45667 1.017537 2.558942 0.1450533

$parameters
        test  p.ajusted     name.t ntr alpha
  Fisher-LSD bonferroni Fertilizer   3  0.05

$means
  Yield       std  r      LCL      UCL  Min  Max    Q25   Q50    Q75
C 12.13 0.5578729 10 12.04761 12.21239 11.3 13.1 11.825 12.15 12.400
F  9.41 0.5065131 10  9.32761  9.49239  8.5 10.3  9.200  9.50  9.575
S 15.83 0.4715224 10 15.74761 15.91239 15.0 16.6 15.650 15.80 15.950

$comparison
NULL

$groups
  Yield groups
S 15.83      a
C 12.13      b
F  9.41      c

attr(,"class")
[1] "group"
```

(Εικόνα 9.10) least significant difference test

Για την διεξαγωγή HSD test μπορεί να χρησιμοποιηθεί η εντολή:
HSD.test(model,"κατηγορική μεταβλητή ",group=TRUE,console=TRUE,main="μεταβλητή απόκρισης")
η οποία με βάση των δεδομένων του παραδείγματος γίνεται:
HSD.test(model,"Fertilizer",group=TRUE,console=TRUE,main="Yield")

```
> HSD.test(model,"Fertilizer",group=TRUE,console=TRUE,main="Yield")

Study: Yield

HSD Test for Yield

Mean Square Error:  0.01606588

Fertilizer,  means

  Yield       std  r  Min  Max
C 12.13 0.5578729 10 11.3 13.1
F  9.41 0.5065131 10  8.5 10.3
S 15.83 0.4715224 10 15.0 16.6

Alpha: 0.05 ; DF Error: 26
Critical Value of Studentized Range: 3.514171

Minimun Significant Difference: 0.1408559

Treatments with the same letter are not significantly different.

  Yield groups
S 15.83      a
C 12.13      b
F  9.41      c
```

(Εικόνα 9.11)HSD Tukey Test

Και από τα δύο τεστ σύγκρισης μέσων όρων διακρίνεται ότι το λίπασμα S είναι πιο αποτελεσματικό από τα άλλα δύο λιπάσματα.

Σύνοψη κώδικα

```
library(readxl)
library (agricolae)
df=read_excel('Ancova.xlsx')
head(df)
str(df)
df$Fertilizer = as.factor(df$Fertilizer)
model = lm(Yield~Height+Fertilizer, data=df)
ancova = aov(model)
summary(ancova)
out1 = LSD.test(model,"Fertilizer", p.adj = "bonferroni")
out1
HSD.test(model,"Fertilizer",group=TRUE,console=TRUE,main="Yield")
```

9.4 Ancova Python:

Η ανάλυση συνδιακύμανσης μέσω της γλώσσας προγραμματισμού Python μπορεί να γίνει με την βοήθεια των βιβλιοθηκών pandas ,numpy και pingouin.

```
import numpy as np
import pandas as pd
pip install pingouin
from pingouin import ancova
```

Αρχικά γίνεται εισαγωγή των δεδομένων μέσα από την εντολή :

```
data = pd.read_excel('filename.xlsx')
```

Με αυτόν τον τρόπο η μεταβλητή data αποτελεί το πλαίσιο δεδομένων. Μετά την εισαγωγή του πλαισίου δεδομένων όπως αυτό της εικόνας 9.1 ακολουθεί διαμόρφωση των δεδομένων έτσι ώστε να είναι κατάλληλα για τα στατιστικά μοντέλα.

```
data = pd.DataFrame(data, columns=[όνομα πρώτης στήλης,όνομα δεύτερης στήλης,όνομα τρίτης στήλης])
```

Για το παράδειγμα της εικόνας 9.1 αυτό θα αντιστοιχούσε σε:

```
data = pd.DataFrame(data, columns=['Yield','Height','Fertilizer'])
```

Μετά την σωστή διαμόρφωση των δεδομένων μπορεί να γίνει η ανάλυση Ancova μέσω της εντολής:

```
ancova(data=data, dv='μεταβλητή απόκρισης', covar='συμμεταβλητή', between='κατηγορική μεταβλητή')
```

Για το παράδειγμα της εικόνας 9.1 αυτό θα αντιστοιχούσε σε:

```
ancova(data=data, dv='Yield', covar='Height', between='Fertilizer')
```

```
In [9]: ancova(data=data, dv='Yield', covar='Height', between='Fertilizer')
Out[9]:
       Source          SS  DF            F         p-unc       np2
0  Fertilizer  213.903804   2  6657.084890  5.855391e-36  0.998051
1      Height    6.693287   1   416.615135  1.580422e-17  0.941258
2    Residual    0.417713  26          NaN           NaN       NaN
```

(Εικόνα 9.12) Πίνακας Ancova dependent~independent+factor

Από τον πίνακα Ancova φαίνεται πως ο τύπος λιπάσματος αλλά και το αρχικό ύψος των φυτών έχουν εξαιρετικά σημαντική επίδραση στην τελική απόδοση των φυτών($p < 0.01$).

Σύνοψη κώδικα

```
import numpy as np
import pandas as pd
pip install pingouin
from pingouin import ancova
data = pd.read_excel('Ancova.xlsx')
data = pd.DataFrame(data, columns=['Yield','Height','Fertilizer'])
ancova = ancova(data=data, dv='Yield', covar='Height', between='Fertilizer')
print(ancova)
```

10. Full Factorial Design

Το πλήρες παραγοντικό πείραμα ή αλλιώς Full Factorial Design αποτελεί έναν πειραματικό σχεδιασμό ο οποίος αποτελείται από τουλάχιστον δύο ανεξάρτητους παράγοντες (ποιοτικές μεταβλητές) με διακριτά επίπεδα και μια εξαρτημένη ποσοτική μεταβλητή. Στον σχεδιασμό αυτό αναλύεται η επίδραση όλων των παραγόντων αλλά και ο συνδυασμός των επιπέδων των παραγόντων. Κατά τον πλήρες παραγοντικό πειραματικό σχεδιασμό ελέγχονται όλοι οι πιθανοί συνδυασμοί των επιπέδων των παραγόντων. Για n υπό εξέταση παράγοντες με τον κάθε παράγοντα να έχει s διαφορετικά επίπεδα, το σύνολο των απαραίτητων πειραματικών μονάδων για την διεξαγωγή του πειράματος είναι sn[32]. Συνεπώς με την αύξηση των παραγόντων και των επιπέδων τους , η αύξηση των απαραίτητων πειραματικών μονάδων είναι πολλαπλασιαστική. Λόγω της εκθετικής αύξησης των πειραματικών μονάδων , ο σχεδιασμός αυτός συνιστάται για την εξέταση μικρού πλήθους παραγόντων και παραγοντικών επιπέδων. Στην πραγματικότητα, είναι απαραίτητες τουλάχιστον δύο επαναλήψεις κάθε συνδυασμού για να προσδιοριστεί ένα άθροισμα τετραγώνων λόγω σφάλματος[33]. Με x επαναλήψεις του πειράματος οι συνολικές πειραματικές μονάδες που χρειάζονται είναι x* sn. Για μεγάλο πλήθος παραγόντων ο σχεδιασμός αυτός απαιτεί μεγάλο αριθμό πειραματικών εκτελέσεων και δεν είναι αποτελεσματικός[34]. Παρά αυτό το μειονέκτημα, ένας πλήρες παραγοντικός σχεδιασμός εφαρμόζεται σε πειράματα που έχουν ως στόχο την εύρεση μόνο των παραγόντων ενδιαφέροντος του πειράματος. Μπορεί ακόμα να εφαρμοστεί και για την διερεύνηση των σχέσεων μεταξύ διαφορετικών παραγόντων ή και σε πειράματα που έχουν ως στόχο την εύρεση συστάσεων μέσα από μια μεγάλη γκάμα συνθηκών[35].

10.1 Παράδειγμα Full Factorial Design

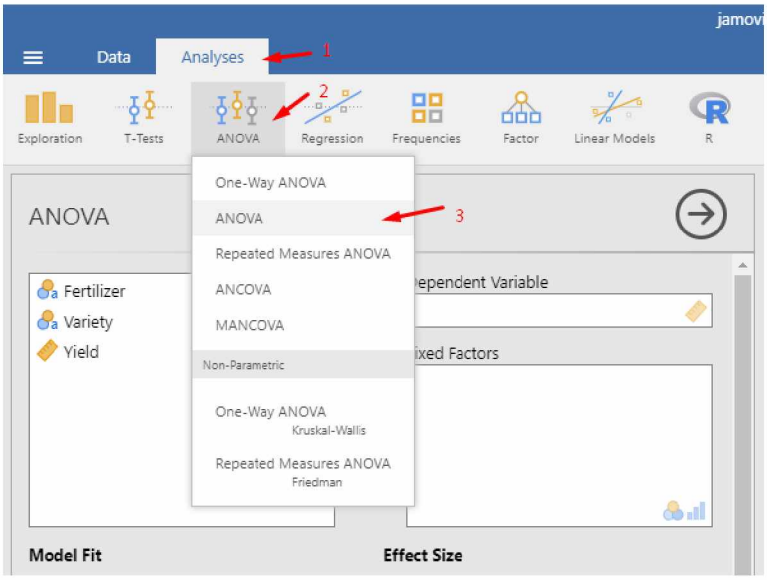
Σε ένα πείραμα αγρού με τρεις επαναλήψεις εξετάζεται η απόδοση δύο διαφορετικών ποικιλιών καλαμποκιού(A,B) μετά από την εφαρμογή τριών διαφορετικών λιπασμάτων(C,F,S).Σε αυτό το πείραμα η απόδοση αποτελεί την εξαρτημένη ποσοτική μεταβλητή ενώ η ποικιλία και το είδος του λιπάσματος αποτελούν τις ανεξάρτητες ποιοτικές μεταβλητές. Ο παράγοντας ποικιλία έχει δύο επίπεδα ενώ ο παράγοντας λίπασμα έχει τρία επίπεδα. Συνεπώς Οι συνολικές πειραματικές μονάδες είναι 18 .Τα δεδομένα του πειράματος φαίνονται στην εικόνα 10.1 στο τέλος του κεφαλαίου. Μιας και εξετάζεται η επίδραση δύο παραγόντων, η ανάλυση ενός τέτοιου πειράματος γίνεται με χρήση two-way Anova[36]. Παρακάτω περιγράφεται ο τρόπος ανάλυσης two-way Anova με βάση το παραπάνω πείραμα. Η ανάλυση αυτή θα γίνει μέσω του στατιστικού λογισμικού Jamovi,της γλώσσας R, αλλά και μέσω Python.

1	Fertilizer	Variety	Yield
2	S	A	3.39
3	S	A	3.34
4	S	A	3.59
5	C	A	3.58
6	C	A	4.12
7	C	A	4.72
8	F	A	5.06
9	F	A	4.05
10	F	A	4.09
11	S	B	2.2
12	S	B	2.6
13	S	B	2.72
14	C	B	3.12
15	C	B	3.28
16	C	B	3.74
17	F	B	3.43
18	F	B	4.6
19	F	B	5.26

(Εικόνα 10.1) Δεδομένα πειράματος προς ανάλυση two-way Anova.

10.2 Two-Way Anova Jamovi:

Με την εισαγωγή του αρχείου excel όπως αυτό της εικόνας 10.1 στο jamovi μπορεί να εκτελεστεί η two-way Anova ανάλυση.



(Εικόνα 10.2)Επιλογή Anova ανάλυσης.

Ως dependent variable τίθεται η μεταβλητή απόκρισης δηλαδή η απόδοση (Yield) ενώ ως Fixed Factors οι κατηγορικές μεταβλητές, δηλαδή το λίπασμα (Fertilizer) και η ποικιλία (Variety). Μετά την κατηγοριοποίηση της κάθε μεταβλητής, γίνεται ο σχηματισμός του πίνακα Anova.

ANOVA

ANOVA - Yield

	Sum of Squares	df	Mean Square	F	p
Fertilizer	6.253	2	3.126	10.99	0.002
Variety	1.383	1	1.383	4.86	0.048
Fertilizer ✻ Variety	0.791	2	0.396	1.39	0.286
Residuals	3.413	12	0.284		

[3]

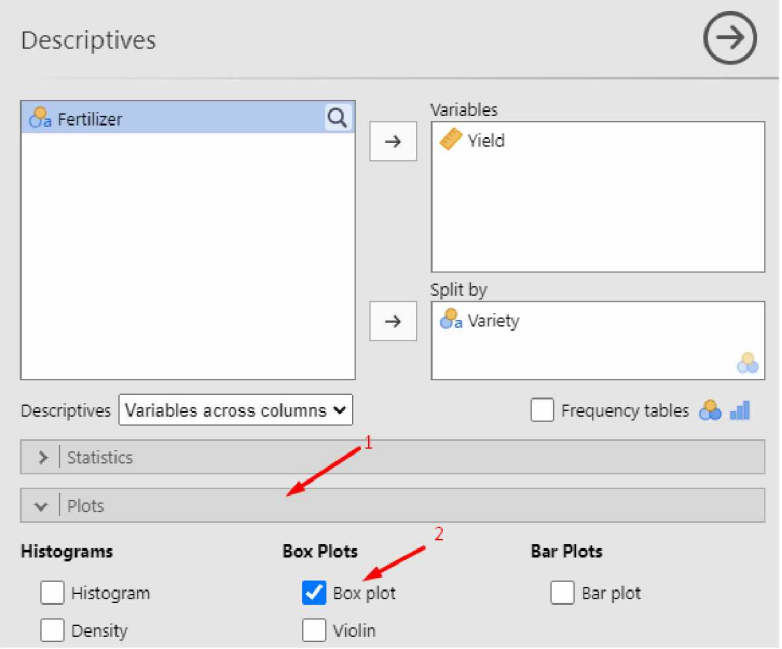
(Εικόνα 10.3) Σχηματισμός πίνακα Anova σε Jamovi.

Από τον πίνακα Anova το συμπέρασμα είναι ότι η απόδοση επηρεάζεται σημαντικά και από τις δύο μεταβλητές των παραγόντων μεμονωμένα. Ωστόσο, η αλληλεπίδραση των δύο παραγόντων δεν έχει σημαντική επίδραση στην απόδοση.

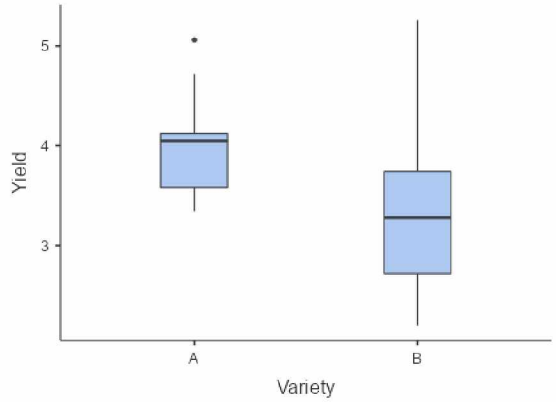
Post-hoc testing

Μιας και το πείραμα έχει μόνο δύο διαφορετικά επίπεδα ποικιλιών, η διάκριση της πιο αποτελεσματικής ποικιλίας μπορεί να γίνει με τον σχηματισμό ενός γραφήματος boxplot. Ο σχηματισμός του γραφήματος μπορεί να γίνει από την καρτέλα “Analyses”, μέσω της επιλογής “Exploration” και στην συνέχεια “Descriptives”.

Ως Variables τίθεται η μεταβλητή απόκρισης δηλαδή το Yield ενώ στο Split by τοποθετείται η μεταβλητή των ποικιλιών , δηλαδή η μεταβλητή Variety. Τέλος από το παράρτημα “Plots” επιλέγεται το “Box plot”.



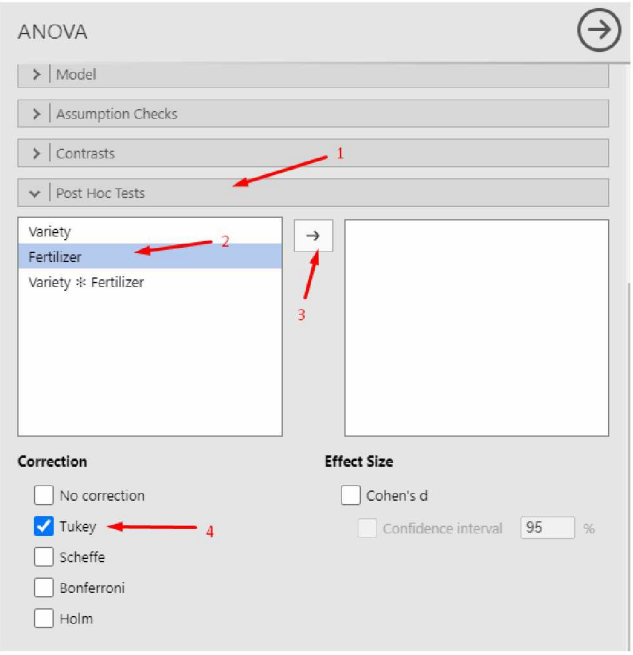
(Εικόνα 10.4)Σχηματισμός γραφήματος box-plot σε Jamovi.



(Γράφημα 10.1) Box-plot Split By Variety.

Από το γράφημα 10.1 είναι εμφανές πως η ποικιλία Α είναι πιο παραγωγική από την ποικιλία Β.

Για την διάκριση του πιο αποτελεσματικού λιπάσματος γίνεται test σύγκρισης μέσων όρων με την μέθοδο Tukey. Από τα Post Hoc Tests επιλέγονται οι παράγοντες που εμφάνισαν σημαντική επίδραση στην απόδοση. Για το συγκεκριμένο παράδειγμα η αλληλεπίδραση μεταξύ ποικιλίας και λιπάσματος δεν εμφάνισε σημαντική επίδραση στην απόδοση, συνεπώς δεν επιλέγεται.



(Εικόνα 10.5) Tukey test σε Jamovi.

Στα δεξιά, κάτω από τον προσχηματισμένο πίνακα Anova, σχηματίζεται ο πίνακας αποτελεσμάτων του Tukey test από τον οποίο διακρίνεται ότι το λίπασμα F είναι το πιο αποδοτικό.

Post Hoc Tests

Post Hoc Comparisons - Fertilizer

Comparison							
Fertilizer		Fertilizer	Mean Difference	SE	df	t	p_{tukey}
C	-	F	−0.655	0.308	12.0	−2.13	0.126
	-	S	0.787	0.308	12.0	2.55	0.061
F	-	S	1.442	0.308	12.0	4.68	0.001

Note. Comparisons are based on estimated marginal means

(Εικόνα 10.6) Πίνακας Tukey.

10.3 Two-Way Anova R:

Η ανάλυση two-way Anova στην R μπορεί να γίνει με χρήση της βιβλιοθήκης stats.

```
library(stats)
```

Η εισαγωγή δεδομένων από ένα αρχείο excel μπορεί να γίνει με την βιβλιοθήκη readxl
```
library(readxl)
```
μέσα από την εντολή :

```
df=read_excel('filename.xlsx')
```

Με αυτόν τον τρόπο η μεταβλητή df αποτελεί το σύνολο των δεδομένων στο οποίο θα γίνει η στατιστική ανάλυση.Με χρήση της εντολής head(df) μπορεί να γίνει έλεγχος για το εάν τα δεδομένα εισήχθησαν σωστά.

```
> head(df)
# A tibble: 6 x 3
  Fertilizer Variety Yield
  <chr>      <chr>   <dbl>
1 S          A        3.39
2 S          A        3.34
3 S          A        3.59
4 C          A        3.58
5 C          A        4.12
6 C          A        4.72
>
```

(Εικόνα 10.7) Έλεγχος σωστής εισαγωγής δεδομένων.

Μετά την εισαγωγή του πλαισίου δεδομένων όπως αυτό της εικόνας 10.1 μπορεί να γίνει χρήση της εντολής str(df) για τον έλεγχο της μορφής κάθε στήλης του πλαισίου δεδομένων.

```
str(df)
```

```
> str(df)
tibble [18 x 3] (S3: tbl_df/tbl/data.frame)
 $ Fertilizer: chr [1:18] "S" "S" "S" "C" ...
 $ Variety   : chr [1:18] "A" "A" "A" "A" ...
 $ Yield     : num [1:18] 3.39 3.34 3.59 3.58 4.12 4.72 5.06 4.05 4.09 2.2 ...
>
```

(Εικόνα 10.8) Έλεγχος μορφής δεδομένων.

Σε αυτό το παράδειγμα οι τύποι των λιπασμάτων καθώς και οι ποικιλίες διαβάζονται ως χαρακτήρες.Θα πρέπει αρχικά να γίνει η μετατροπή τους σε factors με την εντολή as.factor(filename$variabel) ή πιο συγκεκριμένα για το παρών παράδειγμα:

```
df$Fertilizer = as.factor(df$Fertilizer)
```

```
df$Variety = as.factor(df$Variety)
```

Μετά την σωστή διαμόρφωση των δεδομένων μπορεί να γίνει η ανάλυση two-way Anova μέσω της εντολής:

```
aov.res1 =  aov(Μεταβλητή απόκρισης ~ Παράγοντας1 + Παράγοντας2 +
Παράγοντας1:Παράγοντας2, data = df)
summary(aov.res1)
```

Όπως προαναφέρθηκε για το συγκεκριμένο παράδειγμα ,μεταβλητή απόκρισης αποτελεί η απόδοση (Yield), ενώ οι δύο παράγοντες είναι η ποικιλία (Variety) και το λίπασμα (Fertilizer). Επομένως για το συγκεκριμένο παράδειγμα η παραπάνω εντολή γράφεται ως:

```
aov.res1 =  aov(Yield ~ Fertilizer + Variety + Fertilizer:Variety, data = df)
summary(aov.res1)
```

Με τον τρόπο αυτό σχηματίζεται ο πίνακας Anova όπως φαίνεται στην εικόνα 10.6.

```
> aov.res1 <- aov(Yield ~ Fertilizer + Variety + Fertilizer:Variety, data = df)
> summary(aov.res1)
                  Df Sum Sq Mean Sq F value  Pr(>F)
Fertilizer         2  6.253  3.1263  10.992 0.00194 **
Variety            1  1.383  1.3833   4.864 0.04767 *
Fertilizer:Variety 2  0.791  0.3955   1.391 0.28627
Residuals         12  3.413  0.2844
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

(Εικόνα 10.9) Πίνακας Anova dependent~independent1+independent2 + independent1:independent2.

Από τον πίνακα Anova το συμπέρασμα είναι ότι η απόδοση επηρεάζεται σημαντικά και από τις δύο μεταβλητές των παραγόντων μεμονωμένα. Ωστόσο, η αλληλεπίδραση των δύο παραγόντων δεν έχει σημαντική επίδραση στην απόδοση.

Post-hoc testing

Για την διάκριση του πιο αποτελεσματικού λιπάσματος και ποικιλίας γίνεται test σύγκρισης μέσων όρων Tukey HSD.

Για την διεξαγωγή HSD test μπορεί να χρησιμοποιηθεί η εντολή:

```
TukeyHSD(x = aov.res1,ordered = FALSE,which = "Factor",conf.level = 0.95)
```

Η εντολή αυτή εμπεριέχει διαφορετικά στοιχεία τα οποία πρέπει να προσδιοριστούν
Αρχικά το στοιχείο “x” αντιπροσωπεύει το μοντέλο με το οποίο έγινε ο σχηματισμός του πίνακα Anova προηγουμένως. Έτσι το στοιχείο “x” είναι ίσο με την μεταβλητή aov.res1 που τέθηκε προηγουμένως
(x = aov.res1).

Το στοιχείο “ordered” μπορεί να οριστεί έως “TRUE” ή “FALSE” και προσδιορίζει μια λογική τιμή που υποδεικνύει εάν τα επίπεδα του παράγοντα πρέπει να ταξινομηθούν σύμφωνα με τον αυξανόμενο μέσο όρο στο δείγμα πριν ληφθούν οι διαφορές. Εάν το στοιχείο “ordered” έχει οριστεί σε TRUE, τότε οι υπολογισμένες διαφορές στα μέσα θα είναι όλες θετικές.
Το στοιχείο “which” καθορίζει τον παράγοντα με βάση τον οποίο θα διεξαχθεί το test σύγκρισης μέσων όρων. Εάν αυτό το στοιχείο δεν συμπληρωθεί στην εντολή τότε το test διεξάγεται βάση όλων των παραγόντων.
Τέλος το στοιχείο “conf.level” αντιπροσωπεύει το επίπεδο εμπιστοσύνης ενώ εάν αυτό το στοιχείο δεν συμπληρωθεί στην εντολή τότε το test θεωρεί το επίπεδο εμπιστοσύνης ίσο με 95%.
Για το συγκεκριμένο παράδειγμα η εντολή αυτή θα γινόταν:

```
TukeyHSD(x = aov.res1,ordered = FALSE,conf.level = 0.95)
```

Η πιο συνοπτικά:

```
TukeyHSD(x = aov.res1)
```

```
> TukeyHSD(x = aov.res1)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = Yield ~ Fertilizer + Variety + Fertilizer:Variety, data = df)

$Fertilizer
          diff        lwr         upr     p adj
F-C  0.6550000 -0.1664328  1.47643280 0.1257897
S-C -0.7866667 -1.6080995  0.03476613 0.0609218
S-F -1.4416667 -2.2630995 -0.62023387 0.0014209

$Variety
          diff       lwr          upr     p adj
B-A -0.5544444 -1.102195 -0.006694046 0.0476691
```

(Εικόνα 10.10)HSD Tukey Test dependent~independent1+independent2.

Από τον πίνακα των αποτελεσμάτων του Tukey test στην εικόνα 10.9 διακρίνεται ότι το λίπασμα F και η ποικιλία A είναι πιο αποδοτικά.

Σύνοψη κώδικα

```
library(readxl)
library(stats)
df=read_excel('2way.xlsx')
str(df)
df$Fertilizer = as.factor(df$Fertilizer)
df$Variety = as.factor(df$Variety)
aov.res1 <- aov(Yield ~ Fertilizer + Variety + Fertilizer:Variety, data = df)
summary(aov.res1)
TukeyHSD(x = aov.res1)
```

10.4 Two-Way Anova Python:

Η ανάλυση two-way Anova μέσω της γλώσσας προγραμματισμού Python μπορεί να γίνει με την βοήθεια των βιβλιοθηκών pandas και bioinfokit.

```
import pandas as pd
pip install bioinfokit
from bioinfokit.analys import stat
```

Αρχικά γίνεται εισαγωγή των δεδομένων μέσα από την εντολή :

```
data = pd.read_excel('filename.xlsx')
```

Με αυτόν τον τρόπο η μεταβλητή data αποτελεί το πλαίσιο δεδομένων. Μετά την εισαγωγή του πλαισίου δεδομένων όπως αυτό της εικόνας 10. 1 ακολουθεί διαμόρφωση των δεδομένων έτσι ώστε να είναι κατάλληλα για τα στατιστικά μοντέλα.

```
data = pd.DataFrame(data, columns=[όνομα πρώτης στήλης,όνομα δεύτερης στήλης,όνομα τρίτης στήλης])
```

Για το παράδειγμα της εικόνας 10.1 αυτό θα αντιστοιχούσε σε:

```
data = pd.DataFrame(data, columns=['Fertilizer','Variety','Yield'])
```

Μετά την σωστή διαμόρφωση των δεδομένων μπορεί να γίνει η ανάλυση two-way Anova με τον εξής τρόπο:

```
res = stat()
res.anova_stat(df=data, res_var='Μεταβλητή Απόκρισης', anova_model='Μεταβλητή Απόκρισης~C(Παράγοντας1)+C(Παράγοντας2)')
print(res.anova_summary)
```

Εάν στην ανάλυση two-way anova πρέπει να συμπεριληφθεί και η αλληλεπίδραση τότε η παραπάνω εντολή γράφεται ως:

```
res = stat()
res.anova_stat(df=data, res_var='Μεταβλητή Απόκρισης', anova_model='Μεταβλητή
Απόκρισης~C(Παράγοντας1)+C(Παράγοντας2)+C(Παράγοντας1):C(Παράγοντας2)')
print(res.anova_summary)
```

Για το παράδειγμα της εικόνας 10.1 αυτό θα αντιστοιχούσε σε:

```
res = stat()
res.anova_stat(df=data, res_var='Yield',
anova_model='Yield~C(Fertilizer)+C(Variety)+C(Fertilizer):C(Variety)')
print(res.anova_summary)
```

Με τον τρόπο αυτό σχηματίζεται ο πίνακας Anova όπως φαίνεται στην εικόνα 10.8.

	df	sum_sq	mean_sq	F	PR(>F)
C(Fertilizer)	2.0	6.252544	3.126272	10.992304	0.001938
C(Variety)	1.0	1.383339	1.383339	4.863966	0.047669
C(Fertilizer):C(Variety)	2.0	0.791078	0.395539	1.390757	0.286269
Residual	12.0	3.412867	0.284406	NaN	NaN

(Εικόνα 10.11) Πίνακας Anova dependent~independent1+independent2 + independent1:independent2.

Από τον πίνακα Anova το πόρισμα είναι ότι η απόδοση επηρεάζεται σημαντικά και από τις δύο μεταβλητές των παραγόντων μεμονωμένα. Ωστόσο, η αλληλεπίδραση των δύο παραγόντων δεν έχει σημαντική επίδραση στην απόδοση.

Post-hoc testing

Για την διάκριση του πιο αποτελεσματικού λιπάσματος και ποικιλίας γίνεται test σύγκρισης μέσων όρων Tukey HSD.

Για την διεξαγωγή HSD test μπορεί να χρησιμοποιηθεί η εντολή:

```
res.tukey_hsd(df=data, res_var='Μεταβλητή Απόκρισης', xfac_var='Παράγοντας',
anova_model='Μεταβλητή
Απόκρισης~C(Παράγοντας1)+C(Παράγοντας2)+C(Παράγοντας1):C(Παράγοντας2)')
print(res.tukey_summary)
```

Η εντολή αυτή εμπεριέχει διαφορετικά στοιχεία τα οποία πρέπει να προσδιοριστούν.
Αρχικά το στοιχείο "df" αντιπροσωπεύει το το πλαίσιο δεδομένων βάση του οποίου θα γίνει το Tukey test.
Έτσι το στοιχείο "df" είναι ίσο με την μεταβλητή data που τέθηκε προηγουμένως (df=data).

Το στοιχείο “res_var” είναι η μεταβλητή απόκρισης δηλαδή η απόδοση(Yield) για το συγκεκριμένο πείραμα (res_var=’Yield’).
Το στοιχείο “xfarc_var” καθορίζει βάση ποιού παράγοντα θα διεξαχθεί το test σύγκρισης μέσων όρων.
Στο συγκεκριμένο παράδειγμα για την σύγκριση μέσων όρων με βάση το λίπασμα το στοιχείο “xfarc_var” είναι η μεταβλητή Fertilizer (xfarc_var = ‘Fertilizer’).Παρομοίως για την σύγκριση μέσων όρων με βάση την ποικιλία ή την αλληλεπίδραση των δύο παραγόντων, το στοιχείο “xfarc_var” είναι ίσο με την μεταβλητή Variety και Fertilizer:Variety αντίστοιχα(xfarc_var = ‘Variety’ και xfarc_var = ['Fertilizer','Variety'])
Τέλος το στοιχείο “ anova_model ” αντιπροσωπεύει το μοντέλο με το οποίο έγινε ο σχηματισμός του πίνακα Anova προηγουμένως.
Συνεπώς για το λίπασμα:

```
res.tukey_hsd(df=data, res_var='Yield', xfac_var='Fertilizer',
anova_model='Yield~C(Fertilizer)+C(Variety)+C(Fertilizer):C(Variety)')
print(res.tukey_summary)
```

	group1	group2	Diff	Lower	Upper	q-value	p-value
0	S	C	0.786667	-0.034362	1.607695	3.613244	0.060806
1	S	F	1.441667	0.620638	2.262695	6.621730	0.001417
2	C	F	0.655000	-0.166029	1.476029	3.008485	0.125523

(Εικόνα 10.12) HSD Tukey Test για το λίπασμα.

Από την εικόνα 10.12 φαίνεται πως το πιο αποτελεσματικό λίπασμα είναι το λίπασμα F.
Παρομοίως για τις ποικιλίες:

```
res.tukey_hsd(df=data, res_var='Yield', xfac_var='Variety',
anova_model='Yield~C(Fertilizer)+C(Variety)+C(Fertilizer):C(Variety)')
print(res.tukey_summary)
```

	group1	group2	Diff	Lower	Upper	q-value	p-value
0	A	B	0.554444	0.006694	1.102195	3.118963	0.047669

(Εικόνα 10.13) HSD Tukey Test για τις ποικιλίες.

Από την εικόνα 10.13 φαίνεται πως η πιο αποδοτική ποικιλία είναι η ποικιλία A.

Τέλος στο συγκεκριμένο πείραμα η αλληλεπίδραση των δύο παραγόντων δεν έχει σημαντική επίδραση στην απόδοση, με αποτέλεσμα η σύγκριση των μέσων όρων της αλληλεπίδρασης να μην είναι αναγκαία. Παρόλα αυτά η εντολή με την οποία θα γινόταν αυτή η σύγκριση είναι η εξής:

```
res.tukey_hsd(df=data, res_var='Yield', xfac_var=['Fertilizer','Variety'],
anova_model='Yield~C(Fertilizer)+C(Variety)+C(Fertilizer):C(Variety)')
print(res.tukey_summary)
```

Σύνοψη κώδικα

```
import pandas as pd
pip install bioinfokit
from bioinfokit.analys import stat
data = pd.read_excel('2wa.xlsx')
data = pd.DataFrame(data, columns=['Fertilizer','Variety','Yield'])
res = stat()
res.anova_stat(df=data, res_var='Yield',
anova_model='Yield~C(Fertilizer)+C(Variety)+C(Fertilizer):C(Variety)')
print(res.anova_summary)
res.tukey_hsd(df=data, res_var='Yield', xfac_var='Fertilizer',
anova_model='Yield~C(Fertilizer)+C(Variety)+C(Fertilizer):C(Variety)')
print(res.tukey_summary)
res.tukey_hsd(df=data, res_var='Yield', xfac_var='Variety',
anova_model='Yield~C(Fertilizer)+C(Variety)+C(Fertilizer):C(Variety)')
print(res.tukey_summary)
res.tukey_hsd(df=data, res_var='Yield', xfac_var=['Fertilizer','Variety'],
anova_model='Yield~C(Fertilizer)+C(Variety)+C(Fertilizer):C(Variety)')
print(res.tukey_summary)
```

11. Συμπεράσματα

Σκοπός της συγκεκριμένης εργασίας ήταν η στατιστική ανάλυση των δεδομένων που προκύπτουν από διάφορους πειραματικούς σχεδιασμούς, χρησιμοποιώντας ως υπολογιστικά μέσα το στατιστικό λογισμικό Jamovi αλλά και τις γλώσσες προγραμματισμού R και Python.

Από το πρώτο κεφάλαιο της εργασίας φαίνεται πως η εισαγωγή δεδομένων από ένα αρχείο excel αποτελεί μια αρκετά εύκολη διαδικασία τόσο στο Jamovi όσο και στην R, ενώ η δυνατότητα αυτή προσφέρεται στην Python μέσω της βιβλιοθήκης pandas.

Τα test of normality και T-test, που καλύπτονται στα κεφάλαια 2 και 3 της εργασίας, φαίνεται να εκτελούνται με ευκολία και αξιοπιστία και από τα τρία υπολογιστικά μέσα. Τα 2 αυτά test πραγματοποιούνται στο Jamovi μέσα από 4-5 click ενώ η R έχει ενσωματωμένες λειτουργίες για την εκτέλεσή τους. Τέλος στην Python αυτός ο σκοπός εξυπηρετείται μέσω της βιβλιοθήκης scipy.

Γενικά και τα τρία υπολογιστικά μέσα (Jamovi, R, Python) έχουν δυνατότητες αναλύσεις σχεδόν όλων των πειραματικών σχεδιασμών(CRD, RCBD, Λατινικό τετράγωνο, Full factorial design, Ancova), γραφικής απεικόνισης δεδομένων μέσω θηκογράμματος (boxplot) αλλά και εκτέλεσης Post-hoc testing. Όλες αυτές οι λειτουργίες είναι εύκολα προσβάσιμες στο Jamovi μέσα από λίγα click, ενώ στις γλώσσες προγραμματισμού R και Python οι δυνατότητες αυτές προσφέρονται μέσα από ανάλογες βιβλιοθήκες.

Μια σημαντική διαφορά μεταξύ των υπολογιστικών μέσων προκύπτει στην ανάλυση των σχεδιασμών split-plot και strip-plot που καλύπτονται στα κεφάλαια 7 και 8 της εργασίας αντίστοιχα. Σε αντιδιαστολή με την Python, στην R αυτές οι αναλύσεις μπορούν να γίνουν εύκολα μέσω εξειδικευμένων βιβλιοθηκών που προσφέρονται. Από την άλλη στην Python η ανυπαρξία εξειδικευμένων βιβλιοθηκών για τις δύο αυτές αναλύσεις καθιστούν τη διαδικασία πιο χρονοβόρα και δύσκολη σε σύγκριση με την διαδικασία που απαιτείται στα άλλα δύο υπολογιστικά μέσα. Απόδειξη αυτού είναι η έκταση του κώδικα που χρειαζεται στην Python για τις συγκεκριμένες αναλύσεις.

Παρόλα αυτά οι δύο αυτές αναλύσεις φαίνεται να πραγματοποιούνται με πιο σύνθετο τρόπο ακόμα και στο Jamovi, στο οποίο απαιτείται η προσθήκη της βιβλιοθήκης gamlj αλλά και της Rj – Editor για τον υπολογισμό των p-value μέσω προγραμματισμού.

Εν κατακλείδι, αν και όλες οι αναλύσεις μπορούν να πραγματοποιηθούν και από τα τρία υπολογιστικά μέσα, η γλώσσα προγραμματισμού R φαίνεται να είναι το πιο αξιόπιστο μέσο, κυρίως λόγω του μεγάλου εύρους βιβλιοθηκών που παρέχει για την κάλυψη κάθε τύπου ανάλυσης. Το jamovi αν και δεν εξαλείφει πλήρως την ανάγκη για προγραμματισμό σε κάποιος αναλύσεις (split-plot, strip-plot), αποτελεί ένα αρκετά αξιόπιστο υπολογιστικό λογισμικό μέσω του οποίου οι στατιστικές αναλύσεις, γραφικές απεικονίσεις δεδομένων και post-hoc testing γίνονται πολύ απλά μέσα σε λίγα click. Τέλος αν και η Python δεν αποτελεί ούτε την πιο εύκολη στην χρήση αλλά ούτε και την πιο γρήγορα εναλλακτική , προτιμάται από πολλούς λόγο της συμβατότητας που έχει με άλλες εφαρμογές.

12. Παράρτημα

Jamovi Έκδοση 2.2.5

1) Rj – Editor to run R code inside jamovi Έκδοση 1.1.0
2) gamlj – General Analyses for linear models in jamovi Έκδοση 2.6.1 (https://gamlj.github.io)

R Έκδοση 4.1.2 (Build 443)

1) readxl Έκδοση 1.3.1 (https://readxl.tidyverse.org)
2) stats Έκδοση 4.1.2 (https://stat.ethz.ch/R-manual/R-devel/library/stats/html/00Index.html)
3) ggplot2 Έκδοση 3.3.5 (https://ggplot2.tidyverse.org)
4) agricolae Έκδοση 1.3.5 (https://rdrr.io/cran/agricolae/)
5) dplyr Έκδοση 1.0.8 (https://dplyr.tidyverse.org)

Python Έκδοση 3.7.9 (Ipython 7.31.1, spyder 5.1.5)

1) pandas Έκδοση 1.3.4 (https://pandas.pydata.org)
2) scipy Έκδοση 1.7.1 (https://scipy.org)
3) doex Έκδοση 0.0.7 (https://doex.rohitsanjay.com/en/latest/)
4) matplotlib Έκδοση 3.4.3 (https://matplotlib.org)
5) seaborn Έκδοση 0.11.2 (https://seaborn.pydata.org)
6) statsmodels Έκδοση 0.12.2 (https://www.statsmodels.org/0.6.1/)
7) bioinfokit Έκδοση 2.0.8 (https://pypi.org/project/bioinfokit/)
8) numpy Έκδοση 1.20.3 (https://numpy.org)
9) pingouin Έκδοση 0.5.1 (https://pingouin-stats.org)

13. Βιβλιογραφία

1) McLeod S.A. What a p-value tells you about statistical significance. simplypsychology.org. http://www.simplypsychology.org/p-value.html. Published May 20, 2019. Accessed March 29, 2022.

2) Salkind N. *Encyclopedia of Research Design*. SAGE Publications, Inc.; 2010. doi:10.4135/9781412961288

3) Completely Randomized Design. In: *The Concise Encyclopedia of Statistics*. Springer New York; 2008:102-103. doi:10.1007/978-0-387-32833-1_72

4) Tukey JW. Comparing Individual Means in the Analysis of Variance. *Biometrics*. 1949;5(2):99. doi:10.2307/3001913

5) Ahmed S. What is Randomized Complete Block Design (RCBD)? The Open Educator . https://www.theopeneducator.com/doe/Randomized-Complete-Block-Latin-Square-and-Graeco-Latin-Square-Design/Randomized-Complete-Block-Design. Published 2020. Accessed March 29, 2022.

6) Zaiontz C. Randomized Complete Block Design. Real Statistics Using Excel. https://www.real-statistics.com/design-of-experiments/completely-randomized-design/randomized-complete-block-design. Published 2021. Accessed March 29, 2022.

7) Khurram H. Randomized Complete Block Design. Basic Statistics and Data Analysis. https://itfeature.com/design-of-experiment-doe/randomized-complete-block-design. Published April 14, 2019. Accessed March 29, 2022.

8) Banks DL, Fienberg SE. Statistics, Multivariate. In: *Encyclopedia of Physical Science and Technology*. Elsevier; 2003:851-889. doi:10.1016/B0-12-227410-5/00731-6

9) Appa G, Magos D, Mourtos I. An LP-based proof for the non-existence of a pair of orthogonal Latin squares of order 6. *Operations Research Letters*. 2004;32(4):336-344. doi:10.1016/j.orl.2003.10.010

10) The Pennsylvania State University 4.3 - the latin square design: Stat 503. PennState: Statistics Online Courses. https://online.stat.psu.edu/stat503/lesson/4/4.3. Accessed March 29, 2022.

11) Gao L. Latin squares in experimental design . http://compneurosci.com. http://compneurosci.com/wiki/images/9/98/Latin_square_Method.pdf. Published December 10, 2005. Accessed March 29, 2022.

12) Katsileros A. Λατινικό Τετράγωνο. Εργαστήριο Βελτίωσης Φυτών και Γεωργικού Πειραματισμού. Γεωπονικό Πανεπιστήμιο Αθηνών https://www.aua.gr. https://www.aua.gr/katsileros/wp-content/uploads/2020/06/lab8.pdf. Published June 4, 2020. Accessed March 29, 2022.

13) Ott L, Longnecker M. *An Introduction to Statistical Methods and Data Analysis*. Australia: Brooks/Cole Cengage Learning; 2010; 1095-1101.

14) The Pennsylvania State University14.3 - the split-plot designs: Stat 503. PennState: Statistics Online Courses. https://online.stat.psu.edu/stat503/lesson/14/14.3. Published 2022. Accessed March 29, 2022.

15) Μενεξές Γ. Γεωργικοί Πειραματισμοί Χωριστού Σχεδίου: ομάδες με Υποομάδες. docplayer.gr. https://docplayer.gr/29650506-Georgikoi-peirauatisuoi-horistoy-shedioy-ouades-ue-ypoouades-split-plot-plot-designs-r-georgios-menexes.html. Published November 26, 2010. Accessed March 29, 2022;8

16) Kuehl RO. *Design of Experiments: Statistical Principles of Research Design and Analysis*. 2nd ed. Brooks/Cole; 2000;pp. 469-472.

17) The Pennsylvania State University. 13.2 - the anova table: Stat 415. PennState: Statistics Online Courses. https://online.stat.psu.edu/stat415/lesson/13/13.2. Published 2022. Accessed March 29, 2022.

18) Grafton K. Split-split plot arrangement - NDSU. ndsu.edu/faculty. https://www.ndsu.edu/faculty/horsley/spspplot.pdf. Published December 2, 2003. Accessed March 29, 2022.

19) Tukey JW, Brillinger DR. *The Collected Works of John W. Tukey*. Belmont, CA: Wadswprth advanced Books & software; 1984.

20) Dubcovsky J. Topic 12. the split-plot design and its relatives [ST&D Ch 16].UCDAVIS Department of plant sciences. https://psfaculty.plantsciences.ucdavis.edu/agr205/Lectures/2011_Lectures/L12a_SplitPlot.pdf. Published February 22, 2011. Accessed March 29, 2022.

21) Tech AGRONI. Strip plot analysis using R. AGRON Stats. https://agroninfotech.blogspot.com/2018/06/strip-plot-analysis-using-r.html#import-data. Published July 5, 2021. Accessed March 29, 2022.

22) Sehgal K. Split plot and strip plot designs - drs.icar.gov.in. https://drs.icar.gov.in. https://drs.icar.gov.in/Electronic-Book/module4/4Split%20and%20Strip.pdf. Published August 10, 2004. Accessed March 29, 2022.

23) Tech AGRONI. Strip plot analysis using R. AGRON Stats. http://agroninfotech.blogspot.com/2018/06/strip-plot-analysis-using-r.html. Published July 5, 2021. Accessed March 29, 2022.

24) Shalabh . Chapter 6 split-plot and strip-plot designs analysis of ... http://home.iitk.ac.in/~shalab/anova/WordFiles-Anova/AdditionalTopics/Chapter6-ANOVA2-Split-StripPlotDesigns.pdf. Published 2015. Accessed March 29, 2022.

25) Keppel G. *Design and Analysis: A Researcher's Handbook. 3rd Ed.* Englewood Cliffs NJ: Prentice Hall; 1991.

26) Topic 13. Analysis of Covariance (ANCOVA, ST&D Chapter 17). UCDAVIS Department of plant sciences. https://psfaculty.plantsciences.ucdavis.edu/agr205/Lectures/2011_Lectures/L13_ANCOVA.pdf. Published March 14, 2015. Accessed March 29, 2022;3

27) Huang W-M. ANCOVA (Analysis of Covariance). https://www.lehigh.edu/~wh02/ancova.html. Published May 12, 2018. Accessed March 29, 2022.

28) Newsom J. Analysis of covariance (ANCOVA) - Portland State University. Psy 522/622 Multiple Regression and Multivariate Quantitative Methods. http://web.pdx.edu/~newsomj/mvclass/ho_ancova.pdf. Published January 19, 2021. Accessed March 29, 2022.

29) Overall JE. Letter to the editor: the use of inadequate corrections for baseline imbalance remains a serious problem. *Journal of Biopharmaceutical Statistics*. 1993;3(2):271-276. doi:10.1080/10543409308835066

30) Bonate PL. *Analysis of Pretest-Posttest Designs*. Chapman and Hall/CRC; 2000. doi:10.1201/9781420035926

31*) Paul Vogt, R Burke Johnson.* Dictionary of Statistics & Methodology : A Nontechnical Guide for the Social Sciences. *Sage; 2011.*

32) Antony J. Full Factorial Designs. In: *Design of Experiments for Engineers and Scientists.* Elsevier; 2014:63-85. doi:10.1016/B978-0-08-099417-8.00006-7

33) Prasanta Sahoo, Tapan Kr. Barman, in Mechatronics and Manufacturing Engineering, 2012 Pages 159-226 https://doi.org/10.1533/9780857095893.159.

34) Das AK, Dewanjee S. Optimization of Extraction Using Mathematical Models and Computation. In: Computational Phytochemistry. Elsevier; 2018:75-106. doi:10.1016/B978-0-12-812364-5.00003-1

35) Factorial Design - Statistical Experimental Design: Pharmaceutical Engineering. pharmacy180.com. http://www.pharmacy180.com/article/factorial-design-2693/. Accessed March 29, 2022.

36) Zach. Two-way ANOVA: Definition, formula, and example. Statology. https://www.statology.org/two-way-anova/. Published November 30, 2021. Accessed March 29, 2022.