Proceedings from the training on introduction to R software for forest statistics and modelling

Bangladesh Forest Department

15 – 19 November 2015

UN-REDD
PROGRAMME

Food and Agriculture Organization of the United Nations
UNDP
UN environment

The UN-REDD Programme, implemented by FAO, UNDP and UNEP, has two components: (i) assisting in developing countries to prepare and implement national REDD strategies and mechanisms; (ii) supporting the development of normative solutions and standardized approaches based on sound science for a REDD instrument linked with the UNFCCC. The programme helps empower countries to manage their REDD processes and will facilitate access to financial and technical assistance tailored to the specific needs of the countries.

The application of UNDP, UNEP and FAO rights-based and participatory approaches will also help ensure the rights of indigenous and forest-dwelling people are protected and the active involvement of local communities and relevant stakeholders and institutions in the design and implementation of REDD plans.

The programme is implemented through the UN Joint Programmes modalities, enabling rapid initiation of programme implementation and channelling of funds for REDD efforts, building on the in-country presence of UN agencies as a crucial support structure for countries. The UN-REDD Programme encourage coordinated and collaborative UN support to countries, thus maximizing efficiencies and effectiveness of the organizations' collective input, consistent with the "One UN" approach advocated by UN members.

The UN-REDD Bangladesh National Program is implemented by the Bangladesh Forest Department under the leadership of Ministry of Environment and Forests. United Nations Development Program (UNDP) and Food and Agriculture Organization (FAO) are the two implementing partners.

CONTACTS:

Rakibul Hassan Mukul
Project Director
UN-REDD Bangladesh National Programme
Bangladesh Forest Department
Email: pd-unredd@bforest.gov.bd

Matieu Henry
Chief Technical Advisor
Food & Agriculture Organization of The United Nations
Email: matieu.henry@fao.org

Suggested citation: **Costello, L. & Garcia Perez, J.** 2015. Proceedings from the training: Introduction to R software for forest statistics and modelling. 15-19 November 2015, Dhaka, Bangladesh Forest Department, Food and Agriculture Organization of the United Nations.

Disclaimer

This report is designed to reflect the activities and progress related to UNJP/BGD/057/UNJ UN-REDD Bangladesh National Programme. It does not reflect the official position of the supporting international agencies including FAO and UNDP should not be used for official purposes. Should readers find any errors in the document or would like to provide comments for improving the quality they are encouraged to contact one of above contacts.

Executive Summary

Estimating tree and forest biomass is a cornerstone of forest resource assessment. In addition to its application in forest-related commerce, forest resource assessment is becoming increasingly crucial in light of continued global deforestation and climate change. Developing a forest monitoring system is a comprehensive process that includes the collection, analysis and dissemination of forest-related data and the derivation of information and knowledge at regular intervals. It is only by having a comprehensive, reliable and transparent data collection and analysis process so that informed decisions can be made, communicated and defended on scientific grounds.

FAO is currently assisting Bangladesh Forest Department (BFD) to establish a national forest monitoring system. Strengthening national capacities and ownership on data management and analysis is essential to ensure the sustainability of the project. The amount of data collected on the status of forest resources is continuously increasing and improved data management tools are necessary. Such tools are fundamental to ensure the sustainability of the forest monitoring system, but also to ensure the transparency of the data analysis process and its improvement over time. Developing the capacity of BDF in statistical analysis with R software is therefore an important project activity.

R is a free and open source programming language and environment for statistical computing and graphics that will be used to assist the project’s data processing and analysis components. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. The R language is widely used among statisticians and data miners for developing statistical software and data analysis. R is highly extensible through the use of user-submitted packages for specific functions or specific areas of study. In the context of forest assessment and forest monitoring system, R software can be used for various purposes such as the analysis of tree inventory data, soil analysis, biomass assessment, allometric equation development, tree species quality control procedures, remote sensing, Geographic information system etc.

Sample data used in the training was derived from previous FAO / UN REDD led trainings related to field data collection drawn through the Open Foris Collect platform. Participants we shown how to write basic code in the R language, run functions and create charts to identify outliers or erroneous data. The final session of the training demonstrated an approach to calculate allometric equations through R.

Twelve participants (eight male and four female) from FD, BBS, FAO and Bangladesh Forest Research Institute, as well as professors from Khulna University, University of Chittagong and Shahjalal University of Science & Technology attended the event. The diverse stakeholder participation aims to foster a broad technical knowledge base providing on-going collaborations to support forest monitoring in Bangladesh.

Contents

1. Introduction

R software (http://www.r-project.org/) is a key tool to develop robust statistics and modelling on forestry data and is becoming of increasing importance in the academic and forest management world. This software is completely free and open-source and an extensive support is provided by a sizable community of users and experts on internet. It is being used as key language on forest calculation tools developed by FAO (http://www.fao.org/forestry/fma/openforis/en/) and in several training documentation such as the application examples of the Manual for building tree volume and biomass allometric equations (http://globallometree.org/docs/).

The flexibility of the R environment make it a powerful tool for forest inventory data analysis. The obstacle is that learning R imposes a steep learning curve for people not familiar with programming or R's specific language and structure.

A detailed presentation of the R software's language and its functionalities, and its application to forest statistics and modelling with many practical examples was presented. This included important steps toward the selection of the methods and technologies to be used for calculating forest statistics and emission factors in the forestry sector in Bangladesh.

2. Objectives

The general objective of the training course is to improve Forest carbon stocks estimates in Bangladesh. The specific objectives of the training are:

- Introductory course to the R programming language as a statistical tool in forest biomass assessment, targeting national forestry staff.
- Present the key objects, functions and graphic development linked to forest information
- Increased knowledge on statistical methods for biomass modelling based on the use of tree models in national forest inventory databases with the use of R
- Identification of gaps, needs and future steps regarding the improvement of forest biomass assessment.

3. Summary of Exercises

3.1 The R project

The R Environment

R is an integrated suite of software facilities for data manipulation, calculation and graphical display. While R can be thought of as a statistics system, it is in fact an environment within which statistical techniques are implemented. The term "environment" is intended to characterize it as a fully planned and coherent system, rather than a rigid set of specific and inflexible tools, as is frequently the case with other data analysis software.

Packages

Contributing to its flexibility, and indeed a major benefit of the system is its ability to be expanded by the use of packages. Packages are collections of R functions, data, and compiled code that can be imported and run to carry out specific functions. The packages cover a wide range of modern statistics, often developed and made available for specific topics including forestry (See more about this topic at https://www.r-project.org/about.html).

CRAN

Packages can be loaded in R via the “Comprehensive R Archive Network” (CRAN). CRAN is a network of web servers around the world which carry identical material, consisting of packages, documentation for R, and other R resources. The multiple CRAN websites are provided to speed up downloads by minimize network load. The central CRAN repository can be accessed at: https://CRAN.R-project.org/.

From CRAN, you can obtain the latest official release of R, useful code contributed by the development community, access to documentation on R and the R Bug Tracking system.

What R does and does not do

R provides	R does not provide
Matrix algebra;	It is not a database, but it connects to databases;
Hash tables and regular expressions;	It is not a spreadsheet, but it connects to spreadsheets;
Reading and manipulating data;	It does not provide commercial support.
Computation;	Windows, COM to provide graphical interfaces;
Conducting statistical analyses;	It does not provide a graphical interface, but it uses Java, TclTk and other graphical interface platforms
Graphics and tables;	
Displaying the results.	
Programming language: loops, subroutines, functions, etc.;	

R Studio

R Studio is a set of integrated tools designed to help you be more productive with R. It includes a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management. R studio is the interface that connects with the R software. The program is free and available via the following link: https://www.rstudio.com/products/rstudio/download/

User interface in R Studio

The R Studio interface is made up on four components or windows:

- Command: The location where the script are written
- Console: this is where the code will be returned
- Workspace (Environment/History): all variables created / imported are available in the workspace.
- Files/Packages/Help window: Provides a directory of available files, packages and help commands.

3.2 Establishing a workspace

Importing Data

R can read a variety of data sources; however, it is recommended to import data in either .csv or .txt formats. This avoids potential versioning errors that may occur with excel files.

When importing .csv or .txt files, the data format (related to column delimitation) must be considered. Specifying whether the data is comma, space or tab delimited must be a consideration during data import.

Working directory

The working directory is the location where one’s relevant data (preferably in .csv or .txt) is located.
Two key functions define the working directory:

```
> setwd("YOUR FILE PATH") #Note that R uses / in the file path, not \.
> getwd() #Running this command will tell you where you working is located
```

The working directory and import of datasets can also be established by using the “Import Dataset” function available in the Workspace.

Installing and calling CRAN packages

Packages are installed via the Tools>Install Packages menu. From here the CRAN repository can be defined and accessed. Packages only have to be loaded once. The following code is used to run the package in your workspace: library(package_name) or require(package_name)

3.3 Basic functions in R

R is a programming language and is therefore has a particular syntax that must be strictly followed. Below are some considerations to assist in basic code building.

- R is case sensitive, therefore “TRUE” is different to “true”
- Use the left arrow “<-” (“less than” sign followed by a dash) and the c function (for concatenate) and thereby create a “vector”Use # to write explanatory text that will not be considered as code
- Use ?help to find help. For example “?mean” will provide help on calculating mean. Other related functions are apropos(mean) or help(mean)

Some other useful functions in R include:

```
> ls() #List all objects in workspace
> rm(X) # Delete the “X" object.
> rm(list=ls()) # Delete the workspace

#Diverting output as a text file
        > sink(“filename.text")
        > R command(s)
        > sink()

> q() # Quit R
```

Getting help

There are a range of help functions that can be used to find explanations on R code.

```
> help.start()                          # Online help
> help(mean) or ?mean                   # Get help for a function
> help.search(“solve”) or ??solve       # Get help for a topic
```

Save, create, open a script

- Text files containing a list of R commands and given the extension .R. This file type can be saved for later use.
- To create a script go to the File menu > New File > R script
- To open a script go to the File menu > Open Script
- To load independent R files whose functions will be later used: source(“Filepath/file.R”)

Save and restore a workspace

The functions you've created, calculations your run or packages loaded in stored within the memory of your workspace.

To save, go to file menu > Save Workspace

To load workspace go to File menu > Load Workspace

Calculations with numbers

Basic calculations can be carried out in R using standard symbols including +, - , <, >, =< etc. However, instead of using = to calculate the equations, you run the code through the console to generate the answer. For example, the following function creates an object (or vector) called "a" and calculates a+5

```
> a <- c(1,2,3,4)
> a
[1] 1 2 3 4
> a + 5
[1] 6 7 8 9
```

Other useful calculating functions

```
> 2 + 3 * 5       # Note the order of operations.
> log (10)        # Natural logarithm with base e=2.718282
> 4^2             # 4 raised to the second power
> 3/2             # Division
> sqrt (16)       # Square root
> abs (3-7)       # Absolute value of 3-7
> pi              # The mysterious number
> exp(2)          # exponential function
> # This is a comment line
```

3.4 Different types of objects

R is a programming language based on objects. When using R you must define each element to be used as an object. For example, to get data into R you need to define a dataset as an object. For example, you can define an object called "mydata" from an external dataset myata.txt using the following code:

```
> mydata <- read.table("C:/file_path/mydata.txt")
```

Data types

There are numerous different data types in R. These include:

- Numeric # numbers/integers
- Character # text/strings versus factors. This is particular relevant. Factor variables establish the different elements as categorical variables, making it more difficult to add or extract new categories. They are essential to aggregate results but a string variable should be defined as a factor (i.e., function as.factor(variable_name) only when one is sure that the categories are definite
- Logical # logical values for TRUE/FALSE. These can be referred to a boolean or binary values.

Arrays

An array can be considered as a multiply subscripted collection of data entries (numeric, character or logical). An array can be created by the array() function

Vectors

Collection of numeric, character or logical values (one-dimensional array). Functions and logical operators work on each element of the vector. In the following example the values 1, 2 and 3 are assigned to a vector called "vector_name"

```
> vector_name <- c(1, 2, 3)      # Create the vector
> vector_name                    # print the vector
[1] 1 2 3                        # the value of the vector
```

Matrices

Two dimensional structures with data of same type and consists of rows and columns(two- dimensional array)

```
> matrix (1:12,3,4)              # The first argument creates a matrix with the values 1
     [,1] [,2] [,3] [,4]         # to 12, the second defines the structure of the array
[1,]    1    4    7   10         # with 3 rows and 4 columns
[2,]    2    5    8   11
[3,]    3    6    9   12
```

Data frames

A data frame (or data set) is a special type of list in which each variable should have the same number of observations. It is similar to matrix, but can store different data types. More information on data frames is provided below in section 3.6.

3.5 Import table

Data can be imported in various formats. The most common are .csv and .txt. Excel files can be imported however this is generally not recommended because excel files (.xls) create more room for importation errors. Provided below are is the basic syntax for importing in three different formats.

```
> mydata <- read.csv("tree2015.csv") #tree2015.csv was an example file from two plots in
Bangladesh, containing dbh, height, bole height, plot ID and species name
> mydata <- read.table(file='tree2015.txt')
> mydata <- read.xls("name_of_file.xls")
```

For each of the above methods additional information can be added to more specifically define the importation format. This may include:

- Specifying the delimiter for txt files for example: sep='\t' # Specifies the separator (delimiter) as tab.
- Defining that the first line of data is the column headers: header = TRUE
- Defining how R should handle missing values (empty cells): na.strings=c("None","NA")
- Defining string values as factors: stringsAsFactors=FALSE

The additional pieces of information added to a statement are called 'arguments'. The arguments help to define how the data is formatted or stored and therefor has implications for how the data is handled, calculated and managed.

Management of NA values

It is common for databases to include missing values (empty cells). By default R defines missing values with the character string “NA” however this can be changed using the argument na.strings. In many functions (sum, mean max, etc..) the presence of an NA may, or may not, cause trouble. To ignore NA in the function use na.rm(), or to remove NA values, use na.omit().

```
data1    <-    read.table(file='/files/myFile.txt',    sep='\t',    header    =    TRUE,
na.strings  =  c("None"  ,  "No","-","  "  )) #This  code  finds  the  various  values  that  may
indicate missing data as defined by the data collector, in this case: "None" , "No","-","
"

mydata[is.na(mydata)] <- 0 # Replace the NAs with zeros
```

Defining string values as factors

By default data.frame() casts character vectors into factors. You can chose to keep character vectors as is with the option stringsAsFactors

3.6 Operations with data frames

A data frame is a two-dimensional data structure in which every column may have a different type. Data frames are lists of columns that are vectors of identical lengths. The function data.frame() is used for creating this kind of structure. The structure of a data frame is displayed below.

```
        Species dbh height  WD
Tree_1      Euc  82    185 454
Tree_2     Term  81    178 678
Tree_3      Pin  55    165 443
Tree_4      Ced  65    171 234
Tree_5      Euc  68    172 456
Tree_6      Ced   3     76 500
Tree_7      Pin  30    200 430
```

Useful operators used with data frames

The following operators can be used to provide an overview of your data without calling the entire dataset through the consol.

- dim(mydata): Provides the dimensions of data frame
- head(mydata): shows beginning of data the frame
- tail(mydata): shows the end of the data frame
- names(mydata): shows the list of column named
- str(mydata): shows the data types of the variables of the dataframe
- View(mydata): Creates a spreadsheet-style data viewer on a matrix-like R object

Manipulating data frames

Data frames can be manipulated in various ways. During the training, participants were shown how to create a new column, create a subset of data and aggregate calculations within the population. Some example code is shown below.

```
> mydata$radius = mydata$dbh/2 # Creates a new column called ‘radius’ with the
  calculation dbh/2
```

```
> mydata20 <- mydata[1:20,] # Selects the 20 first rows
> aggregate(dbh ~ height_bole, data=mydata, FUN = mean) #calculates a numerical
  summary for subsets “dbh” and “height_bole”
```

3.7 Building functions

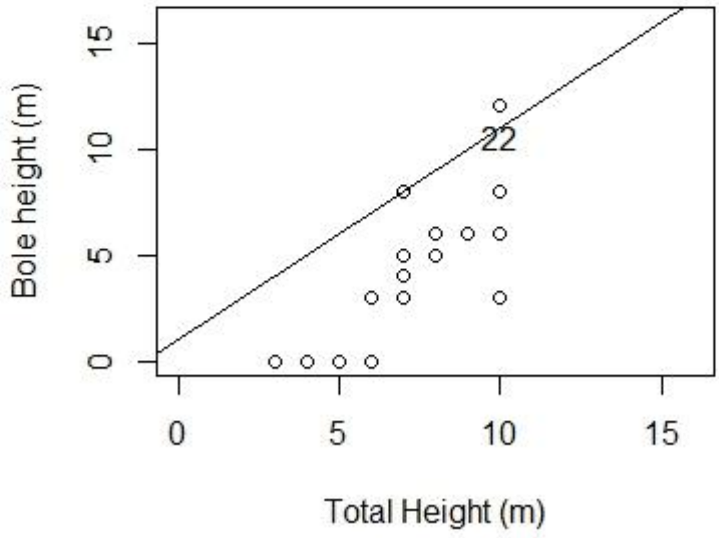
In addition to packages that may contain specific operators for data interrogation, we can also build functions to run specific calculations or analysis. In the example below we developed a function to calculate above ground biomass.

```
Build a function to calculate Aboveground biomass (AGB) based on an equation
(i.e., AGB=0.567 * H * DBH^2)

myFunction=function(DBH,H){
  toto=0.567*H
  toto=toto*DBH^2
  return(toto)
}
```

3.8 Graphic representation

Graphical representation can be quickly generated through R. In the first example, we used a scatter plot with a line of fit to detect data anomalies, in this case, records where bole height exceeded total height. We then used the “identify” function to manually identify (by clicking on it) and label the outlier, point 22.

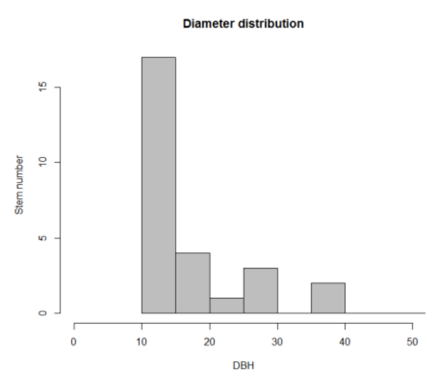
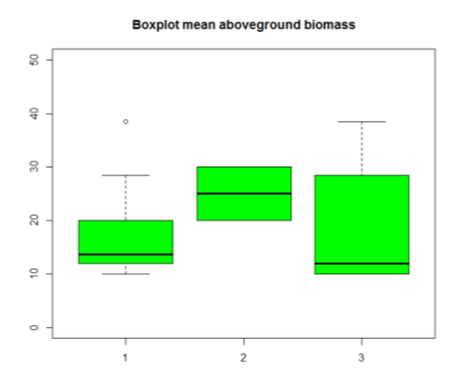
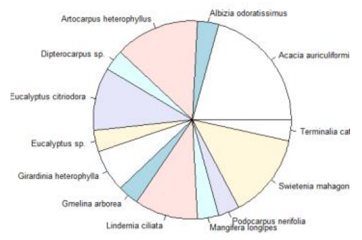
```
plot(mydata$height_tot, mydata$height_bole, xlab="Total Height (m)", ylab="Bole height
(m)", xlim=c(0,16),ylim=c(0,16))
abline(1,1) # Where xlab = x label, xlim = limit of extent, abline = line of identity

identify(mydata$height_tot, mydata$height_bole, labels=row.names(mydata)) # identify
points
```



Other plot types

Numerous other types of graphs can be created in R. In their most basic form, the code to create the plots can be quite simple (a), but can come increasingly explanatory with the addition of colours and lables (b) and with more defined information (c). With additional information the code becomes more complex. Base codes for example plots include:

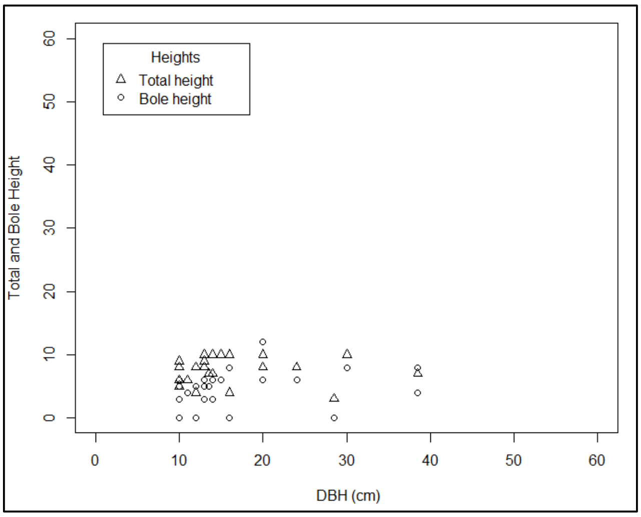
- Histogram: hist(mydata$variable_name)
- Box plot: boxplot(mydata$variable_name)
- Pie: pie(x, labels = names(x)) # where x = a vector of non-negative numerical quantities. The values in x are displayed as the areas of pie slices.

 	 	
hist(mydata$dbh)	boxplot(mydata$dbh ~ mydata$tract_tract_no,ylim=c(0,50), col="green")	pie(table(mydata$tree_species_scientific_namelabels=levels (as.factor(mydata$tree_species_scientific_name)),cex=0.8)

Creating Legend

A legend can be added to the graph to provide additional context. The arguments used to create a legend define the location, legend title, variable names and point shape as shown in the below examples.

```
> legend("topleft",inset=.05, title="Heights",pch=c(2,1),
        legend=c("Total height","Bole height"))
```



3.9 Linking allometric equations with data to calculate volume

One key element of NFMS is the development of country-specific emissions factors for different land-uses types and forest classes, allowing country to report on carbon stocks changes. To develop country-specific emission factors, countries need to undertake field measurements and use collected forest data (DBH, height, wood density) to determine carbon stocks. Upon several repetitive measurements of these standards NFI parameters, countries are enabled to develop allometric equations to assess the carbon content of the measured biomass.

In this exercise, a demonstration on the use of allometric equations in volume calculations was provided. The demonstration provided some context into the next phase of more complex R analysis. The demonstration covered the following:

- Harmonizing units in AEs and data
- Modifying text in allometric equations
- Parsing and evaluation of functions (i.e., A.E.'s)
- Final calculation and plotting of results

4. Recommendations for next steps

Developing capacities in R among Bangladesh's forest management and research communities is a critical first step in establishing a robust data management system for the countries emerging pursuits into national forest monitoring. With the increasing popularity of R in forest related data management systems, such as the Open Foris platform, understanding the basic functions will provide a foundation that data managers can build on as new applications and systems become available and operational.

The concepts presented in this training were pitched at an introductory level. One of the training's strengths was in its applicability to the Bangladesh context. In moving forward, the use of R applied training context, such as calculations for allometric equations, will provide the necessary context and environment for further institutionalisation of R software in Bangladesh's national forest monitoring system.

Appendix 1. Agenda

Date	Topic	Speaker/facilitator
Day 1	**Opening session**	
9.00	Registration	
9:15	Welcoming remarks Training objectives	NPC
9:30	Importance of biomass estimates and allometric equations	Javier Garcia Perez
10:15	Pause	
	Session 1: theoretical key points on allometric equations	
10:45 – 12:15	○ What is and what isn't R, ○ The context of Bangladesh Forest assessment and the need for R, ○ Questions/Answers.	Javier Garcia Perez
	Session 2: Introduction to R	
14:00 – 17:00	○ Installation of the software, ○ Presentation of on-line resources, ○ Presentation of the interface, ○ Definition of the working directory, ○ Basic calculations.	Javier Garcia Perez
Day 2	**Session 3: The different objects in R**	
8:30 – 12:00	○ Variable, ○ Numbers, ○ Text, ○ Vectors, ○ Matrices, ○ Lists, ○ Data frames.	Javier Garcia Perez
	Session 4: Calculation on data frames	
14:00 – 17:00	○ Import and export of forest inventory data as data frames, ○ Selection of rows and columns, ○ Creation of new columns, ○ Calculation of averages, calculations based on several columns.	Javier Garcia Perez
Day 3	**Session 5: Aggregation of tree data to plot and forest estimates**	
8:30 – 12:00	○ function aggregate() , ○ Calculate basal area and biomass per ha at plot and forest type level.	Javier Garcia Perez
	Session 6: Practice	

14:00 – 17:00	○ Import Forest inventory data, ○ Create new columns, ○ Calculate information at plot level,	Javier Garcia Perez
Day 4	**Session 7: graphics**	
8:30 – 12:00	○ Manual detection of outliers ○ Aggregation and calculations within the population ○	Javier Garcia Perez
	Session 8: R for modelling tree allometric equations	
14:00 – 17:00	○ Simple plots: histograms, scatter, pie charts, boxplots, and legends ○ Simple plots with R ○	Javier Garcia Perez
Day 5	**Session 9: Conclusion**	
8:30 – 10:30	○ Linking allometric equation database to field data with R	
10:30	Break	
11:00 – 12:00	○ Conclusion, group photo, closing remarks.	

Appendix 2. Participant List

No.	Name	Department
1	Mr. Md. Akhter Hassan Khan	BBS
2	Mr. S. M. Zahirul Islam	BFRI
3	Ms. Mariam Akhter	FAO
4	Ms. Sharmin Shabnam	FAO
5	Mr. Md. Tariq Aziz	Forest Department
6	Ms. Ummey Habiba	Forest Department
7	Mr. Hossain Mohhamad Nishad	Forest Department
8	Mr. Md. Baktiar Nur Siddiqui	Forest Department
9	Ms. Afroza Begum	Forest Department
10	Mr. Sourav Das	Shahjalal University of Science & Technology
11	Mr. M. M. Abdullah Al Mamun	University of Chittagong
12	Md Raqibul Hasan Siddique	Khulna University

Appendix 3. Evaluation

Nine participants completed the online survey.

How often do you participate in training related to forest monitoring?		
First time	8	89%
1-3 every year	1	11%
More than 3 per year	0	0%
Regularly (approximately one per month)	0	0%
I would describe myself as:		
A professor/academic	2	22%
A student	1	11%
Forest Department staff	4	44%
Government staff (outside Forest Department)	2	22%
NGO staff	0	0%
Private consultant	0	0%
My professional background relates most closely to:		
Forester	2	22%
GIS/RS	3	33%
Statistics	2	22%
Social survey/assessment	1	11%
Economics	0	0%
Natural Resource Management	6	67%
Ecology	2	22%
My years of relevant experience is:		
1-2 years –	5	56%
3-5 years –	1	11%
5-7 years –	1	11%
8-10 years –	0	0%
More than 10 years –	2	22%
The training was relevant to my daily work		
Strongly agree –	1	11%
Agree –	5	56%
Neutral –	3	33%
Disagree –	0	0%
Strongly disagree –	0	0%
The training met my expectations in terms of the content and learning outcomes		
Strongly agree –	1	11%
Agree –	7	78%
Neutral –	1	11%
Disagree –	0	0%
Strongly disagree –	0	0%

The resource person presented information in a way that i could understand and was easy to follow		
Strongly agree –	3	33%
Agree –	6	67%
Neutral –	0	0%
Disagree –	0	0%
Strongly disagree –	0	0%
I feel confident to be able to carry out the tasks described in the training without supervision.		
Strongly agree –	1	11%
Agree –	6	67%
Neutral –	2	22%
Disagree –	0	0%
Strongly disagree –	0	0%

	True –	False –
It is recommended to avoid importing excel spreadsheets. It is better to export the data from Excel in tab-delimited or comma-separated form, and use read.table or read.csv to import it into R.	8 (100%)	0 (0)
Of the two statements below, number (1) is correct: (1) setwd("C:\Users\Documents\R\Bangladesh2015\Data") (2) setwd("C:/Users/Documents/R/Bangladesh2015/Data")	2 (25%)	5 (63%)
While importing a table, the empty fields in numeric columns are regarded as zero (0) and summed accordingly which can confuse results.	5 (63%)	3 (37%)
The symbols -> and = provide similar but different functions	4 (50%)	4 (50%)
Installation and use of packages required two processes: (1) installing the package via the tools menu and (2) activating the package by running the appropriate code, for example: library(package_name)	7 (88%)	1 (12%)
The operation: mydata20 <- mydata[1:20,] will select the first 20 rows and all columns	6 (85%)	1 (15%)
Assuming 'height_bole' values are stored in the 10th column, the following two statement will produce the same result: (1)mydata[complete.cases(mydata[10,]),] (2) mydata[complete.cases(mydata$height_bole),]	3 (37%)	5 (63%)