APPENDIX
R SCRIPT CODE USED TO RUN HOUSEHOLD SURVEY ANALYSIS, ADAPTED FOR WIDE APPLICABILITY

NOTES ON USING R SCRIPT CODE

This appendix provides relevant code for conducting the household data analysis outlined in Chapters 3 to 8 of this book. All of our analysis was completed using the R statistical programming environment (R Core Team 2013). R is freely available software that can be downloaded without payment from numerous servers around the world (see http://www.R-project.org). The program is extremely powerful and versatile, and is rapidly becoming the preferred statistical software for statisticians and scientists. Because it is open source, all the analysis we have performed can be repeated by anyone in the world with a computer and internet access, regardless of financial resources.

R is sometimes difficult for new users. We recommend using accompanying software RStudio (**https://www.rstudio.com/**) as the user interface. RStudio greatly improves the user experience and is much easier for new users to navigate. A valuable resource for learning how to use R is the Quick R website (**http://www.statmethods.net/**), which also includes further training resources.

Finally, the Australian Centre for International Agricultural Research (ACIAR) in conjunction with the SIMLESA project has developed a free online statistical training course called *BEST for Africa: Bespoke eStyle Statistical Training for Africa and South Asia*. The course is specifically tailored to agricultural research in Africa and South Asia and provides training and instruction for R. It can be accessed at **http://yieldingresults.org.** It is our hope that this report and the accompanying R code will allow others to easily conduct similar studies of household diversity.

OBTAINING R CODE FOR HOUSEHOLD DATA ANALYSIS

The R code used for household data analysis described in this publication may be obtained from:

1. GitHub repository at **https://github.com/aciar/SIMLESA**
2. text file published with the online version of this publication, at **www.aciar.gov.au/publication/household-diversity**
3. the following pages of this document if you are reading the online version of this publication.

Coding is provided for:

1. Principal component analysis
2. Cluster analysis
3. Analysing cluster data
4. Generating heatmaps

REFERENCES

R Core Team 2013. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. Available at: **http://www.R-project.org**.

1. R CODE FOR PRINCIPAL COMPONENT ANALYSIS

The following code (below this sentence) can be copied and pasted into the R console.

```
# Factor Analysis of HH survey data
# Instructions.
#
# 1. Run R, change ‘directory’ to your work area (i.e. computer folder where the data is)
# 2. “Source R code” -> P4-2.R (this file)
# 3. Edit your household data file to save as a comma delimited csv file
# 4. Then edit below the name of your .csv file (the spreadsheet with your data) and the list of variables to
include into the ‘keep1’ list
# make sure you type the upper and lower cases for your variables the same as in the csv file
# 5. Interpret the Factor.txt file (see Chapter 2) and select the names of the variables you
# will use for the cluster analysis
# 6. You will need to download the ‘psych’ and ‘nFactors’ R packages to use this script
# (see Quick R for information on downloading packages)

rm(list=ls()) # removes any lists and dataframes already loaded in to R

# Change the file name
HH_survey_data<- “YOUR DATA FILE NAME HERE.csv”

# Change the name of the directory the name of a new folder where your data is located
# You will need to create a new folder on your computer with the same name (letters are case sensitive)
# This is where the outputs of your analysis will be saved
# Example provided for outputs from Western Kenya
out.dir<-”Kenya_West”

# List of variables you want to include for your principal component analysis
# Remember for each variable you should include at least 10 observations
# (i.e. 10 variables requires at least 100 households)
# Principal Component Analysis can only properly run with around 20 variables at most
keep1<-c(“variable 1”,”variable 2”,”variable 3”,”variable 4”,”variable 5”,”variable 6”,”variable 7”,
  “variable 8”,”variable 9”)

# read data
df<-read.csv(HH_survey_data)

# Calculate any extra variables needed
# (example given for new plot size calculated from plot length and plot width variables)
#df$PLOTSIZE<-df$PLOTLENGTH * df$PLOTWIDTH
#keep1 <- c(keep1, “PLOTSIZE”)
# Any extra variables ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
# only keep variables we’re interested in
df<-df[ , names(df) %in% keep1]

## ensure all data is numeric - convert factor variables
for (x in names(df)) {if (is.factor(df[[x]])) {cat(x,” is a factor - now transformed\n”); df[[x]]<-xtfrm(df[[x]])}c}

# replace any missing values
df<-as.data.frame(apply(df,2,function(x) {x[is.na(x)] <- mean(x,na.rm=T) ; x} ))

for (x in names(df)) {if (sd(df[[x]]) == 0 || is.na(sd(df[[x]]))) {cat(x,” has no variance\n”); df <-
df[,!(x==names(df))] }}

# Principal Components Analysis creating 15 principal components (i.e. artificial variables)
cat(“Doing PCA\n”)
library(psych)
# change the number “15” in the code below this line if you want to adjust the number of principal
components to be created from your data
pc <- principal(df, nfactors=min(ncol(df),15), rotate=”varimax”) #rotated
sink(paste(out.dir,”/factors.txt”,sep=””))
print(summary(pc)) # print the variance accounted for by each principal component
print(loadings(pc)) # pc loadings for each observed variable
sink()

# create scree plot (to help decide how many PCs to keep)
png(paste(out.dir,”/scree plot 1.png”,sep=””))
plot(pc$values,type=”l”,main=””, xlab=”# factors”, ylab=”Eigenvalue”) # scree plot
dev.off()

# Create second scree plot of eigenvalues (look for the elbow in the line)
library(nFactors)
ev <- eigen(cor(df))
ap <- parallel(subject=nrow(df),var=ncol(df), rep=100, cent=.05)
nS <- nScree(ev$values, ap$eigen$qevpea)
png(paste(out.dir,”/ scree plot 2 - elbow.png”,sep=””))
plotnScree(nS)
dev.off()

cat(“PC loadings\n”)
print(loadings(pc))
sink(paste(out.dir,”/ScreeValues.txt”,sep=””))
print (pc$values)
sink()
flush(stdout())
```

2. R CODE FOR CLUSTER ANALYSIS

The following code (below this sentence) can be copied and pasted directly into the R console.

```
# Household Cluster Analysis (to be run after PCA)
# Instructions.
# 1. Your household survey data file needs to have a variable for the ID of each household
# and this variable needs to be called “hhldid”
# 2. Run R, change ‘directory’ to your work area (i.e. computer folder where the data is)
# 3. Edit your household data file to save as a comma delimited csv file
# 4. Then edit the name of your .csv file (the spreadsheet with your data) in the code below
# 5. Edit the list of variables to include into the ‘keep1’ list
# make sure you type the upper and lower cases for your variables the same as in the csv file
# 6. Interpret the Cluster dendrogram plot (see Chapter 2) to identify how many clusters you
# want to keep (then run the cluster analysis again with that number)
# 7. You will need to download the ‘ape’ and ‘sparcl’ R packages to use this script
# (see Quick R for information on downloading packages)

# Set Working Directory (location of your datafile) – change example filepath in “” below
setwd(“C:/My Computer/SIMLESA Typologies/ country chapters/Kenya/Analysis”)

# load packages needed from library
library(ape)
library(sparcl)

# remove previously loaded lists and data from workspace
rm(list=ls())

# Load household data file (change name below to your file name - letters are case sensitive)
HH_survey_data<- “YOUR HOUSEHOLD DATA FILE.csv”

# Select the number of clusters to keep (currently set to 3). This should be adjusted after examining
dendrograms and all subsequent code run again.
nclust<-3

# list variables (from PCA + any additions) that you want to use in Cluster Analysis
keep1<-c(“variable 1”,”variable 2”,”variable 3”,”variable 4”,”variable 5”)

# read data
 dfAll<-read.csv(HH_survey_data)
 df<-dfAll[ , names(dfAll) %in% keep1]

 ## ensure all data is numeric - convert factor data
 for (x in names(df)) {if (is.factor(df[[x]])) {cat(x,” is transformed\n”); df[[x]]<-xtfrm(df[[x]])}}
 # replace any missing values with averages
 df<-as.data.frame(apply(df,2,function(x) {x[is.na(x)] <- mean(x,na.rm=T) ; x} ))

#create distance matrix using Euclidean distance
 d<-dist(scale(df), method = "euclidean")

 # Ward's Hierarchical Clustering of distance matrix
 fit <- hclust(d, method="ward")
 # apply number of clusters to keep
 c <- cutree(fit, k = nclust)

 # Create dendrogram plots
 plot(fit, xlab="Households")
# draw red rectangles around clusters – specify number by changing "3" in code below
rect.hclust(fit, k=3, border="red")
# Save dendrogram, (adjust plot dimensions – i.e. width and length - by changing the '500, 500' below)
png("FILENAME FOR SAVING DENDROGRAM HERE.png", 500, 500)
y = cutree(fit, 3)
# create dendrogram with each household coloured per cluster. Don't forget to change the plot title.
ColorDendrogram(fit, y = y, labels = names(y), main = "ENTER PLOT TITLE HERE",
      branchlength = 80)
dev.off()
# Finish export to excel
 dist.cluster <- function(i, distmat, clusters) {
   ind <- (clusters == i)
   return(rowSums( distmat[ind, ind]))
 }
 d1<-as.matrix(d)
 out<-data.frame(hhldid=dfAll$hhldid,cluster=c,dist=NA,cluster.rank=NA)
 for (i in unique(c)) {
             print (i)
  m<-dist.cluster(i,d1,c)
  out$dist[as.numeric(names(m))] <- m
  out$cluster.rank[as.numeric(names(m))] <- order(m)
 }

 write.csv(out, "FILENAME FOR SAVING DATA WITH HOUSEHOLD ID AND ALLOCATED CLUSTER.csv",row.
names=F)
dev.off(filename="FILENAME FOR FINAL COLOUR DENDROGRAM PLOT.png")

#####################################################
##########################
```

```
### Calculating file for medians of each observed variable given for each cluster groups
# Read data
dfAll<-read.csv(HH_survey_data)
for (x in names(dfAll)) {if (is.factor(dfAll[[x]])) {cat(x,” is transformed\n”); dfAll[[x]]<-xtfrm(dfAll[[x]])}}
dfAll<-as.data.frame(apply(dfAll,2,function(x) {x[is.na(x)] <- mean(x,na.rm=T) ; x} ))

statclus <- aggregate(dfAll[,-c(1,2)],list(c),median)
medianstats<-data.frame(Cluster=statclus[,1],Freq=as.vector(table(c)),statclus[,-1])
write.csv(medianstats, file=”FILENAME FOR NEW FILE WITH MEDIANS OF OBSERVED VARIABLES FOR EACH
CLUSTER.csv”)
```

3. R CODE FOR ANALYSING CLUSTER DATA

The following code (below this sentence) can be copied and pasted directly into the R console.

```
# Instructions
# 1. Before you can run this code, you need to add in a variable (i.e. column) into your
# survey data called ‘cluster’. This needs to have the cluster number input for every
# household. These cluster numbers can be taken from the output file created from the
# cluster analysis script
# 2. Run R, change ‘directory’ to your work area (i.e. computer folder where the data is)
# 3. Edit your household data file to save as a comma delimited csv file
# 4. Then edit the name of your .csv file (the spreadsheet with your data) in the code below
# 5. You will need to download the ‘psych’ R package to use this script
# (see Quick R for information on downloading packages)

### Cluster analysis - creating boxplots and descriptive statistics
#Analysing Clusters for describe by stats

# Set Working Directory (location of your datafile) – change example filepath in “” below
setwd(“C:/My Computer/SIMLESA Typologies/ country chapters/Kenya/Analysis/Cluster results”)

# Remove previously loaded data from workspace
rm(list=ls())
# load packages needed
library(psych)

# Name of your csv file, remember this needs to have cluster numbers in a column for
# each household (taken from earlier cluster analysis)
HH_survey_data<- “NAME OF YOUR DATAFILE HERE.csv”

# Change the name of the directory where you want the output files to be
# make sure there is a folder with the exact same name in the same location
# where your household data is
out.dir<-” OUTPUTS DATA FOLDER NAME HERE”

##########################################################
######################
### Run analysis for boxplots and descriptive statistics

# make boxplots
cnt.n <- function(x,n) sum(x>=n,na.rm=T)
draw.boxplot <- function(v1) {
 boxplot(as.formula(paste(v1, “~ cluster”)),data=df,main=v1,ylab=v1,xlab=”cluster”)
 m1 <- tapply(df[[v1]],df$cluster,mean,na.rm=T)
 cls.no<-length(unique(df$cluster))
 y <- matrix(m1,nrow=cls.no,ncol=2)
 x <- outer(1:cls.no,c(-0.4,0.4),”+”)
 for (i in 1:cls.no) lines(x[i,],y[i,],lty=2)
}

# read raw data again
df<-read.csv(HH_survey_data)

printSummaryOld<-function(v) {
 cat(“Dataset,\””, HH_survey_data, “\”\n”)
 cat(“Variable,”, v, “\n”)
 all<-as.data.frame(describeBy(df[[v]], group=rep(“all”,nrow(df)))[1])
 for (clust in describeBy(df[[v]], df$cluster)) { all<- rbind(all, unlist(clust))}

 names(all)<-c(“var”,”n”,”mean”,”sd”,”median”,”trimmed”,”mad”,”min”,”max”,”range”,”skew”,”kurtosis”)
 row.names(all)<-c(“all”, levels(df$cluster))
 cat(write.csv(all))
}

printSummary<-function(v) {
 cat(“Dataset,\””, HH_survey_data, “\”\n”)
 cat(“Variable,”, v, “\n”)
 all<-as.data.frame(describeBy(df[[v]] ))
 all$sum<-sum(df[[v]])
 c<-NULL
 for (clust in describeBy(df[[v]], df$cluster)) {
  c<- rbind(c, unlist(clust))
 }
 c<-cbind(c,sum=as.vector(by(df[[v]], df$cluster,sum)))
 out<-rbind(all, c)
 row.names(out)<-c(“all”, as.character(unique(df$cluster)))
 cat(write.csv(out))
}

for (v1 in names(df)) {
 if (is.numeric(df[[v1]])) {
  png(paste(out.dir,”/”,v1,”.png”,sep=””))
  try(draw.boxplot(v1), silent=T)
  dev.off()
  sink(paste(out.dir,”/”,v1,”.csv”,sep=””))
  try(printSummary(v1), silent=T)
  sink()
 }
}
```

4. R CODE FOR GENERATING HEATMAPS

The R following code (below this sentence) can be copied and pasted directly into the R console.

```
# Instructions
# 1. Before you can run this code, you need to create a datafile with the variables
# you want included in your heatmap (saved as comma delimited file '.csv')
# In this example we use the variables from the heatmaps in this book
# 2. Run R, change 'directory' to your work area (i.e. computer folder where the data is)
# 3. Edit the name of your heatmap .csv file (the spreadsheet with your data) in the code
# 5. You will need to download the 'ggplot2', 'plyr', 'scales' and 'reshape2' R packages to use this code
# (see Quick R for information on downloading packages)

# Generating coloured heatmaps for household clusters
# Load packages needed
library(ggplot2)
library(plyr)
library(scales)
library(reshape2)

# Set Working Directory (location of your datafile) – change example filepath in "" below
setwd("C:/My Computer/SIMLESA Typologies/ country chapters/Kenya/Analysis/Cluster results")

# remove previously loaded data from your R workspace
rm(list=ls())

# The name of your .csv heatmap data file should be inserted below:
# Load data
df0 <- read.csv("YOUR HEATMAP FILE NAME HERE.csv",as.is=T)

# calculate order of typologies by total income (TotalInc) for heatmap
# highest to lowest income
attach(df0)
heatmaporder<-df0[order(-TotalInc),]

detach(df0)

# read order for graph
heatmaporder$Farm.type

####### Rescale data for heatmap figures #######
df1 <- melt(df0)
df2 <- ddply(df1, .(variable), transform,scale = rescale(value))
```

```
# Subset only variables for heatmap
# (if you have more variables in your heatmap file than you want in your figure)
dfs<-subset(df2, variable==”Farm.type”|variable==”Land”|variable==”TLU_all”
|variable==”CE”|variable==”Education”|variable==”wlkminse”|variable==”h.female.
perc”|variable==”cropsales”|variable==”off.farm.inc”|variable==”other.income”|variable==”TotalInc”)
# need to make the variables name above are correctly written as they appear in your datafile

# Reorder clusters for final figure. This can be according to the order in “heatmaporder” values of
# “Farm.type” - SEE ABOVE) or simply however you want your clusters ordered.
# Change cluster names in the quotation marks below
dfs$Farm.type <- factor(dfs$Farm.type,ordered=T,levels= rev(c(“CLUSTER 1”,”CLUSTER 2”,”CLUSTER
3”,”CLUSTER 4”,”CLUSTER 5”,”CLUSTER 6”)))

# Create the graph (red colour)
p <- ggplot(dfs, aes(variable,Farm.type)) + geom_tile(aes(fill = scale),
                    colour = “white”)+labs(y=”Household Cluster”, x=””) + scale_fill_gradient(low =
“white”,
                                        high = “darkred”)+theme_bw()+theme(axis.text.x = element_
text(angle = -45, hjust = 0), axis.text=element_text(size=16), panel.grid.major = element_blank(), panel.
grid.minor = element_blank(), panel.background = element_blank(), axis.line = element_line(colour =
“black”), axis.title=element_text(size=18,face=”bold”), legend.title=element_text(size=16,face=”bold”),
legend.text=element_text(size=16), strip.text=element_text(size=20, face=”bold”), plot.title=element_
text(size=18, face=”bold”))+labs(colour=”Relative value of Variable (0-1)”)

# Create the graph (green colour)
p2 <- ggplot(dfs, aes(variable,Farm.type)) + geom_tile(aes(fill = scale),
                    colour = “white”)+labs(y=”Household Cluster”, x=””) + scale_fill_gradient(low =
“white”,
                                        high = “darkgreen”)+theme_bw()+theme(axis.text.x = element_
text(angle = -45, hjust = 0), axis.text=element_text(size=16), panel.grid.major = element_blank(), panel.
grid.minor = element_blank(), panel.background = element_blank(), axis.line = element_line(colour =
“black”), axis.title=element_text(size=18,face=”bold”), legend.title=element_text(size=16,face=”bold”),
legend.text=element_text(size=16), strip.text=element_text(size=20, face=”bold”), plot.title=element_
text(size=18, face=”bold”))+labs(colour=”Relative value of Variable (0-1)”)

# Create the graph (blue colour)
p3 <- ggplot(dfs, aes(variable,Farm.type)) + geom_tile(aes(fill = scale),
                    colour = “white”)+labs(y=”Household Cluster”, x=””) + scale_fill_gradient(low =
“white”,
                                        high = “darkblue”)+theme_bw()+theme(axis.text.x = element_
text(angle = -45, hjust = 0), axis.text=element_text(size=16), panel.grid.major = element_blank(), panel.
grid.minor = element_blank(), panel.background = element_blank(), axis.line = element_line(colour =
“black”), axis.title=element_text(size=18,face=”bold”), legend.title=element_text(size=16,face=”bold”),
legend.text=element_text(size=16), strip.text=element_text(size=20, face=”bold”), plot.title=element_
text(size=18, face=”bold”))+labs(colour=”Relative value of Variable (0-1)”)
```

```
# Save the graphs to file (may need to adjust height or width dimensions depending on your data)
ggsave(p, filename=”Kenya Heatmap - red.png”, width=350, height=250, units=”mm”, dpi=350)
ggsave(p2, filename=”Kenya Heatmap - green.png”, width=350, height=250, units=”mm”, dpi=350)
ggsave(p3, filename=”Kenya Heatmap - blue.png”, width=350, height=250, units=”mm”, dpi=350)
# Identify whether clusters are either low income or high Income,
# (based on being in the top half of the observed income range)
# (in other words, by scaled income above 0.5) – remember to rename “TotalInc”
# to your income variable
inor<-subset(dfs, variable==”TotalInc”)
inor$incrank<-ifelse(inor$scale > 0.5,c(“High Income”), c(“Low Income”))
View(inor)
##################################################
```