**Manual**

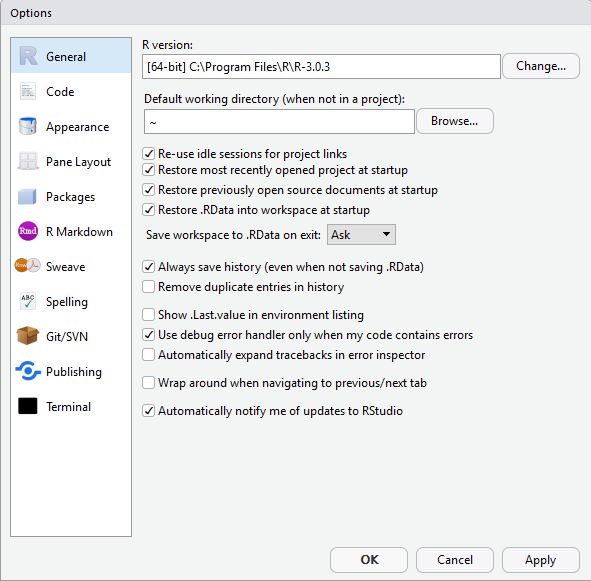
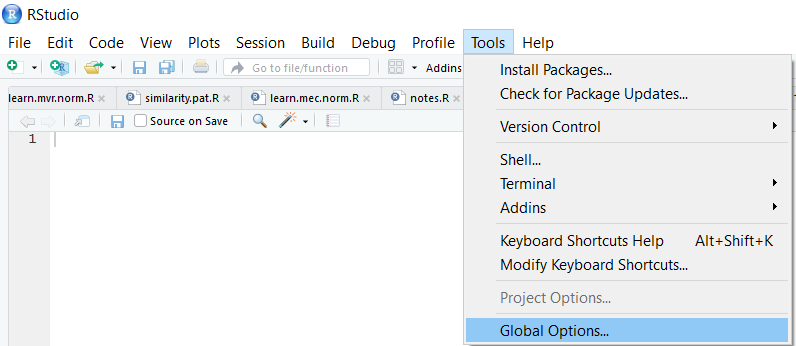
This manual tells you how to use the functions for learning LWF chain graphs (and as a special case, Bayesian networks) via decomposition approach and also the PC-like algorithm.

**RUNNING THE R CODE**:

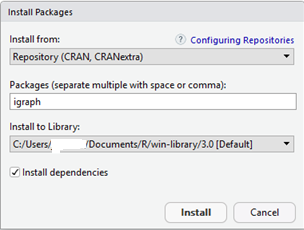
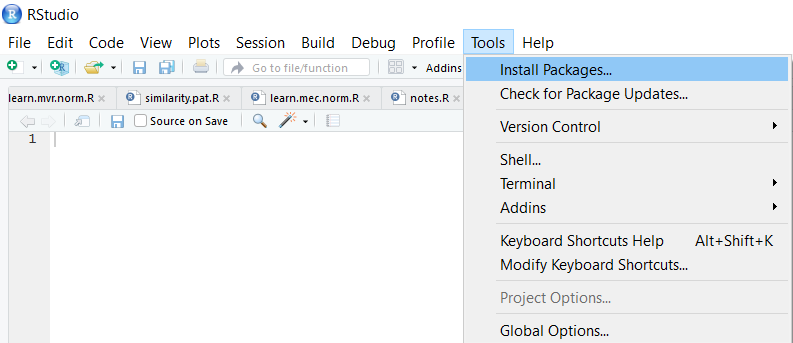
1. Install [R-3.0.3 for Windows (32/64 bit)](https://cran.r-project.org/bin/windows/base/old/3.0.3/).

**NOTICE**: Since LCD R package is not compatible with R versions greater than R-3.0.3, you need to install this version of R. In addition, since R versions can be installed side-by-side on a system, do not worry about the installing this version of R on your machine.

1. Install [RStudio 1.1.463 - Windows Vista/7/8/10](https://www.rstudio.com/products/rstudio/download/).
2. Run the RStudio software and make sure that you are using the right version of R i.e., R-3.0.3:



1. Install the following R packages: igraph, ggm, and lcd. Make sure that the Install Dependencies option is active (tick the box).



1. Install the pcalg R package and its dependencies. Also, copy and paste the following lines in your RStudio Console panel, respectively and press Enter:

source("https://bioconductor.org/biocLite.R")

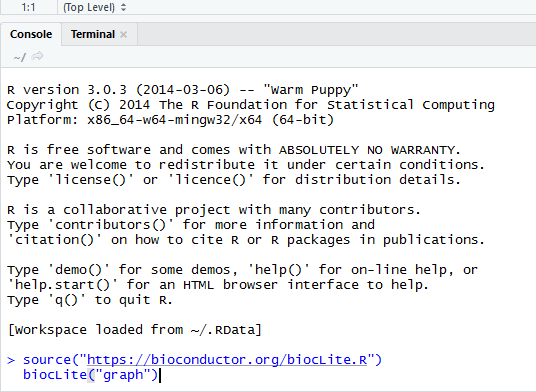
biocLite("graph")

source("https://bioconductor.org/biocLite.R")

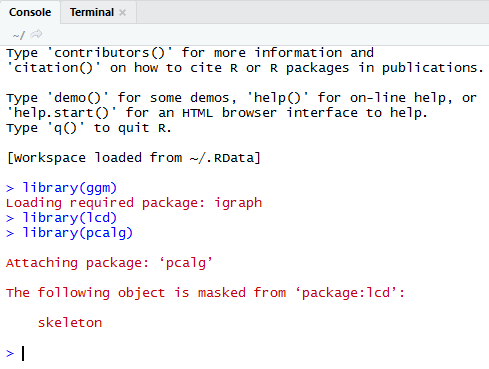
biocLite("RBGL")

source("https://bioconductor.org/biocLite.R")

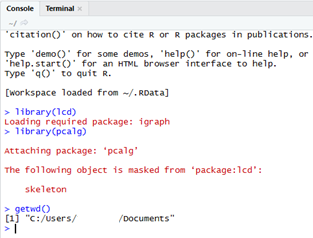
biocLite("Rgraphviz")



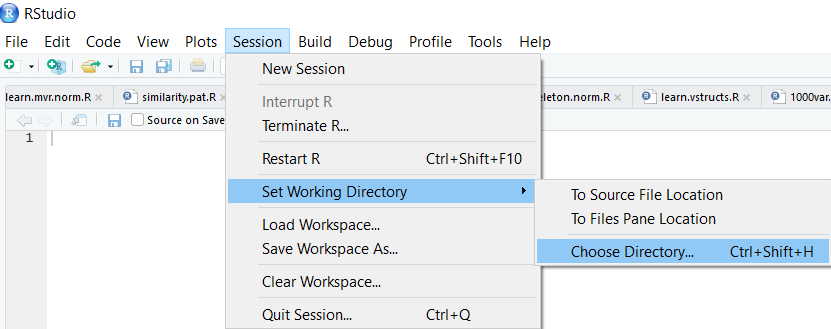
1. Load the following libraries: ggm, lcd and pcalg



1. R is always pointed at a directory on your computer. You can find out which directory by running the getwd (get working directory) function; this function has no arguments.



To change your working directory, use the following instruction and specify the path to the desired folder.



1. Download the R and csv files, and put them in the working directory.

**Examples**

#load the R code source

source("PClike.R")

#plot “toy.graph” from the R package lcd

draw(toy.graph)

#check whether “toy.graph” is a chain graph or not

is.chaingraph(toy.graph)

#Generate a desired number of normal random samples from the given CG

set.seed(100)

p.value <- .01

n <- 3000

tgdata <- rnorm.cg(n, toy.graph, get.normal.dist(toy.graph))

#Learn largest chain graph structure via PC-like algorithm

s<-learn.lwf.norm(tgdata,p.value = 0.01,method = "stable",LCG = TRUE)

#print the result in the console panel of the RStudio

s

#plot the learned LCG

draw(s$matrix)

#compare the learned LCG to the true pattern

comp.pat(pattern(toy.graph), s$matrix)

#Learn largest chain graph structure via LCD algorithm

tgug <- naive.getug.norm(tgdata, p.value)

tg.jtree <- ug.to.jtree(tgug)

tg.pat <- learn.mec.norm(tg.jtree, cov(tgdata), n, p.value, "CG")

learntLCG<-studeny\_rules(tg.pat)

#print the result in the console panel of the RStudio

learntLCG

#plot the learned LCG

draw(learntLCG$matrix)

#compare the learned LCG to the true pattern

comp.pat(pattern(toy.graph), learntLCG$matrix)

#double click on the “studeny.R” and copy & paste it in the console panel

#of the RStudio (or copy & paste the following lines):

studeny <- matrix(c(0,1,0,1,0,

0,0,1,1,0,

0,0,0,1,0,

0,0,0,0,1,

0,0,0,0,0),

nrow = 5, byrow = TRUE)

rownames(studeny) <- colnames(studeny) <- c("a","b","c","d","e")

#plot the “studeny” graph that corresponds to the DAG in (Studeny, 1997).

draw(studeny)

#First, put the “studeny.R” file in your workspace. This file contains 10000 random samples of the DAG in (Studeny, 1997). Then read the file:

cg.data<-read.csv("studeny.csv")

# Learn pattern of the “studeny” chain graph structure via LCD algorithm

cgug <- naive.getug.norm(cg.data, p.value=0.01)

cg.jtree <- ug.to.jtree(cgug)

cg.pat <- learn.mec.norm(cg.jtree, cov(cg.data), 10000, p.value=0.01, "CG")

#check whether the learned pattern of the “studeny” chain graph is a chain graph or not

is.chaingraph(cg.pat)

# plot the learned pattern of the “studeny” chain graph

draw(cg.pat)

# Learn the LCG of the “studeny” chain graph structure via LCD algorithm

learntLCG<-studeny\_rules(cg.pat)

#print the result in the console panel of the RStudio

learntLCG

#plot the learned LCG

draw(learntLCG$matrix)

#compare the learned LCG to the true pattern

comp.pat(pattern(studeny), learntLCG$matrix)

#Learn largest chain graph structure via PC-like algorithm

s<-learn.lwf.norm(cg.data,p.value = 0.01,method = "stable",LCG = TRUE)

#print the result in the console panel of the RStudio

s

#plot the learned LCG

draw(s$matrix)

#compare the learned LCG to the true pattern

comp.pat(pattern(studeny), s$matrix)