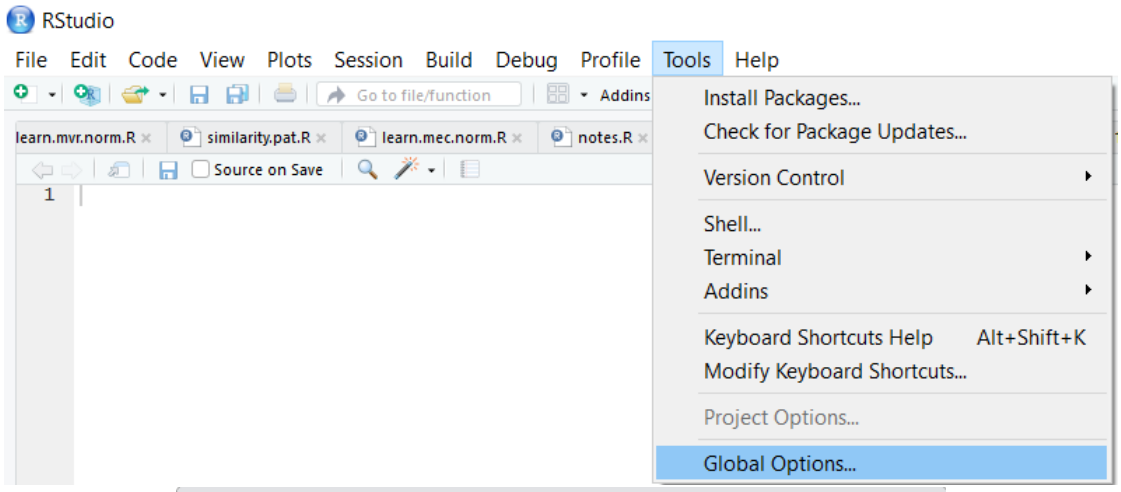
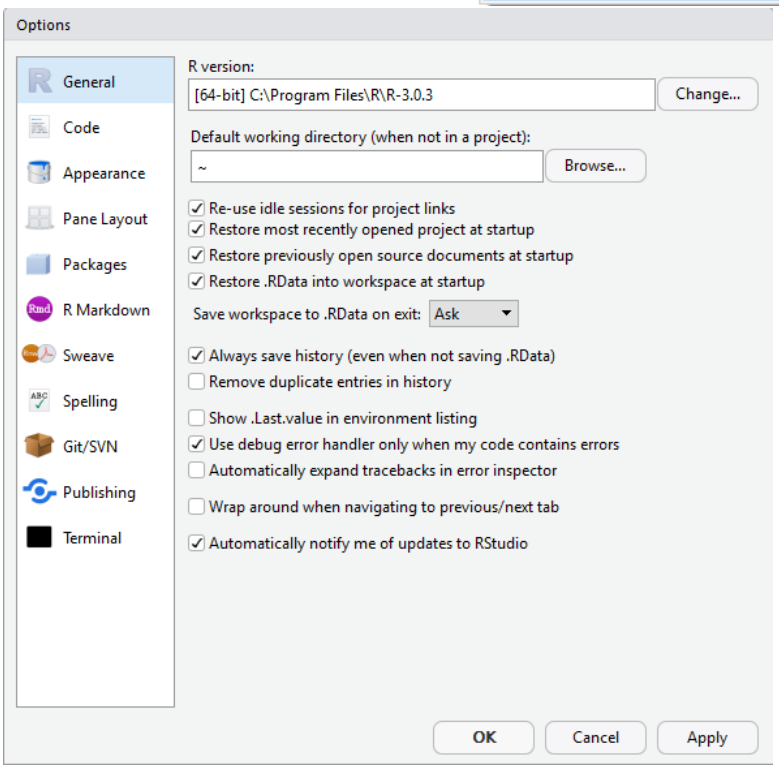
Manual

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

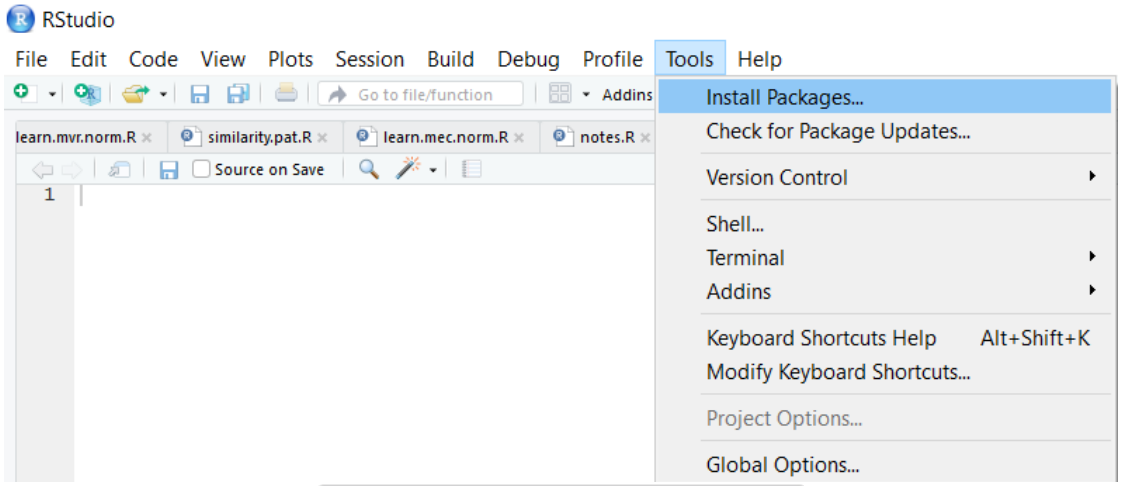
RUNNING THE R CODE:

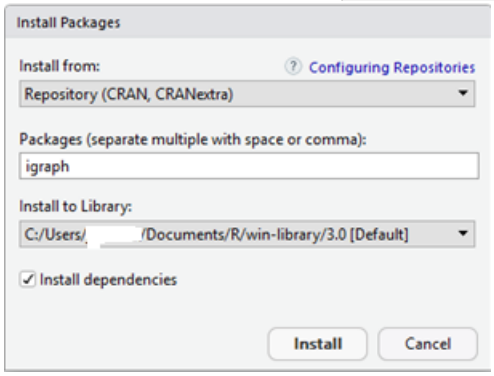
1. Install R-3.0.3 for Windows (32/64 bit).
   NOTICE: Since LCD R package is not compatible with R versions greater than R-3.0.3, you need to install this version of R. In addition, since R versions can be installed side-by-side on a system, do not worry about the installing this version of R on your machine.
2. Install RStudio 1.1.463 - Windows Vista/7/8/10.
3. Run the RStudio software and make sure that you are using the right version of R i.e., R-3.0.3:





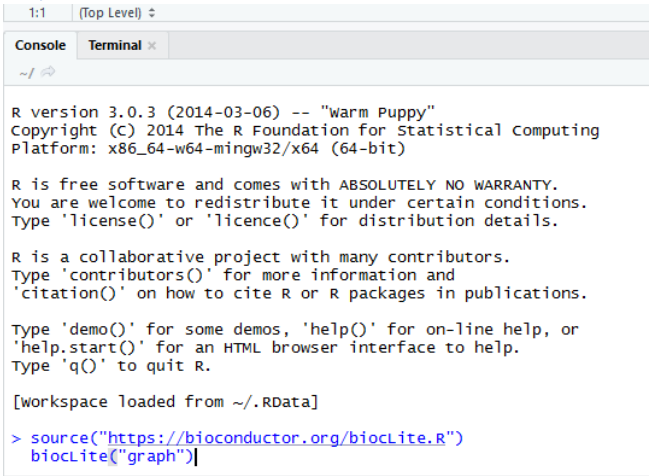
4. Install the following R packages: igraph, ggm, and lcd. Make sure that the Install Dependencies option is active (tick the box).





5. 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")
```



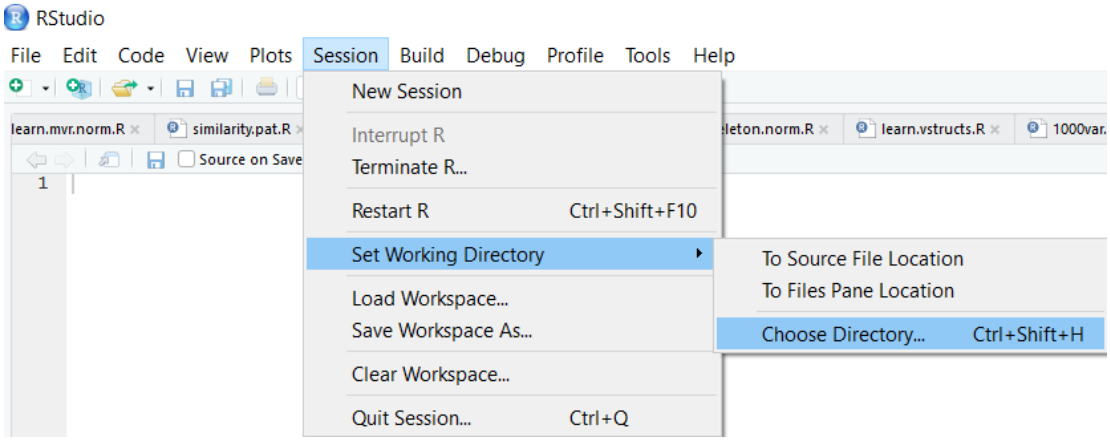
6. Load the following libraries: ggm, lcd and pcalg

```
Console  Terminal
~/
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

> library(ggm)
Loading required package: igraph
> library(lcd)
> library(pcalg)

Attaching package: 'pcalg'

The following object is masked from 'package:lcd':

    skeleton

>
```

7. 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.

```
Console  Terminal
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

> library(lcd)
Loading required package: igraph
> library(pcalg)

Attaching package: 'pcalg'

The following object is masked from 'package:lcd':

    skeleton

> getwd()
[1] "C:/Users/        /Documents"
>
```

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



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

Examples

1) Gaussian Case

```
#load the R code source
```

```
source("AMPCGs2019.R")

# copy & paste the following lines in the console panel
#of the RStudio
dag <- matrix(c(  0, 1, 1, 0, 0, 0,
                  0, 0, 0, 1, 0, 0,
                  0, 0, 0, 0, 1, 0,
                  0, 0, 0, 0, 1, 0,
                  0, 0, 0, 0, 0, 1,
                  0, 0, 0, 0, 0, 0),
                6,6, byrow = TRUE)
N <- c("a","b","c","d","e","f")
dimnames(dag) <- list(N, N)

#plot “dag” from the R package lcd
draw(dag)

#check whether “dag” is a chain graph
is.chaingraph(dag)

#First, put the “DAG.csv” file in your workspace. This file contains 3000 ran
dom samples of the DAG mentioned above. Then read the file:
cg.data<-read.csv("DAG.csv")

#Learn the chain graph structure via the LCD-like algorithm
ampcg<-learn.original.amp.normLCD(cg.data,p.value=0.05)

#print the result in the console panel of RStudio
ampcg

#plot the learned CG
draw(ampcg)

#compare the learned CG to the true CG
comp.cgs(dag,ampcg)

#Learn the largest deflagged graph (LDCG)
ldcg<-Largest_DeflaggedAMPCG(ampcg)

#print the result in the console panel of RStudio
ldcg

#plot the learned LDCG
draw(ldcg)

#compare the learned LDCG to the true LDCG (dag)
comp.cgs(dag,ldcg)
```

```
#learn the chain graph structure via PC-like algorithm
ampcg<-learn.amp.normPC(cg.data,p.value=0.05,method ="stable")

#print the result in the console panel of RStudio
ampcg

#plot the learned CG
draw(ampcg)
```

```
#compare the learned CG to the true CG
comp.cgs(dag,ampcg)

#learn the largest deflagged graph (LDCG)
ldcg<-Largest_DeflaggedAMPCG(ampcg)

#print the result in the console panel of RStudio
ldcg

#plot the learned LDCG
draw(ldcg)

#compare the learned LDCG to the true LDCG (dag)
comp.cgs(dag,ldcg)
```

---

2) Discrete Case

```
#First, put the “asia.csv” and “adj_asia.csv” file in your workspace.
This “asia.csv” file contains 5000 random samples of the ASIA network.
The “adj_asia.csv” file contains the adjacency matrix of the ASIA netw
ork.
#Read the files:
ds<-read.csv("asia.csv")

adj_asia<-read.csv("adj_asia.csv")
ref<-as(adj_asia,"matrix")
rownames(ref)<-colnames(ref)

#plot the ASIA net
draw(ref)

#learn using the stable PC-like algorithm:
cg<-learn.amp.multinomPC(ds,p.value=0.05,method = "stable")

#plot the learned net
draw(cg)

#compare the learned net to the ASIA net
comp.cgs(ref,cg)

#learn using the LCD-like algorithm:
cg<-learn.amp.multinomLCD(ds,p.value = 0.05)

#plot the learned net
draw(cg)

#compare the learned net to the ASIA net
comp.cgs(ref,cg)
```