Learning Genetic and Environmental Graphical Models using the R package FamilyBasedPGMs

Adèle Helena Ribeiro

2019-03-29

First, load the package FamilyBasedPGMs:

```
library(FamilyBasedPGMs)
```

Preparing the Dataset

Let's load the dataset containing the simulated data according to scenario 3:

```
data(scen1)
```

This dataset contains 100 replicates of phenotypic data for 900 individuals (30 families, each with 30 individuals).

The pedigrees of the families and the number of individuals in each family are in the following objects:

```
pedigrees <- scen1$pedigrees
fam.nf <- scen1$fam.nf
```

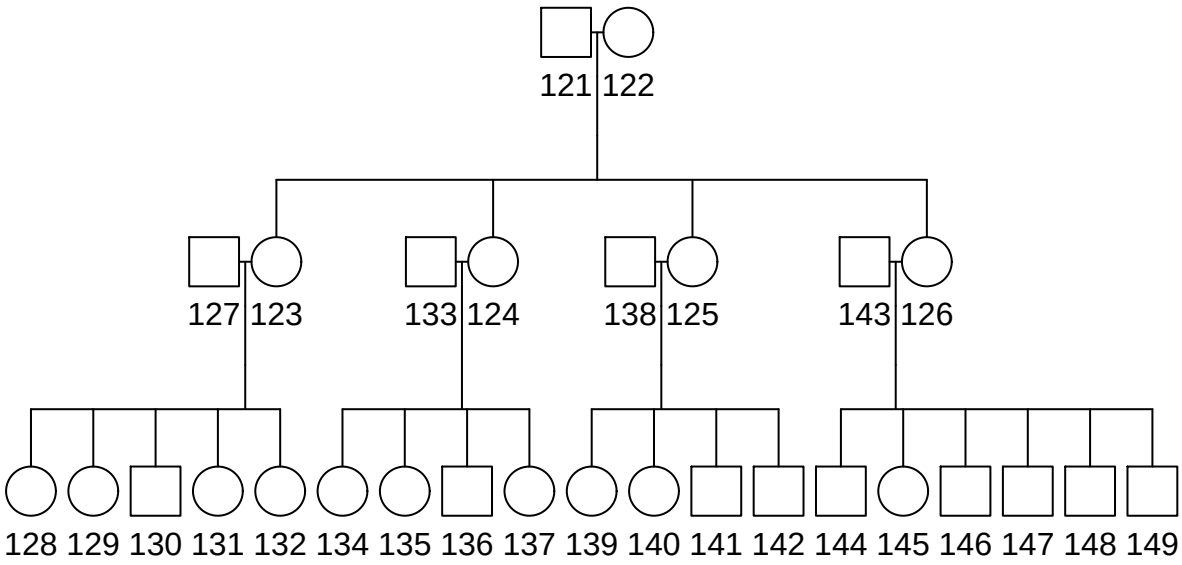
The `pedigrees` object is a data.frame with the columns "famid", "id", "momid", "dadid", and "sex". The first entries for the fifth family are:

Table 1: Pedigrees

	famid	id	momid	dadid	sex
121	5	121			1
122	5	122			2
123	5	123	122	121	2
124	5	124	122	121	2
125	5	125	122	121	2
126	5	126	122	121	2

To plot the pedigree chart of the fifth simulated family, you can run the following code:

```
plotFamilyPedigree(pedigrees, famid=5)
```




```
#> Did not plot the following people: 150
```

In this example, let's use the first replicate of the phenotypic data:

```
phen.df <- scen1$phen.df[[1]]
```

The first rows of `phen.df` are shown in the following:

Table 2: Phenotypes Dataset

	X	Y	Z
1	-1.686	-1.452	-1.101
2	0.919	0.324	0.788
3	-1.928	-0.299	-1.153
4	-0.957	-0.308	-0.085
5	-1.798	0.546	-0.101
6	1.040	-0.148	-0.310

Also, since no covariates was used in this simulation, let's define the covariate dataset as NULL:

```
covs.df <- NULL
```

Learning Total, Genetic, and Environmental Undirected PGMs

```
scenario = 1

# Total number of individuals
N <- sum(fam.nf)

fasterExample = TRUE

# Data was simulated for all individuals, so all individuals were "sampled".
sampled <- rep(1, N)
if (fasterExample) {
  # If you prefer to run a faster example, you can try with a smaller part of the dataset
```

```
  # However, it may compromise the accuracy of the recovered PGMs.
  set.seed(12345)
  sampled <- sample(c(TRUE, FALSE), 900, replace=TRUE)
  phen.df <- phen.df[sampled,]
}

fileID <- paste0("scen", scenario)
dirToSave <- paste0("./objects-UDG-", fileID, "/")
dir.create(dirToSave, showWarnings=FALSE)

alpha = 0.05

udgs.out <- learnFamilyBasedUDGs(phen.df, covs.df, pedigrees, sampled,
                                 fileID, dirToSave, alpha, correction=NULL,
                                 max_cores=NULL, minK=10, maxFC = 0.05,
                                 orthogonal=TRUE, useGPU=FALSE, debug=TRUE)
```

Now, we can check the learned undirected *total* PGM.

Its adjacency matrix is:

```
udgs.out$adjM$t
#>   X Y Z
#> X 0 1 1
#> Y 1 0 1
#> Z 1 1 0
```

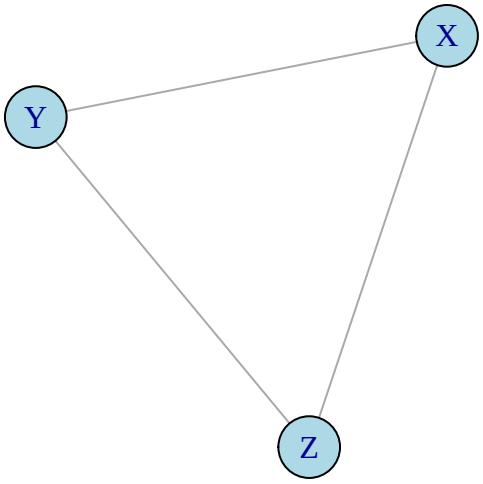
The estimates and p-values of the partial correlations are:

```
udgs.out$pCor$pCor_t
#> $estimates
#>            X          Y         Z
#> X         NA -0.4835768 0.6912148
#> Y -0.4835768         NA 0.7615219
#> Z  0.6912148  0.7615219        NA
#>
#> $pvalues
#>               X            Y            Z
#> X            NA 1.189697e-25 4.282081e-60
#> Y 1.189697e-25           NA 1.315617e-79
#> Z 4.282081e-60 1.315617e-79           NA
```

Plotting the learned PGM using its representation as an `igraph` object:

```
# igraph object
udgs.out$udg$t
#> IGRAPH 9468813 UN-- 3 3 --
#> + attr: name (v/c)
#> + edges from 9468813 (vertex names):
#> [1] X--Y X--Z Y--Z

plot(udgs.out$udg$t, vertex.size=30,
     vertex.color="lightblue")
```



Let's now check the learned undirected *genetic* PGM.

Its adjacency matrix is:

```
udgs.out$adjM$g
#>   X Y Z
#> X 0 1 1
#> Y 1 0 1
#> Z 1 1 0
```

The estimates, p-values, and effective sizes of the partial correlations are:

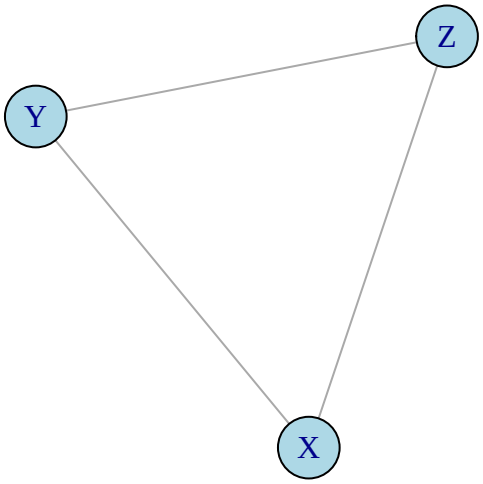
```
udgs.out$pCor$pCor_g
#> $estimates
#>            X          Y         Z
#> X         NA -0.5504830 0.7320944
#> Y -0.5504830         NA 0.7429468
#> Z  0.7320944  0.7429468        NA
#>
#> $pvalues
#>               X            Y            Z
#> X            NA 1.153901e-06 1.509676e-19
#> Y 1.153901e-06           NA 4.718409e-12
#> Z 1.509676e-19 4.718409e-12           NA
#>
#> $k
#>     X  Y   Z
#> X  NA 68 109
#> Y  68 NA  62
#> Z 109 62  NA
```

Plotting the learned PGM using its representation as an `igraph` object:

```
# igraph object
udgs.out$udg$g
#> IGRAPH 6bbe0a4 UN-- 3 3 --
#> + attr: name (v/c)
#> + edges from 6bbe0a4 (vertex names):
#> [1] X--Y X--Z Y--Z

plot(udgs.out$udg$g, vertex.size=30,
```

```
     vertex.color="lightblue")
```



Finally, let's check the learned undirected *environmental* PGM.

Its adjacency matrix is:

```
udgs.out$adjM$e
#>   X Y Z
#> X 0 1 1
#> Y 1 0 1
#> Z 1 1 0
```

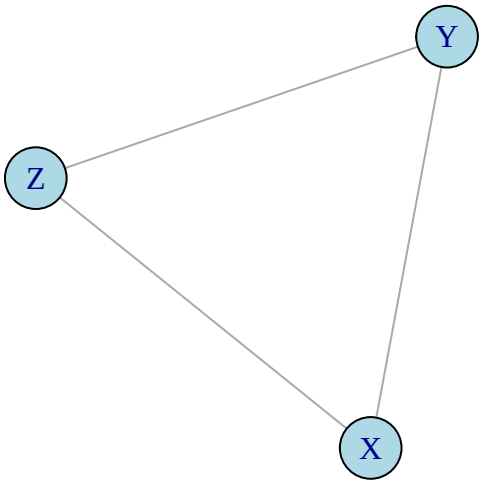
The estimates, p-values, and effective sizes of the partial correlations are:

```
udgs.out$pCor$pCor_e
#> $estimates
#>            X          Y         Z
#> X         NA -0.4759565 0.6333600
#> Y -0.4759565         NA 0.7901565
#> Z  0.6333600  0.7901565        NA
#>
#> $pvalues
#>              X            Y            Z
#> X           NA 1.442939e-03 1.973361e-05
#> Y 1.442939e-03           NA 1.095614e-18
#> Z 1.973361e-05 1.095614e-18           NA
#>
#> $k
#>    X  Y  Z
#> X NA 42 38
#> Y 42 NA 82
#> Z 38 82 NA
```

Plotting the learned PGM using its representation as an `igraph` object:

```
# igraph object
udgs.out$udg$e
#> IGRAPH 1aeada3 UN-- 3 3 --
#> + attr: name (v/c)
#> + edges from 1aeada3 (vertex names):
#> [1] X--Y X--Z Y--Z
```

```r
plot(udgs.out$udg$e, vertex.size=30,
     vertex.color="lightblue")
```



Learning Total, Genetic, and Environmental Directed Acyclic PGMs

```r
fileID <- paste0("scen", scenario)
dirToSave <- paste0("./objects-PC-", fileID, "/")
dir.create(dirToSave, showWarnings=FALSE)

alpha = 0.05 # significance level

dags <- learnFamilyBasedDAGs(phen.df, covs.df, pedigrees, sampled,
   fileID, dirToSave, alpha, max_cores=NULL,
   minK=10, maxFC = 0.05, orthogonal=TRUE, maj.rule=TRUE,
   useGPU=FALSE, debug=TRUE, savePlots=FALSE)
```

Now, we can check the learned directed acyclic *total* PGM.

Its adjacency matrix is:

```r
adjM_t <- as(dags$t, "amat")
adjM_t
#> Adjacency Matrix 'amat' (3 x 3) of type 'cpdag':
#>     X_t Y_t Z_t
#> X_t   .   .   .
#> Y_t   .   .   .
#> Z_t   1   1   .
```

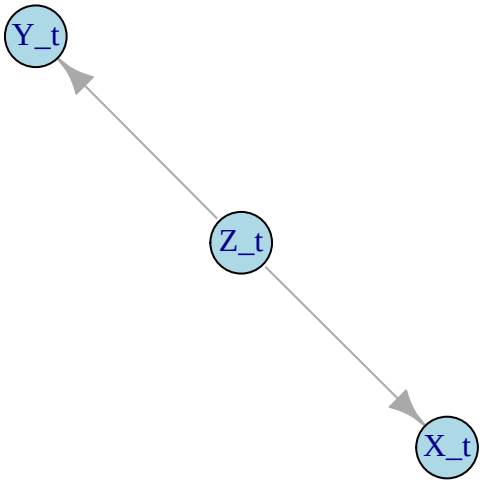
Inspecting and plotting the learned PGM using its representation as an `igraph` object:

```r
# igraph object
dags$t
#> Object of class 'pcAlgo', from Call:
#> pc(suffStat = suffStat, indepTest = familyBasedCITest, alpha = alpha,
#>     labels = paste0(colnames(phen.df), "_", type), skel.method = "stable",
#>     maj.rule = TRUE, solve.confl = TRUE, verbose = TRUE)
#> Number of undirected edges:  0
#> Number of directed edges:    2
```

```
#> Total number of edges:       2

pcalg::showEdgeList(dags$t)
#>
#> Edge List:
#>
#> Undirected Edges:
#>
#> Directed Edges:
#>   X_t  -->  Z_t
#>   Y_t  -->  Z_t

igraph::plot.igraph(igraph::graph.adjacency(adjM_t),
                    vertex.size=30, vertex.color="lightblue")
```



Let's now check the learned directed acyclic *genetic* PGM.

Its adjacency matrix is:

```
adjM_g <- as(dags$g, "amat")
adjM_g
#> Adjacency Matrix 'amat' (3 x 3) of type 'cpdag':
#>     X_g Y_g Z_g
#> X_g   .   .   .
#> Y_g   .   .   .
#> Z_g   1   1   .
```

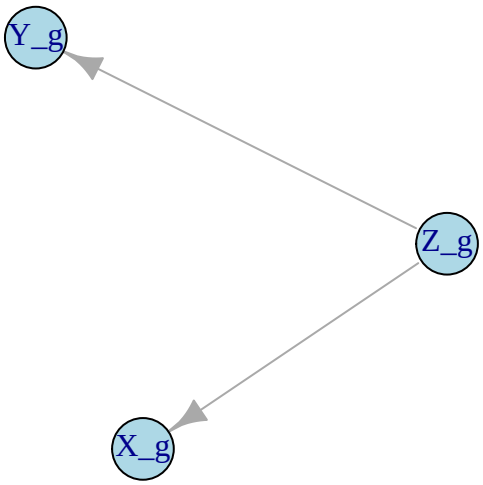
Inspecting and plotting the learned PGM using its representation as an `igraph` object:

```
# igraph object
dags$g
#> Object of class 'pcAlgo', from Call:
#> pc(suffStat = suffStat, indepTest = familyBasedCITest, alpha = alpha,
#>     labels = paste0(colnames(phen.df), "_", type), skel.method = "stable",
#>     maj.rule = TRUE, solve.confl = TRUE, verbose = TRUE)
#> Number of undirected edges:  0
#> Number of directed edges:    2
#> Total number of edges:       2

pcalg::showEdgeList(dags$g)
```

```
#>
#> Edge List:
#>
#> Undirected Edges:
#>
#> Directed Edges:
#>   X_g  -->  Z_g
#>   Y_g  -->  Z_g

igraph::plot.igraph(igraph::graph.adjacency(adjM_g),
                    vertex.size=30, vertex.color="lightblue")
```



Finally, let's check the learned directed acyclic *environmental* PGM.

Its adjacency matrix is:

```
adjM_e <- as(dags$e, "amat")
adjM_e
#> Adjacency Matrix 'amat' (3 x 3) of type 'cpdag':
#>     X_e Y_e Z_e
#> X_e   .   .   .
#> Y_e   .   .   .
#> Z_e   1   1   .
```

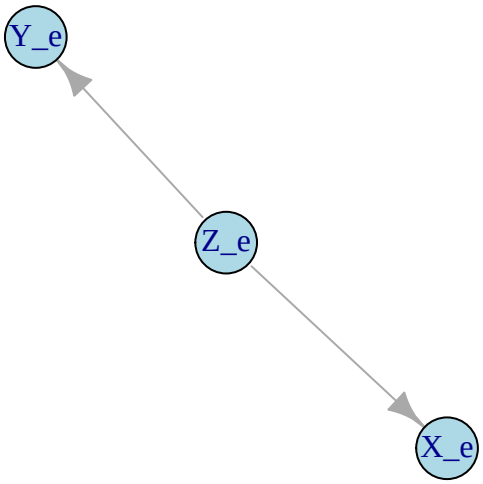
Inspecting and plotting the learned PGM using its representation as an `igraph` object:

```
# igraph object
dags$e
#> Object of class 'pcAlgo', from Call:
#> pc(suffStat = suffStat, indepTest = familyBasedCITest, alpha = alpha,
#>     labels = paste0(colnames(phen.df), "_", type), skel.method = "stable",
#>     maj.rule = TRUE, solve.confl = TRUE, verbose = TRUE)
#> Number of undirected edges:  0
#> Number of directed edges:    2
#> Total number of edges:       2

pcalg::showEdgeList(dags$e)
#>
#> Edge List:
#>
```

```
#> Undirected Edges:
#>
#> Directed Edges:
#>   X_e  -->  Z_e
#>   Y_e  -->  Z_e

igraph::plot.igraph(igraph::graph.adjacency(adjM_e),
                    vertex.size=30, vertex.color="lightblue")
```



The results of the partial correlation tests are saved at `paste0(dirToSave, fileID, "_preprPvalues.csv")`:

```
stargazer::stargazer(round(read.table(
  paste0(dirToSave, fileID, "_preprPvalues.csv"), header=TRUE, sep=";"),
  3), summary=FALSE, title = "Partial Correlation Results", header=FALSE,
  column.sep.width = "-5pt", label="PCpcor")
```

Table 3: Partial Correlation Results

	x	y	S	pcort	pcort_p.value	pcorg	pcorg_p.value	pcore	pcore_p.value	se	sg
1	1	2		0.094	0.055	0	0.998	0	0.999	18	22
2	1	3		0.571	0	0.627	0	0.461	0.004	38	82
3	2	3		0.676	0	0.602	0	0.737	0	68	62
4	2	1		0.094	0.055	0	0.998	0	0.999	18	22
5	3	1		0.571	0	0.627	0	0.461	0.004	38	82
6	3	2		0.676	0	0.602	0	0.737	0	68	62
7	1	2	3	-0.484	0	-0.550	0	-0.476	0.001	42	68
8	1	3	2	0.691	0	0.732	0	0.633	0	38	109
9	2	3	1	0.762	0	0.743	0	0.790	0	82	62
10	2	1	3	-0.484	0	-0.550	0	-0.476	0.001	42	68
11	3	1	2	0.691	0	0.732	0	0.633	0	38	109
12	3	2	1	0.762	0	0.743	0	0.790	0	82	62