Causal Inference for QTL Networks with R/qtlnet Package

Elias Chaibub Neto and Brian S. Yandell

June 8, 2012

This vignette briefly describes the R/qtlnet package. This contains the legacy R/qdg package, and thus has code for Chaibub Neto et al. (2008) and Chaibub Neto et al. (2010) papers. Not all routines are described here. Further, the package has code for parallel processing using Condor that is not yet documented adequately.

1 QTLNET routines

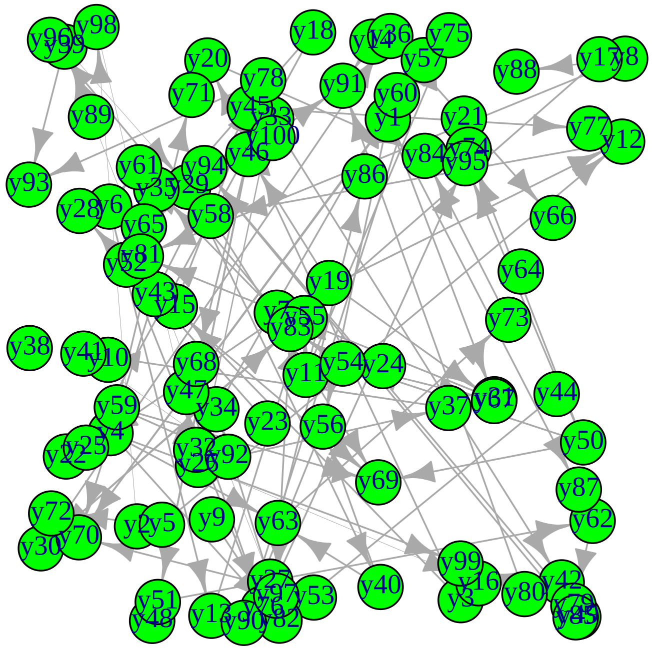
```
> library(qtlnet)
```

Acyclic example:

```
> example(acyclic)
```

```
acyclc> ## Not run:
acyclc> ##D ## This reproduces Figure 1 exactly.
acyclc> ##D set.seed(3456789)
acyclc> ##D
acyclc> ##D tmp <- options(warn=-1)
acyclc> ##D acyclic.DG <- randomDAG(n = 100, prob = 2 / 99)
acyclc> ##D
acyclc> ##D options(tmp)
acyclc> ##D
acyclc> ##D ## Simulate cross object using R/qtl routines.
acyclc> ##D n.ind <- 300
acyclc> ##D mymap <- sim.map(len=rep(100,20), n.mar=10, eq.spacing=FALSE, include.x=FALSE)
acyclc> ##D mycross <- sim.cross(map=mymap, n.ind=n.ind, type="f2")
acyclc> ##D summary(mycross)
acyclc> ##D mycross <- sim.geno(mycross,n.draws=1)
acyclc> ##D
acyclc> ##D
acyclc> ##D ## Produce 100 QTL at three markers apiece.
acyclc> ##D acyclic.qtl <- generate.qtl.markers(cross=mycross,n.phe=100)
acyclc> ##D
acyclc> ##D ## Generate data from directed graph.
acyclc> ##D bp <- runif(100,0.5,1)
acyclc> ##D stdev <- runif(100,0.1,0.5)
acyclc> ##D bq <- matrix(0,100,3)
acyclc> ##D bq[,1] <- runif(100,0.2,0.4)
acyclc> ##D bq[,2] <- bq[,1]+0.1
acyclc> ##D bq[,3] <- bq[,2]+0.1
acyclc> ##D ## Generate phenotypes.
acyclc> ##D acyclic.data <- generate.qtl.pheno("acyclic", cross = mycross,
acyclc> ##D   bp = bp, bq = bq, stdev = stdev, allqtl = acyclic.qtl$allqtl)
```

```
acyclc> ##D
acyclc> ##D acyclic.qdg <- qdg(cross=acyclic.data,
acyclc> ##D                  phenotype.names=paste("y",1:100,sep=""),
acyclc> ##D                  marker.names=acyclic.qtl$markers,
acyclc> ##D                  QTL=acyclic.qtl$allqtl,
acyclc> ##D                  alpha=0.005,
acyclc> ##D                  n.qdg.random.starts=1,
acyclc> ##D                  skel.method="pcskel")
acyclc> ##D save(acyclic.DG, acyclic.qtl, acyclic.data, acyclic.qdg,
acyclc> ##D   file = "acyclic.RData", compress = TRUE)
acyclc> ## End(Not run)
acyclc>
acyclc> data(acyclic)

acyclc> dims <- dim(acyclic.data$pheno)

acyclc> SuffStat <- list(C = cor(acyclic.data$pheno), n = dims[1])

acyclc> pc <- skeleton(SuffStat, gaussCItest, p = dims[2], alpha = 0.005)

acyclc> summary(pc)

Object of class 'pcAlgo', from Call:
 skeleton(suffStat = SuffStat, indepTest = gaussCItest, p = dims[2],      alpha = 0.005)

Nmb. edgetests during skeleton estimation:
===========================================
Max. order of algorithm:  3
Number of edgetests from m = 0 up to m = 3 :  5426 1899 294 36

Graphical properties of skeleton:
=================================
Max. number of neighbours:  4 at node(s) 1 4 19 50 63 65 69 70 78
Avg. number of neighbours:  1.88

acyclc> summary(graph.qdg(acyclic.qdg))
Vertices: 259
Edges: 394
Directed: TRUE
No graph attributes.
Vertex attributes: name, label, color, fill.
Edge attributes: width.

acyclc> gr <- graph.qdg(acyclic.qdg, include.qtl = FALSE)

acyclc> plot(gr)
```

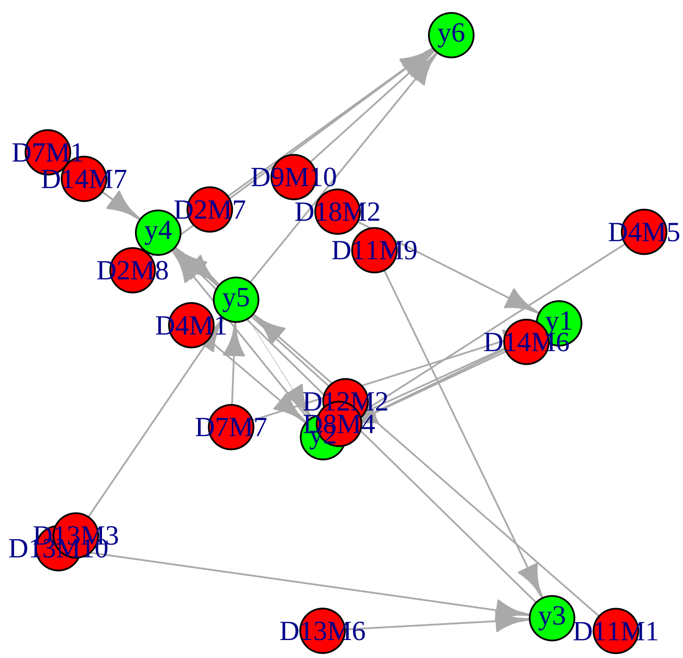


Cyclic A example:

```
> example(cyclica)

cyclic> ## Not run:
cyclic> ##D bp <- matrix(0, 6, 6)
cyclic> ##D bp[2,1] <- bp[4,2] <- bp[4,3] <- bp[5,4] <- bp[2,5] <- bp[6,5] <- 0.5
cyclic> ##D stdev <- rep(0.025, 6)
cyclic> ##D
cyclic> ##D ## Use R/qtl routines to simulate.
cyclic> ##D set.seed(3456789)
cyclic> ##D mymap <- sim.map(len = rep(100,20), n.mar = 10, eq.spacing = FALSE,
cyclic> ##D   include.x = FALSE)
cyclic> ##D mycross <- sim.cross(map = mymap, n.ind = 200, type = "f2")
cyclic> ##D mycross <- sim.geno(mycross, n.draws = 1)
cyclic> ##D
cyclic> ##D cyclica.qtl <- generate.qtl.markers(cross = mycross, n.phe = 6)
cyclic> ##D mygeno <- pull.geno(mycross)[, unlist(cyclica.qtl$markers)]
cyclic> ##D
cyclic> ##D cyclica.data <- generate.qtl.pheno("cyclica", cross = mycross, burnin = 2000,
cyclic> ##D   bq = c(0.2,0.3,0.4), bp = bp, stdev = stdev, geno = mygeno)
cyclic> ##D save(cyclica.qtl, cyclica.data, file = "cyclica.RData", compress = TRUE)
cyclic> ## End(Not run)
```

```
cyclic>
cyclic> data(cyclica)

cyclic> out <- qdg(cross=cyclica.data,
cyclic+                 phenotype.names=paste("y",1:6,sep=""),
cyclic+                 marker.names=cyclica.qtl$markers,
cyclic+                 QTL=cyclica.qtl$allqtl,
cyclic+                 alpha=0.005,
cyclic+                 n.qdg.random.starts=10,
cyclic+                 skel.method="pcskel")

cyclic> gr <- graph.qdg(out)

cyclic> gr
Vertices: 23
Edges: 24
Directed: TRUE
Graph attributes:
Vertex attributes:
       name  label color  fill
[0]      y1     y1 green green
[1]      y2     y2 green green
[2]      y3     y3 green green
[3]      y4     y4 green green
[4]      y5     y5 green green
[5]      y6     y6 green green
[6]   D11M1  D11M1   red   red
[7]   D11M9  D11M9   red   red
[8]   D12M2  D12M2   red   red
[9]  D13M10 D13M10   red   red
[10]  D13M3  D13M3   red   red
[11]  D13M6  D13M6   red   red
[12]  D14M6  D14M6   red   red
[13]  D14M7  D14M7   red   red
[14]  D18M2  D18M2   red   red
[15]   D2M7   D2M7   red   red
[16]   D2M8   D2M8   red   red
[17]   D4M1   D4M1   red   red
[18]   D4M5   D4M5   red   red
[19]   D7M1   D7M1   red   red
[20]   D7M7   D7M7   red   red
[21]   D8M4   D8M4   red   red
[22]  D9M10  D9M10   red   red
Edges and their attributes:
                              width
[0]  'y1'     -> 'y2'     0.9996391
[1]  'y2'     -> 'y4'     1.0000000
[2]  'y5'     -> 'y2'     0.0000000
[3]  'y3'     -> 'y4'     1.0000000
[4]  'y4'     -> 'y5'     1.0000000
[5]  'y5'     -> 'y6'     1.0000000
[6]  'D18M2'  -> 'y1'     1.0000000
[7]  'D8M4'   -> 'y1'     1.0000000
[8]  'D7M7'   -> 'y1'     1.0000000
```

```
[9]  'D4M5'   -> 'y2'     1.0000000
[10] 'D4M1'   -> 'y2'     1.0000000
[11] 'D14M6'  -> 'y2'     1.0000000
[12] 'D13M6'  -> 'y3'     1.0000000
[13] 'D11M9'  -> 'y3'     1.0000000
[14] 'D13M10' -> 'y3'     1.0000000
[15] 'D12M2'  -> 'y4'     1.0000000
[16] 'D7M1'   -> 'y4'     1.0000000
[17] 'D14M7'  -> 'y4'     1.0000000
[18] 'D7M7'   -> 'y5'     1.0000000
[19] 'D13M3'  -> 'y5'     1.0000000
[20] 'D11M1'  -> 'y5'     1.0000000
[21] 'D2M8'   -> 'y6'     1.0000000
[22] 'D2M7'   -> 'y6'     1.0000000
[23] 'D9M10'  -> 'y6'     1.0000000

cyclic> plot(gr)
```

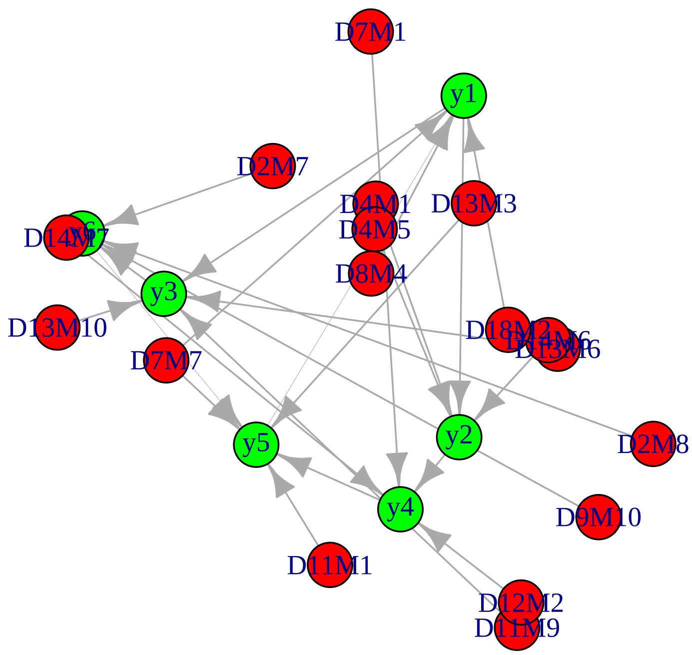


Cyclic B example:

```
> example(cyclicb)

cyclcb> ## Not run:
cyclcb> ##D bp <- matrix(0, 6, 6)
```

```
cyclcb> ##D bp[2,1] <- bp[1,5] <- bp[3,1] <- bp[4,2] <- bp[5,4] <- bp[5,6] <- bp[6,3] <- 0.5
cyclcb> ##D stdev <- rep(0.025, 6)
cyclcb> ##D
cyclcb> ##D ## Use R/qtl routines to simulate.
cyclcb> ##D set.seed(3456789)
cyclcb> ##D mymap <- sim.map(len = rep(100,20), n.mar = 10, eq.spacing = FALSE,
cyclcb> ##D   include.x = FALSE)
cyclcb> ##D mycross <- sim.cross(map = mymap, n.ind = 200, type = "f2")
cyclcb> ##D mycross <- sim.geno(mycross, n.draws = 1)
cyclcb> ##D
cyclcb> ##D cyclicb.qtl <- generate.qtl.markers(cross = mycross, n.phe = 6)
cyclcb> ##D mygeno <- pull.geno(mycross)[, unlist(cyclicb.qtl$markers)]
cyclcb> ##D
cyclcb> ##D cyclicb.data <- generate.qtl.pheno("cyclicb", cross = mycross, burnin = 2000,
cyclcb> ##D   bq = c(0.2,0.3,0.4), bp = bp, stdev = stdev, geno = mygeno)
cyclcb> ##D save(cyclicb.qtl, cyclicb.data, file = "cyclicb.RData", compress = TRUE)
cyclcb> ## End(Not run)
cyclcb>
cyclcb> data(cyclicb)

cyclcb> out <- qdg(cross=cyclicb.data,
cyclcb+                 phenotype.names=paste("y",1:6,sep=""),
cyclcb+                 marker.names=cyclicb.qtl$markers,
cyclcb+                 QTL=cyclicb.qtl$allqtl,
cyclcb+                 alpha=0.005,
cyclcb+                 n.qdg.random.starts=10,
cyclcb+                 skel.method="pcskel")

cyclcb> gr <- graph.qdg(out)

cyclcb> gr
Vertices: 23
Edges: 25
Directed: TRUE
Graph attributes:
Vertex attributes:
       name  label color  fill
[0]      y1     y1 green green
[1]      y2     y2 green green
[2]      y3     y3 green green
[3]      y4     y4 green green
[4]      y5     y5 green green
[5]      y6     y6 green green
[6]   D11M1  D11M1   red   red
[7]   D11M9  D11M9   red   red
[8]   D12M2  D12M2   red   red
[9]  D13M10 D13M10   red   red
[10]  D13M3  D13M3   red   red
[11]  D13M6  D13M6   red   red
[12]  D14M6  D14M6   red   red
[13]  D14M7  D14M7   red   red
[14]  D18M2  D18M2   red   red
[15]   D2M7   D2M7   red   red
[16]   D2M8   D2M8   red   red
```

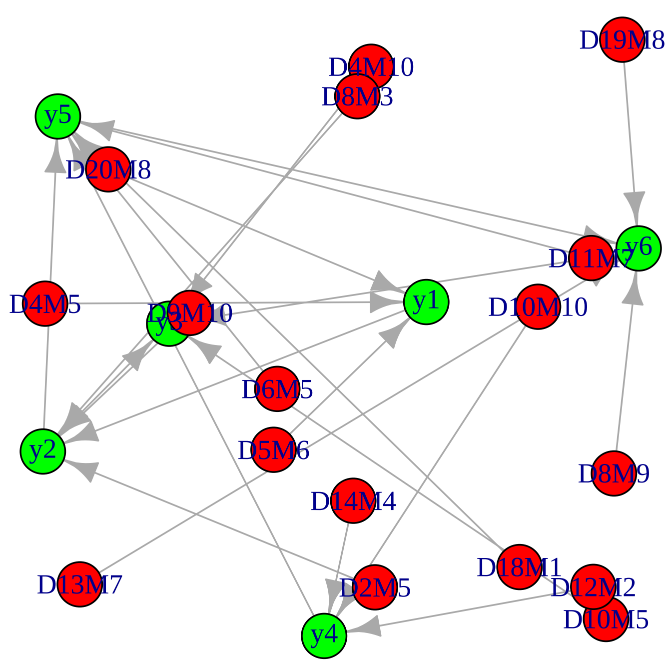
```
[17]   D4M1   D4M1   red   red
[18]   D4M5   D4M5   red   red
[19]   D7M1   D7M1   red   red
[20]   D7M7   D7M7   red   red
[21]   D8M4   D8M4   red   red
[22]  D9M10  D9M10   red   red
Edges and their attributes:
                           width
[0]  'y1'     -> 'y2'          1
[1]  'y1'     -> 'y3'          1
[2]  'y5'     -> 'y1'          0
[3]  'y2'     -> 'y4'          1
[4]  'y3'     -> 'y6'          1
[5]  'y4'     -> 'y5'          1
[6]  'y6'     -> 'y5'          0
[7]  'D18M2'  -> 'y1'          1
[8]  'D8M4'   -> 'y1'          1
[9]  'D7M7'   -> 'y1'          1
[10] 'D4M5'   -> 'y2'          1
[11] 'D4M1'   -> 'y2'          1
[12] 'D14M6'  -> 'y2'          1
[13] 'D13M6'  -> 'y3'          1
[14] 'D11M9'  -> 'y3'          1
[15] 'D13M10' -> 'y3'          1
[16] 'D12M2'  -> 'y4'          1
[17] 'D7M1'   -> 'y4'          1
[18] 'D14M7'  -> 'y4'          1
[19] 'D7M7'   -> 'y5'          1
[20] 'D13M3'  -> 'y5'          1
[21] 'D11M1'  -> 'y5'          1
[22] 'D2M8'   -> 'y6'          1
[23] 'D2M7'   -> 'y6'          1
[24] 'D9M10'  -> 'y6'          1

cyclcb> plot(gr)
```



Cyclic C example:

```
> example(cyclicc)

cyclcc> ## Not run:
cyclcc> ##D bp <- matrix(0, 6, 6)
cyclcc> ##D bp[2,5] <- 0.5
cyclcc> ##D bp[5,2] <- 0.8
cyclcc> ##D bp[2,1] <- bp[3,2] <- bp[5,4] <- bp[6,5] <- 0.5
cyclcc> ##D stdev <- rep(0.025, 6)
cyclcc> ##D
cyclcc> ##D ## Use R/qtl routines to simulate map and genotypes.
cyclcc> ##D set.seed(34567899)
cyclcc> ##D mymap <- sim.map(len = rep(100,20), n.mar = 10, eq.spacing = FALSE,
cyclcc> ##D   include.x = FALSE)
cyclcc> ##D mycross <- sim.cross(map = mymap, n.ind = 200, type = "f2")
cyclcc> ##D mycross <- sim.geno(mycross, n.draws = 1)
cyclcc> ##D
cyclcc> ##D ## Use R/qdg routines to produce QTL sample and generate phenotypes.
cyclcc> ##D cyclicc.qtl <- generate.qtl.markers(cross = mycross, n.phe = 6)
cyclcc> ##D mygeno <- pull.geno(mycross)[, unlist(cyclicc.qtl$markers)]
cyclcc> ##D
cyclcc> ##D cyclicc.data <- generate.qtl.pheno("cyclicc", cross = mycross, burnin = 2000,
```

```
cyclcc> ##D   bq = c(0.2,0.3,0.4), bp = bp, stdev = stdev, geno = mygeno)
cyclcc> ##D save(cyclicc.qtl, cyclicc.data, file = "cyclicc.RData", compress = TRUE)
cyclcc> ## End(Not run)
cyclcc>
cyclcc> data(cyclicc)

cyclcc> out <- qdg(cross=cyclicc.data,
cyclcc+                 phenotype.names=paste("y",1:6,sep=""),
cyclcc+                 marker.names=cyclicc.qtl$markers,
cyclcc+                 QTL=cyclicc.qtl$allqtl,
cyclcc+                 alpha=0.005,
cyclcc+                 n.qdg.random.starts=1,
cyclcc+                 skel.method="pcskel")

cyclcc> gr <- graph.qdg(out)

cyclcc> plot(gr)
```



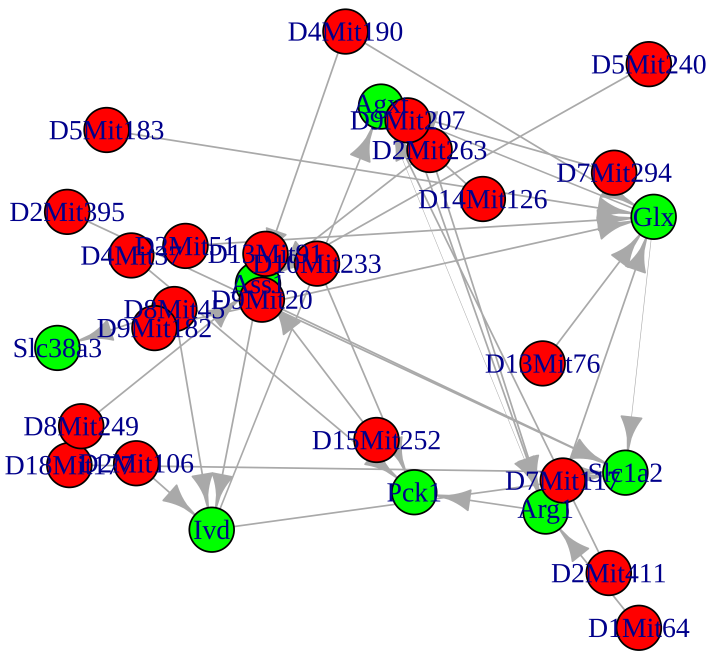
GLX network example (from Chaibub Neto et al. (2008)):

```
> example(glxnet)

glxnet> data(glxnet)
```

```
glxnet> glxnet.cross <- calc.genoprob(glxnet.cross)

glxnet> set.seed(1234)

glxnet> glxnet.cross <- sim.geno(glxnet.cross)

glxnet> n.node <- nphe(glxnet.cross) - 2 ## Last two are age and sex.

glxnet> markers <- glxnet.qtl <- vector("list", n.node)

glxnet> for(i in 1:n.node) {
glxnet+     ac <- model.matrix(~ age + sex, glxnet.cross$pheno)[, -1]
glxnet+     ss <- summary(scanone(glxnet.cross, pheno.col = i,
glxnet+                           addcovar = ac, intcovar = ac[,2]),
glxnet+                   threshold = 2.999)
glxnet+     glxnet.qtl[[i]] <- makeqtl(glxnet.cross, chr = ss$chr, pos = ss$pos)
glxnet+     markers[[i]] <- find.marker(glxnet.cross, chr = ss$chr, pos = ss$pos)
glxnet+ }

glxnet> names(glxnet.qtl) <- names(markers) <- names(glxnet.cross$pheno)[seq(n.node)]

glxnet> glxnet.qdg <- qdg(cross=glxnet.cross,
glxnet+                 phenotype.names = names(glxnet.cross$pheno[,seq(n.node)]),
glxnet+                 marker.names = markers,
glxnet+                 QTL = glxnet.qtl,
glxnet+                 alpha = 0.05,
glxnet+                 n.qdg.random.starts=10,
glxnet+                 addcov="age",
glxnet+                 intcov="sex",
glxnet+                 skel.method="udgskel",
glxnet+                 udg.order=6)

glxnet> glxnet.qdg
$UDG
     node1   node2 edge
1      Glx Slc38a3    0
2      Glx     Ivd    0
3      Glx  Slc1a2    1
4      Glx    Ass1    0
5      Glx    Arg1    0
6      Glx    Pck1    0
7      Glx    Agxt    1
8  Slc38a3     Ivd    0
9  Slc38a3  Slc1a2    0
10 Slc38a3    Ass1    0
11 Slc38a3    Arg1    0
12 Slc38a3    Pck1    0
13 Slc38a3    Agxt    0
14     Ivd  Slc1a2    1
15     Ivd    Ass1    0
16     Ivd    Arg1    0
17     Ivd    Pck1    0
18     Ivd    Agxt    1
19  Slc1a2    Ass1    0
```

```
20  Slc1a2    Arg1    0
21  Slc1a2    Pck1    0
22  Slc1a2    Agxt    0
23    Ass1    Arg1    0
24    Ass1    Pck1    0
25    Ass1    Agxt    0
26    Arg1    Pck1    1
27    Arg1    Agxt    1
28    Pck1    Agxt    0

$DG
  node1 direction  node2  lod score
1   Glx     ----> Slc1a2  0.3464680
2   Glx     ---->   Agxt  1.5834015
3   Ivd     ----> Slc1a2  2.5655168
4   Ivd     ---->   Agxt  1.8999843
5  Arg1     <----   Pck1 -0.3165180
6  Arg1     <----   Agxt -0.5102432

$best.lm
[1] 1

$Solutions
$Solutions$solutions
$Solutions$solutions[[1]]
  node1 direction  node2         lod
1   Glx     ----> Slc1a2  0.08870972
2   Glx     ---->   Agxt  1.20241212
3   Ivd     ----> Slc1a2  2.30775847
4   Ivd     ---->   Agxt  1.51899498
5  Arg1     ---->   Pck1  1.60774597
6  Arg1     <----   Agxt -2.02572245


$Solutions$loglikelihood
[1] 280.6703

$Solutions$BIC
[1] 15.24228


$marker.names
$marker.names$Glx
[1] "D2Mit51"  "D4Mit190" "D5Mit183" "D7Mit117" "D9Mit182" "D13Mit76"

$marker.names$Slc38a3
[1] "D8Mit45"

$marker.names$Ivd
[1] "D2Mit106" "D8Mit45"  "D13Mit91"

$marker.names$Slc1a2
[1] "D2Mit395"  "D9Mit20"   "D18Mit177"
```

```
$marker.names$Ass1
[1] "D2Mit263"  "D4Mit190"  "D5Mit240"  "D8Mit249"  "D15Mit252"

$marker.names$Arg1
[1] "D1Mit64"  "D2Mit263" "D9Mit207"

$marker.names$Pck1
[1] "D4Mit37"   "D10Mit233"

$marker.names$Agxt
[1] "D2Mit411"  "D7Mit294"  "D14Mit126"


$phenotype.names
[1] "Glx"     "Slc38a3" "Ivd"     "Slc1a2"  "Ass1"    "Arg1"    "Pck1"
[8] "Agxt"

$addcov
[1] "age"

attr(,"class")
[1] "qdg"  "list"

glxnet> gr <- graph.qdg(glxnet.qdg)

glxnet> plot(gr)

glxnet> ## Or use tkplot().
glxnet> ## Not run:
glxnet> ##D glxnet.cross <- clean(glxnet.cross)
glxnet> ##D save(glxnet.cross, glxnet.qdg, glxnet.qtl, file = "glxnet.RData", compress = TRUE)
glxnet> ## End(Not run)
glxnet>
glxnet>
glxnet>
```



2 QDG routines

The QDG routines are now incorporated into R/qtlnet. This document shows how to generate data, fit a QDG model and plot the inferred graph. We focus on a simple graph, y1 -> y3, y2 -> y3 and y3 -> y4, with QTLs that affect each of the three phenotypes.

```
> library(qtlnet)
```

Simulate a genetic map (20 autosomes, 10 not equaly spaced markers per chromosome).

```
> mymap <- sim.map(len=rep(100,20), n.mar=10, eq.spacing=FALSE, include.x=FALSE)
```

Simulate an F2 cross object with n.ind (number of individuals).

```
> n.ind <- 200
> mycross <- sim.cross(map=mymap, n.ind=n.ind, type="f2")
```

Produce multiple imputations of genotypes using the sim.geno function. The makeqtl function requires it, even though we are doing only one imputation (since we don't have missing data and we are using the genotypes in the markers, one imputation is enough).

```
> mycross <- sim.geno(mycross,n.draws=1)
```

Use 2 markers per phenotype, samples from the cross.

```
> genotypes <- pull.geno(mycross)
> geno.names <- dimnames(genotypes)[[2]]
> m1 <- sample(geno.names,2,replace=FALSE)
> m2 <- sample(geno.names,2,replace=FALSE)
> m3 <- sample(geno.names,2,replace=FALSE)
> m4 <- sample(geno.names,2,replace=FALSE)
> ## get marker genotypes
> g11 <- genotypes[,m1[1]]; g12 <- genotypes[,m1[2]]
> g21 <- genotypes[,m2[1]]; g22 <- genotypes[,m2[2]]
> g31 <- genotypes[,m3[1]]; g32 <- genotypes[,m3[2]]
> g41 <- genotypes[,m4[1]]; g42 <- genotypes[,m4[2]]
> ## generate phenotypes
> y1 <- runif(3,0.5,1)[g11] + runif(3,0.5,1)[g12] + rnorm(n.ind)
> y2 <- runif(3,0.5,1)[g21] + runif(3,0.5,1)[g22] + rnorm(n.ind)
> y3 <- runif(1,0.5,1) * y1 +  runif(1,0.5,1) * y2 + runif(3,0.5,1)[g31] + runif(3,0.5,1)[g32] + rnorm(n
> y4 <- runif(1,0.5,1) * y3 + runif(3,0.5,1)[g41] + runif(3,0.5,1)[g42] + rnorm(n.ind)
```

Incorporate phenotypes into cross object.

```
> mycross$pheno <- data.frame(y1,y2,y3,y4)
```

Create markers list.

```
> markers <- list(m1,m2,m3,m4)
> names(markers) <- c("y1","y2","y3","y4")
```

Create qtl object.

```
> allqtls <- list()
> m1.pos <- find.markerpos(mycross, m1)
> allqtls[[1]] <- makeqtl(mycross, chr = m1.pos[,"chr"], pos = m1.pos[,"pos"])
> m2.pos <- find.markerpos(mycross, m2)
> allqtls[[2]] <- makeqtl(mycross, chr = m2.pos[,"chr"], pos = m2.pos[,"pos"])
> m3.pos <- find.markerpos(mycross, m3)
> allqtls[[3]] <- makeqtl(mycross, chr = m3.pos[,"chr"], pos = m3.pos[,"pos"])
> m4.pos <- find.markerpos(mycross, m4)
> allqtls[[4]] <- makeqtl(mycross, chr = m4.pos[,"chr"], pos = m4.pos[,"pos"])
> names(allqtls) <- c("y1","y2","y3","y4")
```

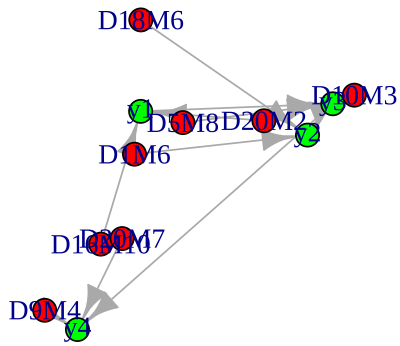
Infer QDG object.

```
> out <- qdg(cross=mycross,
+            phenotype.names = c("y1","y2","y3","y4"),
+            marker.names = markers,
+            QTL = allqtls,
+            alpha = 0.005,
+            n.qdg.random.starts=10,
+            skel.method="pcskel")
> out

$UDG
  node1 node2 edge
1    y1    y3    1
2    y2    y3    1
```

```
5    y3    y4    1

$DG
  node1 direction node2  lod score
1    y1     ---->    y3  0.7041795
2    y2     <----    y3 -0.2555218
3    y3     ---->    y4  1.9634322

$best.lm
[1] 1

$Solutions
$Solutions$solutions
$Solutions$solutions[[1]]
  node1 direction node2       lod
1    y1     ---->    y3  4.216798
2    y2     ---->    y3  3.257096
3    y3     ---->    y4 17.090945


$Solutions$loglikelihood
[1] -1129.342

$Solutions$BIC
[1] 2401.739


$marker.names
$marker.names$y1
[1] "D20M2"  "D16M10"

$marker.names$y2
[1] "D18M6" "D1M6"

$marker.names$y3
[1] "D5M8"  "D10M3"

$marker.names$y4
[1] "D9M4"  "D20M7"


$phenotype.names
[1] "y1" "y2" "y3" "y4"

attr(,"class")
[1] "qdg"  "list"
```

Plot object. The graph is an object of class igraph, which can be plotted using the igraph package.

```
> graph <- graph.qdg(out)
> plot(graph)
```



You can use tkplot() for an interactive plot.